01010
01010
01010 *information*

Special Issue Reprint

Computational Linguistics and Natural Language Processing

Edited by
Peter Z. Revesz

mdpi.com/journal/information

MDPI

Computational Linguistics and Natural Language Processing

Computational Linguistics and Natural Language Processing

Editor

Peter Z. Revesz

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Editor
Peter Z. Revesz
Department of Classics and
Religious Studies, College of
Arts and Sciences, and School
of Computing, College of
Engineering, University of
Nebraska-Lincoln
Lincoln, NE
USA

Editorial Office
MDPI
St. Alban-Anlage 66
4052 Basel, Switzerland

This is a reprint of articles from the Special Issue published online in the open access journal *Information* (ISSN 2078-2489) (available at: https://www.mdpi.com/journal/information/special_issues/Computational_CLNLP2021).

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-1369-8 (Hbk)
ISBN 978-3-7258-1370-4 (PDF)
doi.org/10.3390/books978-3-7258-1370-4

Contents

About the Editor

Peter Z. Revesz

Peter Z. Revesz earned a B.S. summa cum laude with a double major in Computer Science and Mathematics from Tulane University and a Ph.D. in Computer Science from Brown University (dissertation title: Constraint Query Languages; advisor: Paris C. Kanellakis). He was a postdoctoral fellow at the University of Toronto before joining the University of Nebraska-Lincoln, where he is a professor in the School of Computing and the Department of Classics and Religious Studies. He is an expert in computational linguistics, databases, bioinformatics, and geoinformatics. He is the author of *Introduction to Databases: From Biological to Spatio-Temporal* (Springer, 2010) and *Introduction to Constraint Databases* (Springer, 2002). Dr. Revesz has held visiting appointments at the IBM T.J. Watson Research Center, INRIA, the Max Planck Institute for Computer Science, the University of Athens, the University of Hasselt, the University of Helsinki, the U.S. Air Force Office of Scientific Research, and the U.S. Department of State. He is a recipient of an AAAS Science & Technology Policy Fellowship, a J. William Fulbright Scholarship, an Alexander von Humboldt Research Fellowship, a Jefferson Science Fellowship, a National Science Foundation CAREER award, and a Faculty International Scholar of the Year award by Phi Beta Delta, the Honor Society for International Scholars.

Preface

The aim of this Special Issue is to present the latest research on computational linguistics and natural language processing, especially in the areas of sentiment analysis, linguistic profiling, higher-order logical representation, and computational methods for the decipherment of various scripts. The authors are leading experts in these areas from various countries and continents. The readers can apply these ideas to various applications, such as measuring the sentiments of customers, characterizing, or sometimes even identifying, the likely authors of anonymous texts, aiding the diagnoses of neuropsychiatric diseases, improving communication with chatbots, and deciphering encrypted texts. The cutting-edge ideas and open problems presented in this Special Issue will spark additional ideas on the part of researchers who would like to explore novel topics in computational linguistics and natural language processing.

Peter Z. Revesz
Editor

information

Editorial

Preface to the Special Issue on Computational Linguistics and Natural Language Processing

Peter Z. Revesz

Department of Classics and Religious Studies, College of Arts and Sciences, and School of Computing, College of Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA; peter.revesz@unl.edu

Computational linguistics and natural language processing are at the heart of the AI revolution that is currently transforming our lives. We are witnessing the growth of these areas to the point that intelligent, talking robots can now perform many jobs that humans used to do. We are encountering robots in an increasing number of situations. For example, it is becoming common to see robots answer customer inquiries at call centers, replace cashiers with automated talking checkout machine at stores, look up a given address on an online map to plan a path and then autonomously navigate a car to its intended destination, assemble complex products while making decisions according to the particularities of supply and workflow demands in factories, and monitor access to buildings and sound an alarm if a dangerous situation develops. Robots have become such an integral part of our daily lives that while using the internet, people are required to take the "I am not a robot" test on a regular basis. This ambiguity resulting from the difficulty to distinguish humans and robots means that robots have acquired the capacity to replace humans.

In addition, intelligent and talking robots are increasingly used, even in situations and in places that are not as visible for most citizens. This includes various security systems, where robots are analyzing online conversations and chats. Some security robots also make decisions about potential dangers, such as possible illegal drug smuggling or acts of violence. Coupled with sophisticated drone systems, intelligent robots can assist humans by carrying out missions that require flying or being in outer space.

All of these machines need some way to communicate their contributions back to humans for the advancement of the human civilization. This is often best done by using human language, either written or spoken. Thus, the building of useful robots comes with the need to make the robot capable of using and analyzing human language. Therefore, the study of computational linguistics and natural language processing is a foundational part of the AI revolution that is presently resounding in our midst. It is in the light of these sentiments that this Special Issue on computational linguistics and natural language processing was called for, written, and assembled.

There is no doubt that computational linguistics and natural language processing facilitate not only major technological transformations but also influence social transformations. We increasingly live in what a few decades ago would have been termed a sci-fi world. These transformations come with certain challenges, but those challenges need not be feared because the opportunities outweigh the downside for humanity. That is also the overall message of the paper "Analyzing Sentiments Regarding ChatGPT Using Novel BERT: A Machine Learning Approach" by Sudheesh R. et al. [1] in this Special Issue, which is based on selected papers from the International Conference on Computational Linguistics and Natural Language Processing held in Beijing in December 2021. We also invited additional papers on the same topics, and they also underwent a rigorous review process.

Linguistic Profiling

Many of the papers proposed various novel methods of linguistic profiling and categorizing texts. The paper "Computing the Sound–Sense Harmony: A Case Study of William Shakespeare's Sonnets and Francis Webb's Most Popular Poems" by Delmonte [2]

Citation: Revesz, P.Z. Preface to the Special Issue on Computational Linguistics and Natural Language Processing. *Information* **2024**, *15*, 281. https://doi.org/10.3390/info15050281

Received: 28 April 2024
Accepted: 7 May 2024
Published: 15 May 2024

proposes a novel sentiment analysis metric. The main idea is that sounds have presumed meanings, for example, they can be happy and sad. The actual text of a poem also expresses a meaning with a happy or sad connotation. Delmonte's sentiment analysis shows that both Shakespeare and Webb carefully chose their words to make the presumed meaning of the words' sounds and the sentences' explicit meaning be in harmony, or sometimes in disharmony when irony was intended.

The paper "Morphosyntactic Annotation in Literary Stylometry" by Gorman [3] provides a sophisticated stylometric analysis of texts. This computational analysis uses morphosyntactic annotations, such as the frequency of various pronouns, verbal cases, and the ordering of various elements of the sentence. Interestingly, famous authors are demonstrated to have a unique style because they can be identified by their stylometric profile with a very high probability. So, if we take a short quotation from one of their books, then they can be almost 100 percent correctly identified as the authors of their other books. For example, if we take a quotation from the book Oliver Twist by Charles Dickens, then we can identify that he also wrote A Christmas Carol because of the stylometric similarities between the two novels.

The above papers pose the prospect of intelligent robots imitating famous writers by simply adhering to some sentiment analysis measures and stylometric profiles as described in the two previous papers. The good news is that the paper "A Benchmark Dataset to Distinguish Human-Written and Machine-Generated Scientific Papers" by Abdalla et al. [4] proposes a practical approach to distinguish between human-written and machine-generated texts. While the paper focuses on identifying fake scientific papers, many of the proposed techniques can be applied to other types of texts too.

The paper "A Survey on Using Linguistic Markers for Diagnosing Neuropsychiatric Disorders with Artificial Intelligence" by Zaman and Trausan-Matu [5] is also concerned with the categorization of spoken language and written text. The goal of their paper is to aid medical diagnoses by identifying these linguistics markers, including sentiment analysis and stylometric measures, that characterize various mental illnesses. The paper also provides a comprehensive review of this growing subject with an extensive bibliography.

The paper "Linguistic Profiling of Text Genres: An Exploration of Fictional vs. Non-Fictional Texts" by Mendhakar [6] applies linguistic profiling to distinguish between texts that describe fiction versus those that describe real events. Various types of fiction are also categorized, such as fables, myths, mystery, romance, thriller, legends, and science fiction. Non-fiction works are also more finely divided into discussions, explanations, instructions, and persuasions.

The paper "A Literature Survey on Word Sense Disambiguation for the Hindi Language" by Gujjar et al. [7] focuses on the process to determine the exact context-specific meanings of ambiguous words, for example, the English word bark, which can mean the sound emitted by dogs, the outer sheath of a tree trunk, or a kind of ship. While many natural language processing techniques were developed to deal with the disambiguation of English words, the disambiguation of Hindi words sometimes requires language-specific algorithms that are reviewed in this paper.

Higher-Order Logical Representations and Methods

There were some papers that were concerned with the internal computer representation of texts and images. The paper "Agile Logical Semantics for Natural Languages" by Manca [8] introduces predicate abstraction as a new operator, which is argued to be a natural operator when some form of monadic high-order logic is used to express the semantics of linguistic structures. A possible application of predicate abstraction could be to teach more abstract logical thinking to chatbots, such as ChatGPT. For example, the author details a conversation with ChatGPT where ChatGPT was able to learn the predicate abstraction that "Go" and "Goes" represent the same predicate but in different grammatical forms.

Context-free Lindenmayer systems (D0L systems) have been used to describe the generation of growing plants, cities, and fractals, among other applications. Context-free Lindenmayer systems can be viewed as an extension of context-free grammars where the rewriting rules include commands such as ‘draw a line’, ‘turn a specific degree’, and ‘go to a specific position’. They form a special type of language that is studied in the paper “D0L-System Inference from a Single Sequence with a Genetic Algorithm” by Łabędzki and Unold [9]. The aim of this paper is to essentially reverse engineer a context-free Lindenmayer system, that is, when given an image that was generated by a context-free Lindenmayer system, to then find its grammar. The authors demonstrate that their genetic algorithmic method finds satisfying solutions on various types of images, such as binary trees, Barnsley ferns, Koch snowflakes, and Sierpiński triangles.

Deciphering Scripts

A group of papers were concerned with the problem of deciphering inscriptions. The paper “Minoan Cryptanalysis: Computational Approaches to Deciphering Linear A and Assessing its Connections with Language Families from the Mediterranean and the Black Sea Areas” by Nepal and Perono Cacciafoco [10] provides a linguistic analysis of Linear A inscriptions, which were written by Minoan scribes during the Bronze Age, mainly on the island of Crete. The linguistic analysis uses the feature-based comparison of signs method introduced in Revesz [11]. However, in their study, Nepal and Perono Cacciafoco [9] obtain slightly different sign matches between Linear A signs, Carian alphabet letters, and Cypriot syllabary signs than Revesz [11] obtained, because they use different weights for the various features. The matches provide candidate phonetic values for Linear A signs. This allows for a phonetic transcription of Linear A inscriptions to be carried out.

Next, they applied a linguistic analysis of the Linear A inscriptions by finding possible words from the following languages: Ancient Egyptian, Hittite, Luwian, Proto-Celtic, and Proto-Uralic. The latter two languages were chosen because they may have been spoken on the coastal areas of the Black Sea, which has been shown to be the likely source of some Minoans [12,13]. The analysis yielded eight Ancient Egyptian, nine Hittite, seven Luwian, eleven Proto-Celtic, and twelve Proto-Uralic words as good matches with the Linear A inscriptions. While the analysis of Nepal and Perono Cacciafoco [10] is inconclusive in deciding the underlying language of the Linear A inscriptions, it nicely demonstrates that it was premature of many earlier authors to focus their attention only on Mediterranean languages, ignoring the fact that the Bosporus and Dardanelles Straits enable easy sailing between the Aegean Sea and the Black Sea areas. The analysis of Nepal and Perono Cacciafoco [10] is compatible with Revesz [11], which provided a translation of twenty-eight Linear A inscriptions into a Uralic language.

The paper “A Proposed Translation of an Altai Mountain Inscription Presumed to Be from the 7th Century BC” by Revesz and Varga [14] originated when someone brought to our attention an inscription from a book by Karžaubaj Sartkožauly, who is a member of the Kazakhstan Academy of Sciences. Sartkožauly presumed the inscription to be from the 7th century BC, and this was also our initial assumption. However, we became increasingly suspicious about the dating of the inscription during our decipherment work. For example, the inscription used a personal woman’s name, Enikő, that was only created in the 19th century, although it became popular afterwards. This paper also proposed two different solutions because there was an ambiguous part in the inscription.

After the study by [14] was published, it received great publicity and became the subject of a popular YouTube video, which happened to be watched by the scribe, Peter Kun, who admitted in a comment below the video that he wrote the inscription while he was visiting the Altai Mountains as a young man. That was a fascinating turn of events because there is no other known case when a scribe practically “came alive” to be able to judge the correctness of the decipherment of a presumably ancient inscription.

The paper “Decipherment Challenges due to Tamga and Letter Mix-Ups in an Old Hungarian Runic Inscription from the Altai Mountains” [15] came as a natural follow up of [14] after contacting Peter Kun. He provided a fascinating explanation of the cryptic section of his inscription, while verifying the correctness of the rest of our decipherment in [14]. I thought that the readers deserve to learn about the entire correct decipherment. In addition, ref. [15] also provides a mathematical analysis of the process of incorrectly mixing up some visually similar signs by Peter Kun. Since such mix-ups are frequent in other inscriptions too, this mathematical analysis may benefit other scholars who are working on the decipherment of ancient scripts.

Finally, I would like to thank the many reviewers who have reviewed the papers submitted to this Special Issue. I also would like to thank Janessy Zhan, Section Managing Editor at MDPI of this Special Issue, for her outstanding help in every aspect of organization, including arranging independent reviewers of my contributions to this Special Issue. I am also grateful to all parties that made contributions to this Special Issue. It was great to work with such a talented group of authors, and I wish them much success in their future research.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Sudheesh, R.; Mujahid, M.; Rustam, F.; Shafique, R.; Chunduri, V.; Villar, M.G.; Ballester, J.B.; de la Torre Diez, I.; Ashraf, I. Analyzing Sentiments Regarding ChatGPT Using Novel BERT: A Machine Learning Approach. *Information* **2023**, *14*, 474.
2. Delmonte, R. Computing the Sound–Sense Harmony: A Case Study of William Shakespeare's Sonnets and Francis Webb's Most Popular Poems. *Information* **2023**, *14*, 576. [CrossRef]
3. Gorman, R. Morphosyntactic Annotation in Literary Stylometry. *Information* **2024**, *15*, 211. [CrossRef]
4. Abdalla, M.H.I.; Malberg, S.; Dementieva, D.; Mosca, E.; Groh, G. A Benchmark Dataset to Distinguish Human-Written and Machine-Generated Scientific Papers. *Information* **2023**, *14*, 522. [CrossRef]
5. Zaman, I.-R.; Trausan-Matu, S. A Survey on Using Linguistic Markers for Diagnosing Neuropsychiatric Disorders with Artificial Intelligence. *Information* **2024**, *15*, 123. [CrossRef]
6. Mendhakar, A. Linguistic Profiling of Text Genres: An Exploration of Fictional vs. Non-Fictional Texts. *Information* **2023**, *13*, 357. [CrossRef]
7. Gujjar, V.; Mago, N.; Kumari, R.; Patel, S.; Chintalapudi, N.; Battineni, G. A Literature Survey on Word Sense Disambiguation for the Hindi Language. *Information* **2022**, *14*, 495. [CrossRef]
8. Manca, V. Agile Logical Semantics for Natural Languages. *Information* **2024**, *15*, 64. [CrossRef]
9. Łabędzki, M.; Unold, O. D0L-System Inference from a Single Sequence with a Genetic Algorithm. *Information* **2023**, *14*, 343. [CrossRef]
10. Nepal, A.; Perono Cacciafoco, F. Minoan Cryptanalysis: Computational Approaches to Deciphering Linear A and Assessing its Connections with Language Families from the Mediterranean and the Black Sea Areas. *Information* **2024**, *15*, 73. [CrossRef]
11. Revesz, P.Z. Establishing the West-Ugric Language Family with Minoan, Hattic and Hungarian by a Decipherment of Linear A. *WSEAS Trans. Inf. Sci. Appl.* **2017**, *14*, 306–335.
12. Revesz, P.Z. Data Mining Autosomal Archaeogenetic Data to Determine Minoan Origins. In Proceedings of the 25th International Database Engineering & Applications Symposium, New York, NY, USA, 7 September 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 46–55.
13. Revesz, P.Z. Minoan archaeogenetic data mining reveals Danube Basin and western Black Sea littoral origin. *Int. J. Biol. Biomed. Eng.* **2019**, *13*, 108–120.
14. Revesz, P.Z.; Varga, G. A Proposed Translation of an Altai Mountain Inscription Presumed to be from the 7th Century BC. *Information* **2022**, *13*, 243. [CrossRef]
15. Revesz, P.Z. Decipherment Challenges due to Tamga and Letter Mix-Ups in an Old Hungarian Runic Inscription from the Altai Mountains. *Information* **2023**, *13*, 422. [CrossRef]

information

Article

Analyzing Sentiments Regarding ChatGPT Using Novel BERT: A Machine Learning Approach

Sudheesh R [1,†], Muhammad Mujahid [2,†], Furqan Rustam [3,†], Rahman Shafique [4], Venkata Chunduri [5], Mónica Gracia Villar [6,7,8], Julién Brito Ballester [6,9,10], Isabel de la Torre Diez [11,*] and Imran Ashraf [4,*]

1 Kodiyattu Veedu, Kollam, Valakom 691532, India; sudheeshrofficial@gmail.com
2 Department of Computer Science, Khwaja Fareed University of Engineering and Information Technology, Rahim Yar Khan 64200, Pakistan; mujahidws890@gmail.com
3 School of Computer Science, University College Dublin, D04 V1W8 Dublin, Ireland; furqan.rustam1@gmail.com
4 Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Republic of Korea; rahmanshafique47@gmail.com
5 Indiana State University, Terre Haute, IN 47809, USA; chunduriv1@gmail.com
6 Faculty of Social Science and Humanities, Universidad Europea del Atlántico, Isabel Torres 21, 39011 Santander, Spain; monica.gracia@uneatlantico.es (M.G.V.); julien.brito@uneatlantico.es (J.B.B.)
7 Department of Project Management, Universidad Internacional Iberoamericana Arecibo, Puerto Rico, PR 00613, USA
8 Department of Extension, Universidade Internacional do Cuanza, Cuito EN250, Bié, Angola
9 Universidad Internacional Iberoamericana, Campeche 24560, Mexico
10 Universitaria Internacional de Colombia, Bogotá 11001, Colombia
11 Department of Signal Theory, Communications and Telematics Engineering, Unviersity of Valladolid, Paseo de Belén, 15, 47011 Valladolid, Spain
* Correspondence: isator@tel.uva.es (I.d.l.T.D.); imranashraf@ynu.ac.kr (I.A.)
† These authors contributed equally to this work.

Citation: R, S.; Mujahid, M.; Rustam, F.; Shafique, R.; Chunduri, V.; Villar, M.G.; Ballester, J.B.; Diez, I.d.l.T.; Ashraf, I. Analyzing Sentiments Regarding ChatGPT Using Novel BERT: A Machine Learning Approach. *Information* **2023**, *14*, 474. https://doi.org/10.3390/info14090474

Academic Editor: Peter Revesz

Received: 11 July 2023
Revised: 15 August 2023
Accepted: 19 August 2023
Published: 25 August 2023

Abstract: Chatbots are AI-powered programs designed to replicate human conversation. They are capable of performing a wide range of tasks, including answering questions, offering directions, controlling smart home thermostats, and playing music, among other functions. ChatGPT is a popular AI-based chatbot that generates meaningful responses to queries, aiding people in learning. While some individuals support ChatGPT, others view it as a disruptive tool in the field of education. Discussions about this tool can be found across different social media platforms. Analyzing the sentiment of such social media data, which comprises people's opinions, is crucial for assessing public sentiment regarding the success and shortcomings of such tools. This study performs a sentiment analysis and topic modeling on ChatGPT-based tweets. ChatGPT-based tweets are the author's extracted tweets from Twitter using ChatGPT hashtags, where users share their reviews and opinions about ChatGPT, providing a reference to the thoughts expressed by users in their tweets. The Latent Dirichlet Allocation (LDA) approach is employed to identify the most frequently discussed topics in relation to ChatGPT tweets. For the sentiment analysis, a deep transformer-based Bidirectional Encoder Representations from Transformers (BERT) model with three dense layers of neural networks is proposed. Additionally, machine and deep learning models with fine-tuned parameters are utilized for a comparative analysis. Experimental results demonstrate the superior performance of the proposed BERT model, achieving an accuracy of 96.49%.

Keywords: ChatGPT; sentimental analysis; BERT; machine learning; LDA; app reviewers; deep learning

1. Introduction

AI-based chatbots, powered by natural language processing (NLP), are computer programs designed to simulate human interactions by understanding speech and generating human-like responses [1]. They have gained popularity across various industries as a tool to enhance digital experiences. The utilization of chatbots is experiencing continuous growth, with predictions indicating that the chatbot industry is expected to reach a market size of $3.62 billion by 2030, accompanied by an annual growth rate of 23.9% [2]. Additionally, the chatbot market is expected to reach approximately 1.25 billion U.S. dollars by 2025 [3]. The adoption of chatbots in sectors such as education, healthcare, banking, and retail is estimated to save around $11 billion annually by 2023 [4]. Especially in recent developments in the field of education, chatbots have the potential to significantly enhance the learning experience for students.

ChatGPT, an AI-based chatbot that is currently gaining attention, is being discussed widely across various platforms [5–7]. It has become a prominent topic of conversation due to its ability to provide personalized support and guidance to students, contributing to an improved academic performance. Developed by OpenAI, ChatGPT utilizes advanced language generation techniques based on the GPT language model technology [8]. Its impressive capabilities in generating coherent and contextually relevant responses have captivated individuals, communities, and social media platforms. The widespread discussions surrounding ChatGPT highlight its significant impact on natural language processing and artificial intelligence, and its potential to revolutionize our interactions with AI systems. People are fascinated by its usefulness in various domains including learning, entertainment, and problem-solving, which further contributes to its popularity and widespread adoption.

While there are many advantages to using ChatGPT, there are also some notable disadvantages and criticisms of the AI chatbot. Some raised concerns include the potential for academic dishonesty, as ChatGPT could be used as a tool for cheating in educational settings, similar to using search engines like Google [9]. There is also a concern that ChatGPT may perpetuate biases when used in research, as the language model is trained on large amounts of data that may contain biased information [9]. Another topic of discussion revolves around the potential impact of ChatGPT on students' critical thinking and creativity. Some argue that an over-reliance on ChatGPT may lead to a decline in these important skills among students [10]. Additionally, the impact of ChatGPT on the online education business has been evident, as seen in the case of Chegg Inc., where the rise of ChatGPT contributed to a significant decline of 47% in the company's shares during early trading [11]. To gain insights into people's perceptions of ChatGPT, opinion mining was conducted using social media data. This analysis aimed to understand the general sentiment and opinions surrounding the use of ChatGPT in various contexts: people, in this sense, tweet on Twitter concerning their thoughts about ChatGPT, which could provide valuable information.

Opinion mining involves evaluating individuals' perspectives, attitudes, evaluations, and emotions towards various objects including products, services, companies, individuals, events, topics, occurrences, and applications, along with their attributes. When making decisions, we often seek the opinions of others, whether as individuals or organizations. Sentiment analysis tools have found application in diverse social and corporate contexts [12]. Social media platforms, microblogging sites, and app stores serve as rich sources of openly expressed opinions and discussions, making them valuable for a sentiment analysis [13]. The sentiment analysis employs NLP, a text analysis, and computational methods such as machine learning and data mining to automate the categorization of sentiments based on feedback and reviews [14]. The sentiment analysis process involves identifying sentiment from reviews, selecting relevant features, and performing sentiment classification to determine polarity.

1.1. Research Questions

To meet the objective of this study by analyzing people's attitudes toward ChatGPT, this study formulates the following questions (RQs):

i. **RQ1:** What are people's sentiments about ChatGPT technology?

ii. **RQ2:** Which classification model is most effective, such as the proposed transformer-based models, machine learning-based models, and deep learning-based models, for analyzing sentiments about ChatGPT tweets?
iii. **RQ3:** What are the impacts of ChatGPT on student learning?
iv. **RQ4:** What role does topic modeling play in the sentiment analysis of social media tweets?

1.2. Contributions

The sentiment analysis of tweets regarding ChatGPT aims at providing users' perceptions of ChatGPT and analyzing the ratio of positive and negative comments from users. In addition, a topic analysis can provide insights on frequently discussed topics concerning ChatGPT and provide feedback to further improve its functionality. In particular, the following contributions are made:

- This study aims to analyze people's perceptions of the trending topic of ChatGPT worldwide. The research contributes by collecting relevant data and examining the sentiments expressed by individuals toward this significant development.
- Tweets related to ChatGPT are collected by utilizing the Tweepy application programming interface (API) and employing various keywords. The collected tweets undergo preprocessing and annotation using Textblob and the valence aware-dictionary (VADER). The bag of words (BoW) feature engineering technique is employed to extract essential features.
- A deep transformer-based BERT model is proposed for the sentiment analysis. It consists of three dense layers of neural networks for enhanced performance. Additionally, machine learning and deep learning models with fine-tuned parameters are utilized for comparison purposes. Notably, this study is the first to investigate ChatGPT raw tweets using Transformers.
- The study utilizes the latent Dirichlet allocation (LDA) approach to extract highly discussed topics from the dataset of ChatGPT tweets. This analysis provides valuable insights into the frequently discussed themes and subjects.

The remaining sections of the paper are structured as follows: Section 2 provides a comprehensive review of relevant research works on sentiment analyses, offering a valuable background for the proposed approach. Section 3 presents a detailed description of the proposed approach. Section 4 presents and discusses the experimental results obtained from the analysis. Finally, Section 5 concludes the study, summarizing the key findings and suggesting potential directions for future research.

2. Related Work

The analysis of reviews has gained significant attention in recent years, mainly due to the widespread use of social media platforms. These platforms serve as a hub for discussions on various topics, providing researchers with valuable insights and information. For instance, in a study conducted by Lee et al. [15], social media data were utilized to investigate the Taliban's control over Afghanistan. By analyzing the discussions and conversations on social media, the study aimed to gain a deeper understanding of the situation. Similarly, the study by Lee et al. [16] focused on extracting tweets related to racism to shed light on the issue of racism in the workplace. By analyzing these tweets, the researchers aimed to uncover patterns and gain insights into the prevalence and nature of racism in professional environments. They utilized Twitter data and annotated it with the TextBlob approach. The authors attained 72% accuracy for the racism classification. In a different context, Mujahid et al. [17] conducted a study on public opinion about online education during the COVID-19 pandemic. By analyzing social media data, the researchers aimed to understand the sentiment and perceptions surrounding online education during this challenging time. These studies highlight the significance of a social media data analysis in extracting meaningful information and gaining insights into various subjects. By harnessing the vast amount of discussions and conversations on social media platforms, researchers can delve into important topics and uncover valuable findings. The researchers employed 17,155 tweets for the analysis and attained 95% accuracy using the SMOTE technique with bag of word features by the SVM model.

ChatGPT is a hot topic nowadays and exploring people's perceptions about it using Twitter data can provide valuable insights. Many studies have previously done such kinds of analyses on different topics. In the study conducted by Tran et al. [18], the focus was on examining consumer sentiments towards chatbots in various retail sectors and investigating the impact of chatbots on their sentiments and expectations regarding interactions with human agents. Through the application of the automated sentiment analysis, it was observed that the general sentiment towards chatbots is more positive compared to that towards human agents in online settings. They collected a limited dataset of 8190 tweets and used ANCOVA for the test. They only classify the tweets into their exact sentiments and do not properly use performance metrics like accuracy. Additionally, sentiments varied across different sectors, such as fashion and telecommunications, with the implementation of chatbots resulting in more negative sentiments towards human agents in both sectors. The study [19] aimed to develop an effective system for analyzing and extracting sentiments and mental health during the COVID-19 pandemic. By utilizing a vast amount of data and leveraging hashtags, we employed the BERT machine learning algorithm to classify customer perspectives into positive and negative sentiments with high accuracy. Ensuring user privacy, our main objective was to facilitate self-understanding and the regulation of mental states through end-to-end encrypted user-bot interactions. The researchers were able to achieve 95.6% accuracy and 95% recall for automated sentiment classification related to chatbots.

Some studies, such as [20], focus on a sentiment analysis of disaster-related tweets at different time intervals for specific locations. By using the LSTM network with word embedding, keywords are derived from the tweet history and context. The proposed algorithm, RASA, classifies tweets and identifies sentiment scores for each location. RASA outperforms other algorithms, aiding the government in post-disaster management by providing valuable insights and preventive measures. Another study [21] tries to predict cryptocurrency prices using Twitter data. They focus on a sentiment analysis and emotion detection using tweets related to cryptocurrency. An ensemble model, LSTM-GRU, combines LSTM and GRU to enhance the analysis' accuracy. Multiple features and models, including machine learning and deep learning, are examined. Results reveal a predominance of positive sentiment, with fear and surprise also as prominent emotions. The dataset consists of five emotions extracted from Twitter. The proposed ensemble model achieves 83% accuracy using a balanced dataset for emotion prediction. This research provides valuable insights into the public perception of cryptocurrency and its market implications.

Additionally, it is also observed that most of the time, a service provider asks for feedback regarding the quality or satisfaction level of the services or products via a customer feedback form provided in an online mode, most probably by using a social media platform [22]. Such assessments are critical in determining the quality of services and products. However, it is necessary to examine the views of user concepts and impressions. Negative sentiment ratings, in particular, include more relevant recommendations for enhancing the quality of the product/service. Given the significance of the text analysis, there is a huge amount of work on the sentiment analysis. For example, studies [23–25] classify app reviews by using machine learning and deep learning models. Another piece of research [26] looked at the Shopify app reviews and classified them as pleased or dissatisfied. For sentiment classification, many feature extraction approaches are used in conjunction with supervised machine learning algorithms. For the experiments, 12,760 samples of app reviews were utilized with machine learning. Different hybrid approaches to combining the features were used to enhance the performance. But LR performed with 83% accuracy and an 86% F score. The performance of machine learning models in the sentiment analysis can be influenced by the techniques used for feature engineering. Research studies [27,28] indicate that altering the feature engineering process can result in changes to the models' performance. The research [29] provides a method for categorizing and evaluating employee reviews. For employee review classification, it employs an ETC with BoW features. The study classified employee reviews using both numerical and text elements and achieved 100% and 79% accuracy, respectively. Ref. [30] used NB in conjunction with the RF and SVM to categorize mobile app reviews from the Google Play store. The researcher collected over 90,000 reviews posted in the English language for 10 applications available on the Google Play Store. A total of 7500 reviews were annotated from a dataset of 90,000 tweets. The final experiments implemented the use of 7500 reviews. The results indicated that a baseline 10-fold validation yielded an accuracy of 90.8%. Additionally, the precision was found to be 88%, the recall was 91%, and the f score was 89%. Ref. [31] also used an RF algorithm to identify the variables that distinguish reviews from those from other nations. The research [32] looked at retail applications in Bangladesh. The authors gathered data from Google Play and utilized VADER and AFFIN to annotate sentiments. For sentiment categorization, many machine learning models are employed, and RF outperforms with substantial accuracy. Bello et al. [33] proposed a BERT model for a sentiment analysis on Twitter data. The authors used the BERT model with different variants including the recurrent neural network (RNN) and Bi-long short-term memory (BILSTM) for classification. Catelli et al. [34] and Patel et al. [35] also employed the BERT model for a sentiment analysis on app reviews with lexicon-based approaches.

The study [36] presented a hybrid approach for the sentiment analysis of ChatGPT tweets. Raw tweets were transformed into structured and normalized forms to improve the accuracy of the model and a lower computing complexity. For the objective of classifying tweets from ChatGPT, the authors developed hybrid models. Although state-of-the-art models are unable to provide correct predictions, hybrid models incorporate multiple models to eliminate bias, improve overall outcomes, and make precise predictions. Bonifazi et al. [37] proposed a framework for determining the spatial and spatio-temporal extent of a user's sentiment regarding a topic on a social network. First, the authors introduced the idea of their research, observing that it summarizes a number of previously discussed ideas about social websites. In reality, each of these ideas represents a unique fact about the concept. Then, they established a framework capable of expressing and controlling a multidimensional view-of scope, which is the sentiment of an individual regarding a topic. After that, they recommended a number of parameters and a method for assessing the spatial and spatio-temporal scope of a user's opinion on a topic on a social platform. They conducted several experiments on actual data collected through Reddit to test the proposed framework. Similarly, Bonifazi et al. [38] presented another Reddit-based study. They proposed a model for evaluating and visualizing the eWoM Power of Reddit blog posts.

In a similar way, ref. [39] examined app reviews, where the authors initially extracted negative reviews, constructed a time series of these reviews, and subsequently trained a model to identify key patterns. Additionally, the study focused on an automatic review classification to address the challenge of handling a large volume of daily submitted reviews. To tackle this, the study presented a multi-label active-learning technique, which yielded superior results compared to state-of-the-art methods. Given the impracticality of manually analyzing a vast number of reviews, many researchers have turned to topic modeling, a technique that aids in identifying the main themes within a given text. For instance, in the study [40], the authors investigated the relationship between Arabic app elements and assessed the accuracy of reflecting the type and genre of Arabic mobile apps available on the Google Play store. By employing the LDA approach, valuable insights were provided, offering potential improvements for the future of Arabic apps. Furthermore, in [41], the authors developed an NB and XGB technique to determine user activity within an app.

The literature review provides an analysis of the advantages, disadvantages, and limitations associated with different approaches. Nevertheless, it is worth noting that a significant number of researchers have directed their attention toward the utilization of Twitter datasets for the purpose of analyzing tweets and app evaluations. The researchers employed natural language processing (NLP) techniques and machine learning primarily for the purpose of a sentiment analysis. Commonly utilized Machine learning models, including random forests, support vector machines, and extra tree classifiers, are limited in their ability to learn intricate patterns and are typically not utilized for large datasets. When the aforementioned models are employed on extensive datasets, their performance is inadequate and demands an excessive amount of time for training, especially in the case of handcrafted features. Furthermore, the existing literature employs a limited collection of datasets, which are only comprised of tweets that are not linked to ChatGPT tweets. Previous research has not extensively examined the topic of ChatGPT or OpenAI-related tweets and achieved a low accuracy. Table 1 shows the summary of the literature review.

Table 1. Summary of related work.

Authors	Techniques	Advantages	Disadvantages	Limitations
[16]	TextBlob, CNN, RNN, GRU, DT, RF, SVM	The authors make an ensemble model by combining the GRU, CNN, and RNN for the extraction of features from the tweets and detection. They also performed seven experiments to test the proposed ensemble approach.	The authors develop ensemble models, which need a significant amount of time to both train and identify the sentiments.	The authors used a limited dataset and did not develop transformer-based models that are the most up-to-date and that provide high accuracy.
[17]	TextBlob, CNN, LSTM, SVM, GBM, KNN, DT, LSTM-CNN	This study employed machine learning as well as deep learning for the analysis of tweets. They utilized various annotation and feature engineering techniques. Machine learning outperformed deep learning with an accuracy of 95%.	The study did not clearly describe the preprocessing stages and their implementations.	The dataset included in this study was restricted to tweets that were not associated with ChatGPT tweets.
[18]	BERT	The authors conducted this research to analyze the depression tweets during the period of COVID-19 and achieved remarkable results with BERT.	To speed up computation, the research did not remove stopwords, punctuation, numerical values, etc., from the text. Additionally, the accuracy was inadequate.	The research only proposed one model, which was BERT, and did not compare with other studies.
[19]	Naïve Bayes	The data in the study was labeled using the Vader technique, and the Nave Bayes model was implemented to examine the influence of chatbots on customer opinions and demands within the retail industry.	The study detected positive, neutral, and negative sentiments and used the Ancova test only for the experiments.	The study did not use the most important metrics like accuracy, deep models, or transformers. The study is limited to the Nave Bayes model.
[21]	LSTM + GRU, CNN, SVM, DT, TFIDF	Their primary area of research revolves around sentiment evaluation and detecting emotions using tweets that are associated with cryptocurrencies. The utilization of an ensemble model, namely the LSTM-GRU model, involves the integration of both LSTM and GRU architectures in order to improve the accuracy of the analysis.	The author used ensemble models, which necessitate substantial time for both training and sentiment identification.	The study is regarding the cryptography analysis. Also, transformers are ignored in this study.
[26]	RF, LR, and AC	The study used various feature engineering strategies, including bag-of-words; term frequency, inverse document-frequency, and Chi-2 are employed individually and collectively in order to attain meaningful information from the tweets.	The study employed various feature engineering strategies but did not use cross-dataset experiments with machine learning classifiers. The LR achieved a 83% lowest accuracy.	The study does not use Chatbots or ChatGPT-related tweets for the experiments. In addition, their focus is on utilizing machine learning models for Shopify reviews.
[30]	SVM, RF, and NB	The dataset was obtained by the authors from the most popular ten applications. The findings of the study revealed that a baseline 10-fold validation approach resulted in an accuracy rate of 90.8%.	The paper is about app reviews, not ChatGPT tweets.	The accuracy achieved is very low, and the study did not use any deep transformers to improve its efficiency.

As a result, this paper proposes a transformer-based BERT model that leverages self-attention mechanisms, which have demonstrated remarkable efficacy in the context of machine learning and deep learning. The proposed model addresses the problems mentioned in the literature review. They have the ability to comprehend the correlation between consecutive items that are widely separated. The transformers achieved an exceptional performance. Additionally, the performance of the proposed method was evaluated using cross-validation findings and statistical tests. The ChatGPT tweets study utilizes BERTopic and LDA-based topic modeling techniques to ascertain the most pertinent topics or keywords within the datasets.

3. Methodology

The proposed methodology's workflow is depicted in Figure 1, illustrating the steps involved. Firstly, unstructured tweets related to ChatGPT are collected from Twitter using the Twitter Tweepy API. These tweets undergo several preprocessing steps to ensure cleanliness and remove noise. Lexicon-based techniques are then utilized to assign labels of positive, negative, or neutral to the tweets. Feature extraction is performed using the Bag of Words (BoW) technique on the labeled dataset. The data is subsequently split into an 80/20 ratio for training and testing purposes. Following model training, evaluation metrics such as accuracy, precision, recall, and the F1 score are employed to analyze the model's performance. Each component of the proposed methodology for sentiment classification is discussed in greater detail in the subsequent sections.

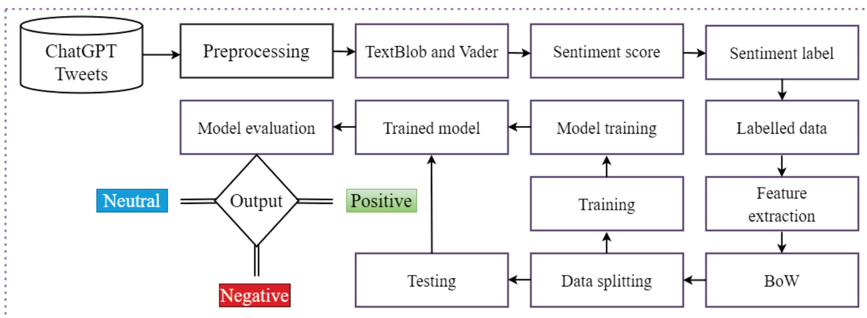


Figure 1. The workflow diagram of the proposed approach for sentiment classification.

3.1. Dataset Description and Preprocessing

In this study, the ChatGPT tweets dataset is utilized, which is scraped from Twitter using the Tweepy API Python library. A total of 21,515 raw tweets are collected for this purpose. The dataset contains the date, user name, user friends, user location, and text features. The dataset is unstructured and requires several preprocessing steps to make it appropriate for machine learning models.

Text preprocessing is very important in NLP tasks for a sentiment analysis. The dataset used in this paper is unstructured, unorganized, and contains unnecessary and redundant information. The machine learning or deep learning models do not perform well on these types of datasets, which increases the computational cost [42]. Different preprocessing techniques are utilized to remove unnecessary, meaningless information from the tweets. Preprocessing is a crucial step in data analysis that involves transforming unstructured data into a meaningful and comprehensible format [43]. The purpose of preprocessing is to enhance the quality of the dataset while preserving its original content, enabling the model to identify significant patterns that can be utilized to extract valuable and efficient information from the preprocessed data. There are many steps in preprocessing to convert unstructured text into structured data. These techniques are used to remove the least important information from the data and make it easier for the machine to train in less time.

The dataset consists of 20,801 tweets, 8095 of which are positive, 2727 of which are negative, and 9979 of which are neutral. Following the split, 6476 positive tweets were used for training and 1619 for testing. There were 1281 negative tweets utilized for training and 546 for testing. For neutral tweets, 7983 were training and 1996 were testing. The hashtags #chatgpt, #ChatGPT, #OpenAI, #ChatGPT-3, #Chatbots, #Powerful OpenAI, etc., were used to collect all of the tweets in English. Table 2 shows the dataset statistics.

Table 2. Dataset statistics after splitting.

Tweets	Training	Testing	Total
Positive	6476	1619	8095
Negative	1281	546	2727
Neutral	7983	1996	9979
Total	16,640	4161	20,801

The most important step in natural language processing (NLP) is the pre-processing stage. It enables us to remove any unnecessary information from our data so that we can proceed to the following processing stage. The Natural Language Toolkit (NLTK), which provides modules, is an open-source Python toolkit that can be used to perform operations such as tokenization, stemming, classification, etc. The first step in preprocessing is to convert all textual data to lowercase. Conversion is an essential step in sentiment classification, as the machine considers "ChatGPT" and "chatgpt" as individual words. The dataset contains text in upper, lower, and sentence case, which the model takes separately, which affects the classification performance as well and makes the data more complex if we do not convert it all into lowercase. The second step is to remove numbers from the text because they do not provide meaningful information and are useless in the decision-making process. The removal of numerical data enhances the quality of the data [44]. The third step is to remove punctuation such as [?,@,#,/,&,%] to increase the quality of the dataset and the performance of the models. The fourth step is to remove HTML and URL tags that also provide no important information. The URLs in the text data are meaningless because they expand the dataset and require extra computation. It has no impact on the machine learning performance. The fifth step is to remove stopwords like 'an', 'the', 'are', 'was', 'has', 'they', etc., from the tweets during preprocessing. The model's accuracy improves, and the training process is faster, with only relevant information [44]. Additionally, the removal of stopwords allows for a more thorough analysis, which is advantageous for a limited dataset [45]. The last step is to perform stemming and lemmatization. The effectiveness of machine learning is slightly influenced by the stemming and lemmatization steps. After performing all important preprocessing steps, the sample tweets are presented in Table 3.

Table 3. Sample Tweets before preprocessing and after preprocessing.

Unstructured Tweets	Structured Tweets (Preprocessed)
I asked #chatgpt to write a story instalment with Tim giving the octopus a name. Originality wasn't its strongpoint €¦ https://t.co/rbB5prcJ2r (accessed on 2 April 2023).	asked chatgpt write story instalment tim giving octopus name originality strongpoint
ChatGPT is taking the web by storm; If you're unable to try it on their site, feel free to test it out through us! €¦ https://t.co/jfmOQmjSHo (accessed on 2 April 2023).	chatgpt taking web storm unable try site feel free test
People weaponizing a powerful AI tool like ChatGPT days into launch has to be the most predictable internet	people weaponizing powerful tool like chatgpt days launch predictable internet

3.2. Lexicon Based Techniques

TextBlob [46] and VADER [47] are the two most important lexicon-based techniques used in this study to label the dataset. TextBlob provides the subjectivity and polarity scores, where 1 represents the positive response and -1 represents the negative response in polarity. The subjectivity score is represented by [0, 1]. The VADER technique calculates the sentiment score by adding the intensity of each word in the preprocessed text.

3.3. Feature Engineering

The labeled dataset is divided into training and testing subsets. The training data has been used to fit the model, while the test data is used by the model for predictions on unseen data, which are then compared to determine the model's efficacy.

Important features from the cleaned tweets are extracted using the BoW approach. The BoW approach extracts valuable features from the data to enhance the performance of machine learning models. Features are very crucial and have a great impact on sentiment classification. This approach reduces processing time and effort. The BoW approach creates a bag of words of text data and converts it into a numeric format. The models learn and understand complex patterns and sequences from the numeric format [48].

3.4. Machine and Deep Learning Models

This subsection provides details about the machine and deep learning models. The applications of machine and deep learning span across various domains, such as disease diagnosis [49], education [50], computer/machine vision [51,52], text classification [53], and many more. In this study, we utilize these techniques for text classification. The objective of text classification is to automatically classify texts into predetermined categories. Deep learning and machine learning are both forms of artificial intelligence [54]. Classification of text using machine learning entails the transformation of input data into a numeric form. Then, manually extracting features from the data using a bag of words, term frequency, inverse document frequency, word2vec, etc., to extract crucial features. Frequently employed models of machine learning, such as random forests, support vector machines, extra tree classifiers, etc., cannot learn complex patterns and are not employed for large datasets. When we apply these models to large datasets, they perform poorly and require excessive training time, particularly for handcrafted features. If the researchers applied machine learning to complex problems, they would require manual feature engineering to retain only the essential information, which is time-consuming and requires expertise in the same fields to improve classification results.

Deep learning [55], on the other hand, has a method for automatically extracting features. Large and complex patterns are automatically learned from the data using DL models like CNN, LSTM, GRU, etc., minimizing the need for manual feature extraction. When there is a lack of data, the model could get overfitted and perform poorly. These models address the issue of vanishing gradients. In terms of computing, gated recurrent units (GRU) outperform LSTM, reduce the chances of overfitting, and are better suited for small datasets. Additionally, GRU has a straightforward structure with fewer parameters. The authors only used models that are quick and effective in terms of computing.

We developed transform-based models that use self-attention mechanisms since they are the most effective after machine and deep learning. They have the capacity to comprehend the relationship between consecutive elements set far apart from one another. They achieve an outclass performance. They give each component of the sequence the same amount of attention. The large data can be processed and trained by transformers in a shorter period of time. They are capable of processing almost any form of sequenced information. The hyperparameters and their fine-tuned values are represented in Table 4. These parameters are obtained using the GridSearchCV method which performs an exhaustive search for the given parameters to evaluate a model's performance and provides the best set of parameters for obtaining optimal results.

Table 4. Hyperparameters and their tuned values for experiments.

Model	Parameters Tuning
RF	n_estimators = 100, random_state = 50, max_depth = 150
GBM	n_estimators = 100, random_state = 100, max_depth = 300
LR	random_state = 150, solver = 'newton-cg', multi_class = 'multinomial', C = 2.0

Table 4. *Cont.*

Model	Parameters Tuning
SVM	kernel = 'linear', C = 1.0, random_state = 200
KNN	n_neighbors = 3
DT	random_state = 100, max_depth = 150
ETC	n_estimators = 100, random_state = 150, max_depth = 300
SGD	loss = "hinge", penalty = "l1", max_iter = 6
CNN	616,003 trainable parameters
RNN	633,539 trainable parameters
LSTM	655,235 trainable parameters
BILSTM	726,787 trainable parameters
GRU	692,547 trainable parameters

- Logistic Regression: LR [56] is a simple machine learning model used in this study for sentiment classification. LR provides accurate results with preprocessed and highly relatable features. It is simple to implement and utilizes low computational resources. This model may not perform well on large datasets, cause overfitting, and does not learn complex patterns due to its simplicity.
- Random Forest: The RF is an ensemble supervised machine learning model used for classification, regression, and other NLP tasks [57]. The RF ensembles multiple decision trees to form a forest. A large amount of textual data and the ensemble of trees make the model more complex which takes a higher amount of time to train. The RF is powerful and has attained high accuracy for the sentiment analysis.
- Decision Tree: A DT is a supervised non-parametric learning model for classification and regression. The DT predicts a target variable using learned features to classify objects. A decision tree requires less data cleaning than other machine learning methods. In other words, decision trees do not require normalization during the early stages of machine learning tasks. They can handle both categorical and numerical information [58].
- K Nearest Neighbour: The KNN model requires no previous knowledge and does not learn from training data. It is also called the lazy learner. It does not perform well when data is not well normalized and structured. The performance can be manipulated with the distance metrics and K value [59].
- Support Vector Machine: The SVM is mostly used for classification tasks. It performs well where the number of dimensional spaces is greater than the number of samples [17]. The SVM does not perform well on large datasets because the training time increases. It is more robust and handles imbalanced datasets efficiently. The SVM can be used with 'poly', 'linear', and 'rbf' kernels.
- Extra Tree Classifier: The ETC is used for classification and regression [60]. Extra trees do not use the bootstrapping approach and train faster. The ETC requires fewer parameters for tuning compared to RF. Also, with extra trees, the chances of overfitting are less.
- Gradient Boosting Machine (GBM) and Stochastic Gradient Descent (SGD): The GBM [61] and SGD are supervised learning models for classification. To enhance the performance, the GBM combines multiple decision trees, and the SGD optimizes the gradient descent. The GBM is more complex and handles imbalanced data better than the SGD.
- Convolutional Neural Networks (CNN): The CNN [62] is a deep neural network model that is used for image classification, sentiment classification, object detection, and many other tasks. For sentiment classification, it first converts textual data into a numeric format, then make a matrix of word embedding layers. These embedding

layers are then passed into convolutional, max-pooling, and dense layers, and the final output is passed through a dense softmax layer for classification.
- Recurrent Neural Network (RNN): The RNN [63] is a widely used model for text classification, speech recognition, and NLP tasks. The RNN can handle sequential data with complex long-term dependencies. This model is expensive to train and has the vanishing gradient issue for text classification.
- Long Shor-Term Memory: The LSTM [64] model was released to handle long-term dependencies, the gradient vanishing issue, and the complex training time. When compared to RNN, this model is much faster and uses less memory. It has three gates, including input, output, and forget, which are used to manage the data flow.
- Bidirectional LSTM: The BiLSTM is a deep learning model which is used for several tasks, including text classification as well [65]. The model provides better results for understanding the text in past and future contexts than the LSTM. It can learn information from both directions.
- Gated Recurrent Unit (GRU): The GRU solves the problem of vanishing gradient, faced by RNN [66]. It is fast and performs well on small datasets. The model has two gates: an update gate and a reset gate.

3.5. Transformer Based Architecture

BERT is a transformer-based model presented by Devlin et al. [67] in 2018. The BERT model uses an attention mechanism that takes actual input from the text. The BERT has two parts: an encoder and a decoder. The encoder gets the input as text and produces output such as predictions. The BERT model is particularly well suited for NLP tasks, including a sentiment analysis and questioning-and-answering, because it is trained on a large amount of textual data. The traditional models only use word context-of-word in just one direction, normally from left to right. The BERT model considers the context of words in NLP in both directions. In contrast to previous deep learning models, this model has a clear understanding of word meanings. The BERT model is trained on a large amount of data to obtain accurate results and to learn complex patterns and structures [68].

The BERT with fine-tuned hyperparameters works well for a variety of NLP tasks. Santiago Gonzalez and Eduardo C. Garrido-Merchan [69] published a study that compared the BERT architecture to traditional machine learning models for sentiment classification. The traditional models were trained using features extracted from TF-IDF. The performances demonstrate that the BERT transformer-based model outperforms the traditional models. To solve NLP-related problems, the BERT model has also been used for low-resource languages. BERT was used to pre-train text data and fine-tuned low-resource languages by Jan Christian Blaise Cruz and Charibeth Cheng [70]. Because this model takes input words with multiple word sequences at once, the results for that language were improved.

Figure 2 shows the proposed architecture of BERT for sentiment classification. The BERT uses a large, pre-trained vocabulary to generate input ids that are numeric values of the input text. First of all, a sequence of tokens is created from whole input text tokens, and unique ids are assigned to the tokens. Basically, input ids are numerical representations of input text. In BERT, the input mask works like an attention mechanism, which clearly differentiates between input text tokens and padding. The input mask identifies which tokens in the input sequence are evaluated by the model and which ones are not evaluated. Segment ids indicate extra tokens to differentiate different sentences. After that, it is concatenated with the BERT Keras layer. This study uses three dense layers in BERT with 128, 64, and 32 units and two 20% dropout layers. The final dense layer is used for classification with the softmax activation function.

XLNet was released by Ashish Vaswani in 2019, and its architecture is similar to BERT. The BERT is an auto-encoder, and the XLNet is an autoregressor model [71]. The BERT model cannot correctly model the dependencies between tokens in a sentence. XLNet overcomes this problem by adopting permutation-based training objectives as compared to mask-based objectives. The permutation-based objective permits XLNet to represent the dependencies with all tokens in a paragraph.

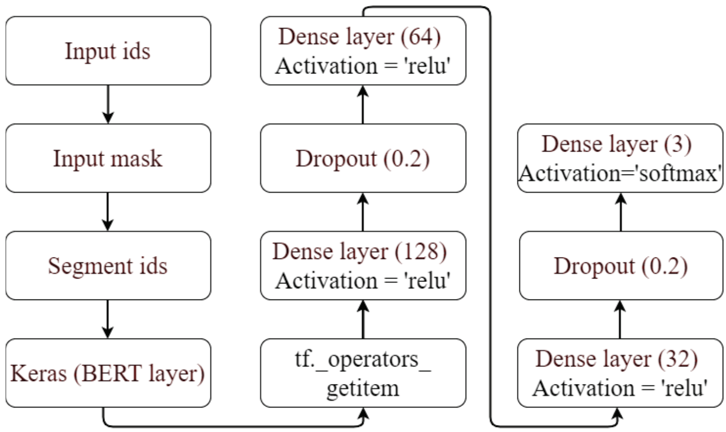


Figure 2. The architecture for the proposed sentiment classification.

Robustly optimized BERT pretraining (RoBERTa) [72] is a transformer-based model used for various NLP tasks. It was developed in 2019. RoBERTa is a modification of the BERT model to overcome the limitations of the BERT model. RoBERTa is trained on 160 billion words, whereas BERT is trained on only 3.3 billion words. RoBERTa is trained on large data sets, is fast to train, and may use large batch sizes. RoBERTa uses a dynamic masking approach, and BERT uses a static approach.

3.6. Performance Metrics

The performance of the machine, deep, and transformer-based models are also measured using evaluation metrics including accuracy, precision, recall, and the F1 score [73]. Accuracy is calculated using

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

where TP stands for true positive, TN for true negative, FP for false negative, and FN for false negative.

Precision is another performance metric used to measure performance. Precision is defined as the ratio of actual positives to the total number of positive predictions.

$$Precision = \frac{TP}{(TP + FP)} \tag{2}$$

The recall is also used to measure the performance of models. The recall is calculated by dividing the true positives by the sum of true positives and false negatives.

$$Recall = \frac{TP}{(TP + FN)} \tag{3}$$

The F1 score is a better metric than other metrics in a situation where classes are imbalanced because it considers both precision and recall and provides a better understanding of the model's performance.

$$F1 - score = 2 * \frac{(Recall * precision)}{(Recall + precision)} \tag{4}$$

4. Results and Discussion

This section presents the details regarding experiments on the ChatGPT Twitter dataset using machine learning, deep learning, and transformer-based models. The Colab Notebook in Python with Tensorflow, Keras, and Sklearn libraries is used to evaluate the research experiments. Different measures including accuracy, precision, recall, and the F1 score are used to assess the performance of various models. For deep and transformer-based models, a graphics processing unit (GPU) and 16 GB of RAM are used to speed up the training process. Experimental results are presented in the subsequent sections.

4.1. Results of Machine Learning Models

Table 5 shows the results of eight machine learning models utilizing Textblob and VADER lexicon-based techniques on ChatGPT Twitter data. With an accuracy of 94.23%, SVM outperforms while SGD achieves an accuracy of 92.74%. A 91% accuracy is attained by ETC, GBM, and LR while the lazy learner KNN obtains only a 58.03% accuracy. The SVM model has 88% accuracy, 89% recall, and an 83% F1 score for the negative class, whereas the GBM model has 91% precision, 63% recall, and a 74% F1 score. Utilizing BoW features, the neutral tweets get the highest recall scores.

Table 5. Results of machine learning models using VADER and TextBlob techniques.

			Vader			TextBlob			
Model	**Accuracy**	**Class**	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
SGD	89.13	Positive	93	92	93	92.76	94	93	93
		Negative	84	69	76		89	75	81
		Neutral	87	94	90		93	95	97
RF	82.40	Positive	92	83	88	86.99	94	85	89
		Negative	92	43	58		94	47	63
		Neutral	73	98	84		82	99	90
DT	82.26	Positive	93	82	87	88.29	94	85	90
		Negative	82	47	60		89	56	69
		Neutral	94	97	84		84	99	91
ETC	87.11	Positive	93	89	91	91.80	94	91	93
		Negative	92	56	69		90	66	76
		Neutral	81	98	89		90	99	94
KNN	54.38	Positive	95	47	22	58.03	95	20	34
		Negative	83	20	33		80	18	30
		Neutral	47	99	64		54	99	70
SVM	90.72	Positive	95	92	94	94.23	96	94	95
		Negative	85	73	79		88	89	83
		Neutral	89	96	92		94	99	96
GBM	89.56	Positive	93	92	92	92.28	94	94	94
		Negative	92	65	76		91	63	74
		Neutral	85	97	91		91	99	95
LR	88.44	Positive	93	91	92	91.56	95	91	93
		Negative	89	63	74		92	66	77
		Neutral	84	96	90		89	99	96

Table 5 also shows the results of various models using the VADER technique. Using a VADER lexicon-based technique, SVM performs best with an accuracy of 90.72%. The models SGD and GBM both achieved an 89% accuracy score. The model that performs worse, in this case, is KNN, with a 54.38% accuracy. This model also performs poorly on the TextBlob technique. The only model in machine learning that performs with the highest accuracy is SVM with the linear kernel. The accuracy score of various machine learning models using TextBlob and Vader are compared in Figure 3.

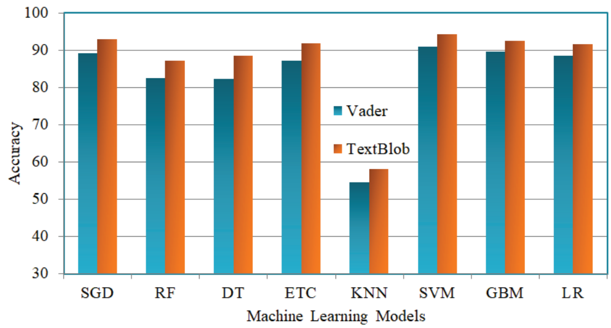


Figure 3. Performance of models using the TextBlob and VADER techniques. The X-axis presents the machine learning models that we utilized in this study, and the Y-axis presents the accuracy score.

4.2. Performance of Deep Learning Models

Deep learning models are also used to perform a sentiment classification and analysis. Results using the TextBlob technique are shown in Table 6. The experimental results on the ChatGPT preprocessed Twitter dataset show that the BiLSTM deep model achieves a 93.12% accuracy score, which is the highest as compared to CNN, RNN, LSTM, and GRU. The LSTM model also performs well, with an accuracy score of 92.95%. The other two deep models, GRU and RNN, reached an accuracy higher than 90%. The performance of the CNN model is not good. The CNN model achieved a 20% lower accuracy than other models.

Table 6. Results of deep learning models using the TextBlob technique.

Model	Accuracy	Class	Precision	Recall	F1 Score
CNN	70.88	Positive	73	66	69
		Negative	56	48	52
		Neutral	71	81	77
RNN	90.35	Positive	91	92	92
		Negative	80	71	75
		Neutral	92	94	93
LSTM	92.95	Positive	93	94	93
		Negative	83	82	82
		Neutral	96	96	96
BiLSTM	93.12	Positive	91	96	93
		Negative	86	81	83
		Neutral	97	94	12
GRU	92.33	Positive	92	94	93
		Negative	82	81	82
		Neutral	95	94	95

Table 7 shows the results of deep learning using the VADER technique. The performance of five deep learning models is evaluated using accuracy, precision, recall, and the F1 score. The LSTM model achieves the highest accuracy of 87.33%, while the CNN model achieves the lowest accuracy of 68.77%. The GRU and BiLSTM models achieve a 93% recall score for the positive sentiment class. The lowest recall of 44% is obtained by CNN. The CNN model shows poor performance both with the TextBlob and VADER techniques.

Table 7. Results of deep learning models using the VADER technique.

Model	Accuracy	Class	Precision	Recall	F1 Score
CNN	68.77	Positive	77	68	72
		Negative	56	44	50
		Neutral	65	80	72
RNN	82.40	Positive	809	88	89
		Negative	62	66	64
		Neutral	83	82	83
LSTM	87.33	Positive	89	92	90
		Negative	74	75	75
		Neutral	91	87	89
BiLSTM	86.95	Positive	88	93	90
		Negative	76	74	75
		Neutral	91	86	88
GRU	86.48	Positive	88	93	90
		Negative	74	70	72
		Neutral	90	86	88

4.3. Results of Transformer-Based Models

Currently, transformer-based models are very effective and perform well on complex natural language understanding (CNLU) tasks in sentiment classification. Machine learning and deep learning models are also used for sentiment analyses, but machine learning performs well on small datasets and deep learning models require large datasets to achieve a high accuracy.

Table 8 shows the results of transformer-based models using the TextBlob technique. The transformer-based robustly optimized BERT model achieves the lowest accuracy of 93.68% while 96% of recall scores are achieved for positive and neutral classes by RoBERTa. The XLNet model achieves an 85.96% accuracy which is low as compared to the RoBERTa and proposed BERT model. In comparison to any other machine or deep learning model, the proposed approach achieves the highest accuracy of 96.49%. The precision, F1 score, and recall of the proposed approach are also higher than those of others.

The results of transformer-based models are also evaluated using the VADER technique. The proposed approach also performs well using the VADER technique with the highest accuracy, as shown in Table 9. The proposed approach understands full contextual content, gives importance to relevant parts of textual data, and makes efficient predictions. The RoBERTa and XLNet transformer-based models achieve 59.59% and 68.51% accuracy scores, respectively. Using the VADER technique, the proposed method achieved a 93.37% accuracy which is higher than all of the other transformer-based models when used with VADER. The other performance metrics, such as precision, recall, and the F1 score, achieved by the proposed model are also better than the other models.

Table 8. Performance of transformer-based models using the TextBlob technique.

Model	Accuracy	Class	Precision	Recall	F1 Score
RoBERTa	93.68	Positive	95	96	93
		Negative	84	85	85
		Neutral	95	96	96
XLNet	85.96	Positive	93	83	87
		Negative	66	77	71
		Neutral	86	91	89
Proposed BERT	96.49	Positive	96	98	97
		Negative	92	90	91
		Neutral	98	97	98

Table 9. Performance of transformer-based models using the VADER technique.

Model	Accuracy	Class	Precision	Recall	F1 Score
RoBERTa	86.68	Positive	75	79	77
		Negative	88	88	88
		Neutral	90	88	89
XLNet	68.51	Positive	66	72	69
		Negative	25	45	32
		Neutral	85	70	76
Proposed BERT	93.37	Positive	97	92	95
		Negative	87	89	88
		Neutral	93	96	94

Table 10 shows the correct and wrong predictions by deep learning and BERT models using the TextBlob. Results are given only for the TextBlob technique, as the models perform well using the TextBlob technique. Out of 4000 predictions, the RNN made 3614 correct predictions and 386 wrong predictions. The LSTM made 3718 correct predictions while 282 predictions are wrong. The BiLSTM has 3725 correct and 275 wrong predictions. The GRU shows 3693 correct predictions, compared to 307 wrong ones. Out of 4160 predictions, the XLNet made 3576 correct and 584 wrong predictions. On the other hand, the RoBERTa made 3897 correct and 263 wrong predictions. The BERT made 4015 correct predictions whereas 146 predictions are wrong. The results demonstrate that the BERT model performed better than the machine learning and deep learning models. Only with 2835 correct and 1165 wrong predictions, the only CNN model performed poorly.

Table 10. Correct and wrong predictions by various models using the TextBlob technique.

Model	Correct-Predictions	Wrong-Predictions	Total-Predictions
CNN	2835	1165	4000
RNN	3614	386	4000
LSTM	3718	282	4000
BiLSTM	3725	275	4000
GRU	3693	307	4000

Table 10. *Cont.*

Model	Correct-Predictions	Wrong-Predictions	Total-Predictions
XLNet	3576	584	4160
RoBERTa	3897	263	4160
Proposed BERT	**4015**	**146**	**4161**

4.4. Results of K-Fold Cross-Validation

K-fold cross-validation is the most effective method for assessing the model's robustness and validating its performance. Table 11 shows the results of Transformer-based models with K-fold cross-validation. Experiments show that the proposed BERT model is highly efficient in the sentiment analysis for ChatGPT tweets with an average accuracy of 96.49% using the TextBlob approach with a ±0.01 standard deviation. The proposed model also works well using the VADER approach with a ±0.01 standard deviation. The RoBERTa on the K-fold achieves a 91% accuracy with a ±0.06 standard deviation, while XLNet achieves a 68% accuracy with a ±0.18 standard deviation.

Table 11. K-fold cross-Validation results using TextBlob and VADER approaches.

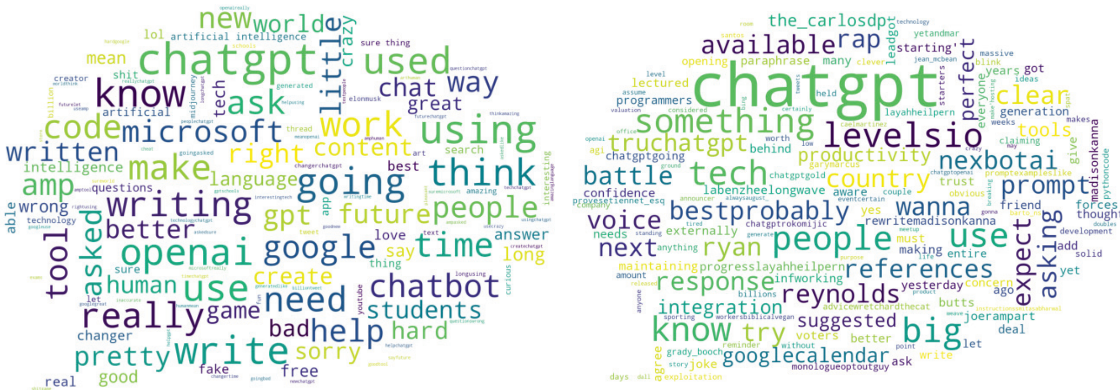
	Model	Accuracy	Standard Devation
TextBlob	RoBERTa	0.91	±0.06
	XLNet	0.68	±0.18
	Proposed BERT	0.95	±0.01
VADER	RoBERTa	0.85	±0.02
	XLNet	0.66	±0.02
	Proposed BERT	0.93	±0.01

4.5. Topic Modeling Using BERTopic and LDA Method

Topic modeling is an important approach in NLP, as it automatically extracts the most significant topics from textual data. There is a vast amount of unstructured data available on social media, and traditional approaches are incapable of handling such data. Topic modeling can handle and extract meaningful information from unstructured text data efficiently. In Python, topic modeling is applied to the preprocessed data with important libraries to improve the results. Topic modeling is also used to discover related topics from frequently discussed tweets' datasets.

In various NLP tasks, transformer-based models have produced very promising results. BERTopic is a new topic modeling method that employs the BERT transformer model to extract key trends or keywords from large datasets. BERTopic gathers semantic information that better represents topics. BERT extracts contextual and complicated problems more accurately and efficiently. Furthermore, BERTopic extracts relevant recent trends from Twitter. When compared to LDA modeling, LDA is incapable of extracting nuanced and complicated contextual issues from tweets. In comparison to BERTopic, LDA employs outdated techniques and is unable to extract current patterns. However, BERTopic is a better choice for topic modeling for large datasets.

LDA [74] is an approach used for topic modeling in NLP problems. It is easy to use, efficient, and faster than other approaches for topic modeling. LDA modeling is performed on textual data, and then a document term matrix is created that shows the frequency of each term in a document. The BoW features are utilized to understand the most crucial terms in a document. After that, the most prominent keywords are extracted from ChatGPT tweets using BERTopic, and the LDA are shown in Figure 4.



(**a**) (**b**)

Figure 4. Comparison of LDA-based and BERT-based topic modeling techniques through word clouds: (**a**) Visualization of tweets using LDA topic modeling, and (**b**) Visualization of tweets using BERTopic modeling.

Figure 5 depicts the most prominent topics extracted by BERTopic. First, we load the BERT model and associated tokenizers. The tweet data are then preprocessed to extract the embeddings for the BERT model. Then, for dimension reduction or clustering, we used k-means clustering and the principal component analysis (PCA). The BERT model was used to extract the most prominent topics, which were then displayed in a scatter plot.

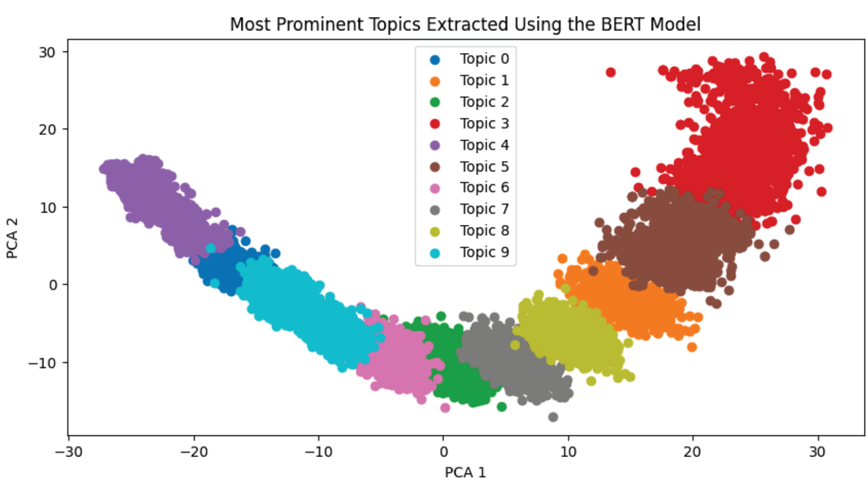


Figure 5. Most Prominent Topics extracted from ChatGPT Tweets using BERTopic.

Figure 6 expresses the content or words of the top ten positive and negative topics and their frequency. The word ChatGPT is mostly discussed in the Twitter tweets in a positive context, and negative words like fake and wrong are discussed but less. The words good, best, and love have the lowest frequency in the top ten topics.

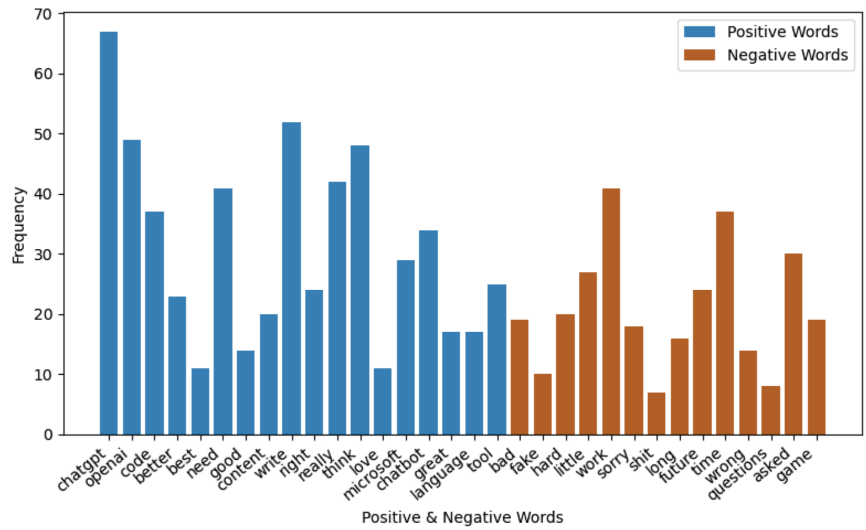


Figure 6. Words extracted from top ten topics with their frequency using the LDA model.

Figures 7 and 8 show the most discussed positive and negative topics, extracted from the ChatGPT tweets using the LDA approach with BoW features. These Figures illustrate positive and negative words in the context of various topics. The users shared their opinions regarding ChatGPT on social media platforms like Twitter. The user posted positive or negative opinions about ChatGPT. The authors extract these tweets from Twitter and perform an analysis to analyze how people feel about or discuss this technology. The authors used LDA-based Topic modeling to extract the most prominent keywords from the tweets. These keywords provide important themes to understand the main context and identify the emotions; they also capture semantic meanings. In the tweets, the word "good" indicates a cheerful mood. It represents anything beneficial or pleasurable. The majority of the time, "good" refers to a positive quality. It is classified as positive sentiment in the sentiment analysis because this inference is generally understood to be positive. It is important to clarify that these words are not inherently positive or negative; rather, their categorization depends on the positive or negative topics they are associated with. For instance, words like "better", "best", and "good" are included in positive topics and are used in a positive context within GPT. Better indicates an advance over a previous state or condition, indicating a positive development. ChatGPT is frequently spoken of favorably due to its features and potential applications in a variety of industries. The development of AI language models like ChatGPT is demonstrated by their ability to comprehend and generate text responses that resemble human responses. ChatGPT allows users to partake in entertaining and engaging conversations. On the other hand, ChatGPT in the negative context indicates that it sometimes produces irrelevant or incorrect results, raises privacy concerns, and an excessive dependence on ChatGPT may impair the ability to think critically and solve technical problems. Social media users frequently use words like "bad", "wrong", "little", and "hot" in a negative sense, aligning with negative topics. Sentiment analysis models can be refined and improved over time based on feedback and real-world data to better capture the nuances of sentiments expressed in different contexts. The performance can be analyzed by policymakers based on these prominent keywords, and they can modify their product according to this.

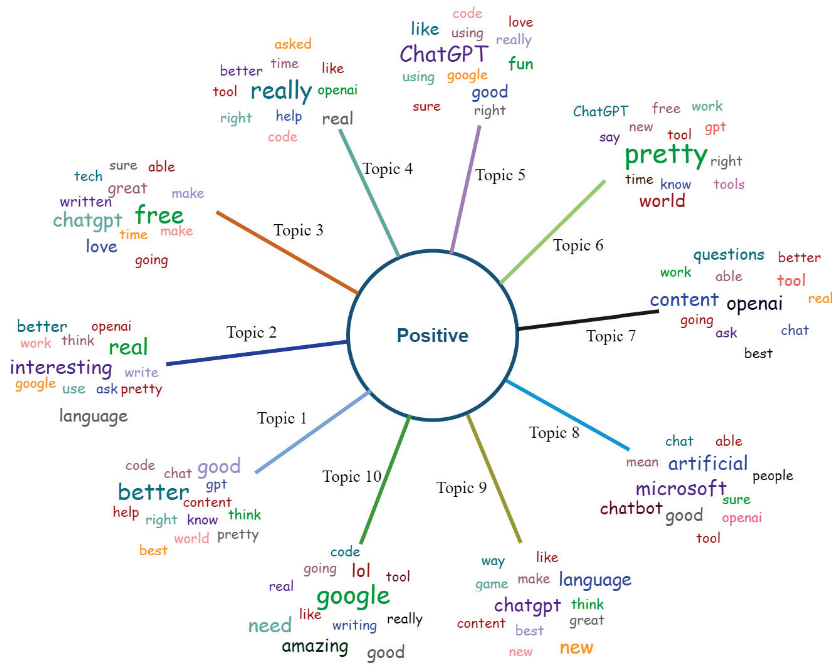


Figure 7. Visualization of highly discussed positive topics.

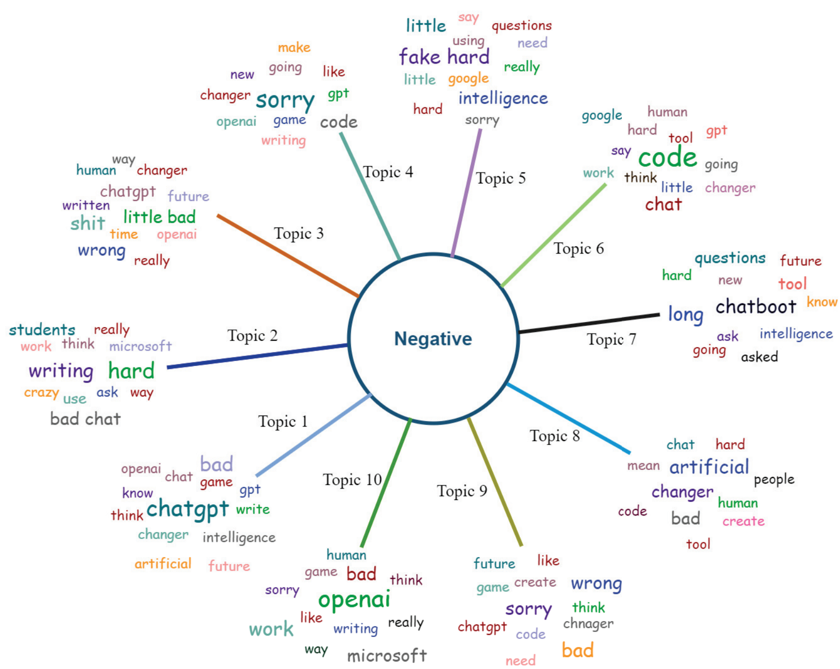


Figure 8. Visualization of highly discussed negative topics.

4.6. Comparison of Proposed Approach with Machine Learning Models Using Statistical Test

The comparison between the machine learning and the proposed Transformer-based BERT model is presented in Table 12. Machine learning models are fine-tuned to optimize the results. The authors evaluated the proposed approach using the TexBlob and Vader technique. In all scenarios, the proposed approach rejects the H_o and accepts the H_a, which means that the proposed approach is statistically significant in comparison with other approaches.

Table 12. Statistical test comparison with the proposed model.

	TextBlob			Vader		
Scenario	**Statistics**	***p*-Value**	H_o	**Statistics**	***p*-Value**	H_o
Proposed BERT Vs. SGD	−7.999	0.015	Rejected	−31.128	7.284	Rejected
Proposed BERT Vs. RF	−39.167	3.661	Rejected	−3.695	0.343	Rejected
Proposed BERT Vs. DT	0.633	0.571	Rejected	−34.097	5.545	Rejected
Proposed BERT Vs. ETC	−63.516	8.598	Rejected	−3.43	0.041	Rejected
Proposed BERT Vs. KNN	−8.225	0.003	Rejected	−6.140	0.008	Rejected
Proposed BERT Vs. SVM	−9.792	0.002	Rejected	−3.257	0.047	Rejected
Proposed BERT Vs. GBM	−9.845	0.002	Rejected	−3.313	0.045	Rejected
Proposed BERT Vs. LR	−17.691	0.000	Rejected	−3.368	0.043	Rejected

4.7. Performance Comparison with State-of-the-Art Studies

For evaluating the robustness and efficiency of the proposed approach, its performance is compared with the state-of-the-art existing studies. Table 13 shows the results of state-of-the-art studies. The study [26] used machine learning models for a sentiment analysis and LR performed well with 83% accuracy. Khalid et al. [27] performed an analysis on Twitter data using an ensemble of machine learning models and achieved 93% accuracy with the BBSVM model. Another study [75] carried out a sentiment analysis on Twitter data using machine learning models. Machine learning models do not perform well due to small datasets and show poor accuracy. As a result, the authors used transformer-based models for the sentiment analysis. For example, Bello et al. [33] used the BERT model on tweets. The proposed BERT model utilizes contextual information to produce a vector representation. When integrated with neural network classifiers such as CNN, RNN, or BiLSTM for prediction, it attains an accuracy rate of 93% and an F measure of 95%. The BiLSTM model exhibits some shortcomings, one of which is its inability to effectively capture the underlying contextual nuances of individual words. Other authors, such as [34,35], used the BERT models for the sentiment analysis with various datasets. They conducted an evaluation of the efficacy of Google's BERT method in comparison to other machine learning methods. Moreover, this study investigates the Bert architecture, which received pre-training on two natural language processing tasks, namely Masked language Modeling and sentence Prediction. The Random Forest (RF) is commonly employed as a benchmark for evaluating the performance of the BERT language model due to its superior performance among various machine learning methods. Previous methodologies are mostly on natural language techniques for the classification and analysis of tweets, yielding insufficient results. The aforementioned prior research indicates the need for an approach that can effectively analyze tweets based on their precise classification. The performance analysis indicates that the proposed BERT model shows efficient results with a 96.49% accuracy and outperforms existing studies.

Table 13. Comparison of proposed approach with state-of-the-art existing studies.

Authors	Model	Dataset	Accuracy	Publication
Rustam et al. [26]	Logistic Regression	App reviews	83%	2020
Khalid et al. [27]	GBSVM	Twitter Data	93%	2020
Wadhwa et al. [75]	Logistic Regression	Twitter Data	86.51%	2021
Bello et al. [33]	BERT	Twitter Data	93%	2022
Catelli et al. [34]	BERT	E-commerce reviews	75%	2021
Patel et al. [35]	BERT	Reviews	83	2022
Proposed	**BERT**	**Twitter Data**	**96.49%**	**2023**

4.8. Validation of Proposed Approach on Additional Dataset

The validation of the proposed approach is carried out using an additional public benchmark dataset. For this purpose, experiments are performed on the well-known SemEval2013 dataset [76]. The proposed TextBlob+BERT approach is applied to the SemEvel2013 dataset, where TextBlob generates new labels for the dataset, and the proposed BERT model performs classification. Moreover, experiments are also done using the original labels of SemEvel2013. Experimental results are presented in Table 14 which indicate the superior performance of the proposed approach. It can be observed that the proposed approach performs significantly well on the SemEvel2013 dataset with a 0.97 accuracy score when labels are assigned using the TextBlob and BERT is used for classification. For the second set of experiments which involves using the original labels of the SemEvel2013 dataset, LR shows the best performance with a 0.65 accuracy score.

Table 14. Experimental results on the SemEvel2013 dataset.

Approach	Accuracy	Class	Precision	Recall	F1 Score
TextBlob + BERT	0.97	Negative	0.97	0.91	0.94
		Neutral	0.98	0.99	0.98
		Positive	0.96	0.98	0.97
		macro avg	0.97	0.96	0.97
		weighted avg	0.97	0.97	0.97
Original + LR	0.65	Negative	0.65	0.47	0.54
		Neutral	0.63	0.72	0.67
		Positive	0.69	0.65	0.67
		macro avg	0.65	0.62	0.63
		weighted avg	0.65	0.65	0.65

4.9. Statistical Significance Test

This study performs a statistical significance *t*-Test to show the significance of the proposed approach. For the statistical test, several scenarios are considered, as mentioned in Table 15. The *t*-test shows the significance of one approach on the other by accepting or rejecting the null hypothesis (H_o). In this study, we consider two cases [77]:

- Null Hypothesis $(H_o) => \mu 1 = \mu 2$: The population means of the proposed approach's results is equal to the compared approach's results. (No statistical significance)
- Alternative Hypothesis $(H_a) => \mu 1 \neq \mu 2$: The population means of the proposed approach's results is not equal to the compared approach's results. (Proposal approach is statistically significant)

Table 15. Statistical significance *t*-test.

Scenario	Statistic	*p*-Value	H_o
Proposed BERT Vs. RoBERTa	3.304	3.304	Rejected
Proposed BERT Vs. XLNet	7.292	0.0003	Rejected
Proposed BERT Vs. GRU	4.481	0.004	Rejected
Proposed BERT Vs. BiLSTM	2.621	0.003	Rejected
Proposed BERT Vs. LSTM	2.510	0.045	Rejected
Proposed BERT Vs. RNN	6.474	0.000	Rejected
Proposed BERT Vs. CNN	8.980	0.000	Rejected

The *t*-test can be interpreted as if the output *p*-value is greater than the alpha value (0.05), it indicates that the H_o is accepted and there is no statistical significance. Moreover, if the *p*-value is less than the alpha value, it indicates that H_o is rejected and H_a is accepted which means that there is statistical significance between the compared results. We perform a *t*-test on results using Textblob and compare all models' performances. In all scenarios, the proposed approach rejects the H_o and accepted the H_a, which means that the proposed approach is statistically significant in comparison with other approaches.

4.10. Discussion

In this study, we observed that the majority of sentiment towards chatGPT was positive, indicating a generally favorable perception of the tool. This aligns with the notion that chatGPT has gained significant attention and popularity on various online platforms. The positive sentiment towards chatGPT can be attributed to its advanced language generation capabilities and its ability to engage in human-like conversations. Figure 9 shows the sentiment ratio for chatGPT.

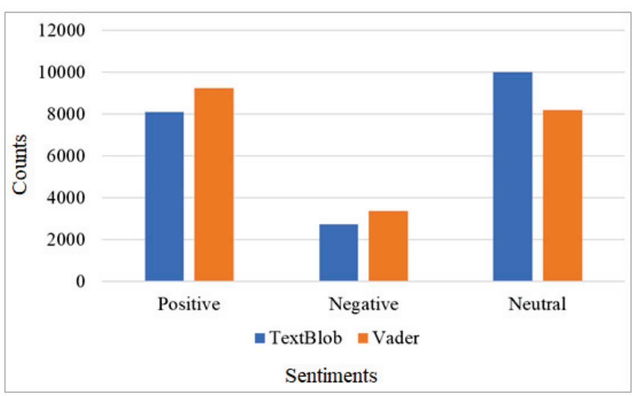


Figure 9. Sentiment ratio in extracted data.

The positive sentiment towards chatGPT is also reflected in the widespread discussions and positive experiences shared by individuals, communities, and social media platforms. People are fascinated by its ability to understand and respond effectively, enhancing user engagement and satisfaction. However, it is important to acknowledge that there are varying opinions and discussions surrounding chatGPT. While most sentiments are positive, some individuals criticize its services and express negative sentiments, particularly concerning its suitability for students. These discussions highlight the need for a further analysis and exploration to address any concerns and improve the tool's effectiveness.

If students rely excessively on ChatGPT, they will lose their capacity to independently compose or generate answers to questions. Students' writing skills may not have improved if they used ChatGPT for projects. As the exam date approaches, individuals have difficulty writing and responding to queries efficiently. There is also the possibility of receiving erroneous information, becoming excessively reliant on technology, and having poor reasoning skills when utilizing ChatGPT. When utilized for personalized learning, ChatGPT may necessitate a comprehensive understanding of the course being taken, the learning preferences of each individual student, and the cultural context in which the students are based. Another negative sentiment regarding ChatGPT is that when students completely rely on AI chatbots to search for specific information about their subject, their level of knowledge does not improve. They cannot advance or increase the topic's knowledge, and it is extremely difficult to maintain concentration when studying. Additionally, students enter data into ChatGPT while looking up specific queries, which could pose a security concern because ChatGPT stores the data that users submit. Over fifty percent of students are motivated to cheat and use ChatGPT to generate information for their submissions. While most students did not admit to using ChatGPT in their writing, integrity may be compromised when ChatGPT generates text.

Additionally, we conducted an analysis using an external sentiment analysis tool called SentimentViz [78]. This tool allowed us to visualize people's perceptions of ChatGPT based on their data. The sentiment analysis results obtained from SentimentViz complemented and validated the findings of the proposed approach. Figure 10 presents visual representations of the sentiment expressed by individuals regarding ChatGPT. This visualization provides further support for the positive sentiment observed in our study and reinforces the credibility of our results.

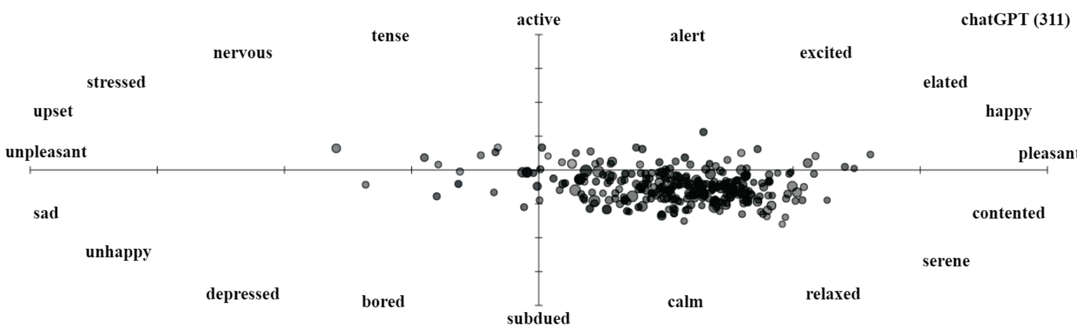


Figure 10. SentimentViz output for chatGPT sentiment.

Discussions regarding the set RQs for this study are also given here.

i. **RQ1:** What are people's sentiments about ChatGPT technology?
Response: The authors analyzed a large dataset of tweets and were able to determine how individuals feel about ChatGPT technology. The results indicate that users have mixed feelings about ChatGPT, with some expressing positive opinions and others expressing negative views. These results provide useful information about how the public perceives ChatGPT and can assist researchers and developers in understanding the chatbot's strengths and weaknesses. The favorable perception of chatGPT is attributable to its advanced language generation features and its ability to become involved in human-like interactions. Individuals are attracted by its cognitive power as well as its ability to effectively respond, thereby increasing user interest and satisfaction. The positive sentiments, like the new openai ChatGPT, writes user-generated content in a better way; it is a great language tool that codes you for your specific queries, etc.

ii. **RQ2:** Which classification model is most effective, such as the proposed transformer-based models, machine learning-based models, and deep learning-based models, for analyzing sentiments about ChatGPT tweets?
Response: The experiments indicate that transformer-based BERT models are more effective and accurate for analyzing sentiments about the ChatGPT tweets. Since transformers make use of self-attention mechanisms, they give the same amount of attention to each component of the sequence that they are processing. They have the ability to virtually process any kind of sequential information. When it comes to natural language processing (NLP), the BERT model takes into account the context of words in both directions (left to right and right to left). Transformers have an in-depth understanding of the meanings of words and are useful for complex problems. In contrast, manual feature engineering, rigorous preprocessing, and a limited dataset are required for machine learning in order to improve accuracy. Additionally, deep learning has a less accurate automatic feature extraction method.

iii. **RQ3:** What are the impacts of ChatGPT on student learning?
Response: The findings show that ChatGPT may have a significant impact on students' learning. ChatGPT's learning capabilities can help students learn when they do not attend school. ChatGPT is not recommended to be used as a substitute for analytical thinking and creative work, but also as a tool to develop research and writing skills. Students' writing skills may not have improved if they relied completely on ChatGPT. There is also the possibility of receiving erroneous information, becoming excessively reliant on technology, and having poor reasoning skills.

iv. **RQ4:** What role does topic modeling play in the sentiment analysis of social media tweets?
Response: Topic modeling refers to an unsupervised statistical method to assess whether or not a particular batch of documents contains any "topics" that are more generic in nature. In order to create a summary that is the most accurate depiction of the document's contents, it extracts the text for commonly used words and phrases. There is a vast amount of unstructured data related to OpenAI ChatGPT, and traditional approaches are incapable of handling such data. Topic modeling can handle and extract meaningful information from unstructured text data efficiently. LDA-based modeling extracts the most discussed topics and prominent positive or negative keywords. It also provides clear information from the large corpus, which is very time-consuming if an individual extracts topics manually.

5. Conclusions

This study conducted a sentiment analysis on ChatGPT-related tweets to gain insight into people's perceptions and opinions. By analyzing a large dataset of tweets, we were able to identify the overall sentiment expressed by users towards ChatGPT. The findings indicate that there are mixed sentiments among users, with some expressing positive views and others expressing negative views about ChatGPT. These results provide valuable insights into the public perception of ChatGPT and can help researchers and developers understand the strengths and weaknesses of the chatbot. Further, this study utilized the BERT model to analyze tweets related to ChatGPT. The BERT model proved to be effective in understanding and classifying sentiments expressed in these tweets. By employing the BERT model, we were able to accurately classify sentiments and gain a deeper understanding of the overall sentiment trends surrounding ChatGPT.

The experimental results demonstrate the outstanding performance of the proposed model, achieving an accuracy of 94.96%. This performance is further validated through k-fold cross-validation and comparison with existing state-of-the-art studies. Our conclusions indicate that the majority of people expressed positive sentiments towards the ChatGPT tool, while a minority had negative sentiments. It was observed that many users appreciate the tool for its assistance across various domains. However, some individuals criticized the ChatGPT tool's services, particularly its suitability for students, expressing negative sentiments in this regard.

This study recognizes the limitation of a relatively small dataset, comprising only 21,515 tweets, which may restrict comprehensive insights. To overcome this limitation, future research will prioritize the collection of a larger volume of data from Twitter and other social media platforms to gain a more accurate understanding of people's perceptions of the trending chatGPT tool. Moreover, the study aims to develop a machine learning approach that incorporates the sentiment analysis, enabling exploration of how such technologies can be developed to mitigate potential societal harm and ensure responsible deployment.

Author Contributions: Conceptualization, S.R. and M.M.; Data curation, M.M. and F.R.; Formal analysis, S.R., F.R., R.S. and I.d.l.T.D.; Funding acquisition, I.d.l.T.D.; Investigation, V.C. and M.G.V.; Methodology, F.R., M.M. and R.S.; Project administration, R.S. and V.C.; Resources, M.G.V. and J.B.B.; Software, M.G.V. and J.B.B.; Supervision, I.d.l.T.D. and I.A.; Validation, J.B.B. and I.A.; Visualization, R.S. and V.C.; Writing—original draft, M.M., R.S., F.R. and S.R.; Writing—review & editing, I.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the European University of Atlantic.

Data Availability Statement: https://www.kaggle.com/datasets/furqanrustam118/chatgpt-tweets.

Conflicts of Interest: The authors declare no conflict of interests.

References

1. Meshram, S.; Naik, N.; Megha, V.; More, T.; Kharche, S. Conversational AI: Chatbots. In Proceedings of the 2021 International Conference on Intelligent Technologies (CONIT), Hubli, India, 25–27 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
2. The Future of Chatbots: 10 Trends, Latest Stats & Market Size. Available online: https://onix-systems.com/blog/6-chatbot-trends-that-are-bringing-the-future-closer (accessed on 23 May 2023).
3. Size of the Chatbot Market Worldwide from 2021 to 2030. Available online: https://www.statista.com/statistics/656596/worldwide-chatbot-market/ (accessed on 23 May 2023).
4. Chatbot Market in 2022: Stats, Trends, and Companies in the Growing AI Chatbot Industry. Available online: https://www.insiderintelligence.com/insights/chatbot-market-stats-trends/ (accessed on 23 May 2023).
5. Malinka, K.; Perešíni, M.; Firc, A.; Hujňák, O.; Januš, F. On the educational impact of ChatGPT: Is Artificial Intelligence ready to obtain a university degree? *arXiv* **2023**, arXiv:2303.11146.
6. George, A.S.; George, A.H. A review of ChatGPT AI's impact on several business sectors. *Partners Univers. Int. Innov. J.* **2023**, *1*, 9–23.
7. Lund, B.D.; Wang, T.; Mannuru, N.R.; Nie, B.; Shimray, S.; Wang, Z. ChatGPT and a new academic reality: Artificial Intelligence-written research papers and the ethics of the large language models in scholarly publishing. *J. Assoc. Inf. Sci. Technol.* **2023**, *74*, 570–581.
8. Kirmani, A.R. Artificial Intelligence-Enabled Science Poetry. *ACS Energy Lett.* **2022**, *8*, 574–576.
9. Cotton, D.R.; Cotton, P.A.; Shipway, J.R. Chatting and cheating: Ensuring academic integrity in the era of ChatGPT. *Innov. Educ. Teach. Int.* **2023**, 1–12. [CrossRef]
10. Tlili, A.; Shehata, B.; Adarkwah, M.A.; Bozkurt, A.; Hickey, D.T.; Huang, R.; Agyemang, B. What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart Learn. Environ.* **2023**, *10*, 15.
11. Edtech Chegg Tumbles as ChatGPT Threat Prompts Revenue Warning. Available online: https://www.reuters.com/markets/us/edtech-chegg-slumps-revenue-warning-chatgpt-threatens-growth-2023-05-02/ (accessed on 23 May 2023).
12. Liu, B. *Sentiment Analysis and Opinion Mining*; Synthesis Lectures on Human Language Technologies; Springer: Cham, Switzerland, 2012; Volume 5, 167p.
13. Medhat, W.; Hassan, A.; Korashy, H. Sentiment analysis algorithms and applications: A survey. *Ain Shams Eng. J.* **2014**, *5*, 1093–1113.
14. Hussein, D.M.E.D.M. A survey on sentiment analysis challenges. *J. King Saud Univ.-Eng. Sci.* **2018**, *30*, 330–338.
15. Lee, E.; Rustam, F.; Ashraf, I.; Washington, P.B.; Narra, M.; Shafique, R. Inquest of Current Situation in Afghanistan Under Taliban Rule Using Sentiment Analysis and Volume Analysis. *IEEE Access* **2022**, *10*, 10333–10348.
16. Lee, E.; Rustam, F.; Washington, P.B.; El Barakaz, F.; Aljedaani, W.; Ashraf, I. Racism detection by analyzing differential opinions through sentiment analysis of tweets using stacked ensemble gcr-nn model. *IEEE Access* **2022**, *10*, 9717–9728. [CrossRef]
17. Mujahid, M.; Lee, E.; Rustam, F.; Washington, P.B.; Ullah, S.; Reshi, A.A.; Ashraf, I. Sentiment analysis and topic modeling on tweets about online education during COVID-19. *Appl. Sci.* **2021**, *11*, 8438. [CrossRef]
18. Tran, A.D.; Pallant, J.I.; Johnson, L.W. Exploring the impact of chatbots on consumer sentiment and expectations in retail. *J. Retail. Consum. Serv.* **2021**, *63*, 102718. [CrossRef]

19. Muneshwara, M.; Swetha, M.; Rohidekar, M.P.; AB, M.P. Implementation of Therapy Bot for Potential Users With Depression During Covid-19 Using Sentiment Analysis. *J. Posit. Sch. Psychol.* **2022**, *6*, 7816–7826.
20. Parimala, M.; Swarna Priya, R.; Praveen Kumar Reddy, M.; Lal Chowdhary, C.; Kumar Poluru, R.; Khan, S. Spatiotemporal-based sentiment analysis on tweets for risk assessment of event using deep learning approach. *Softw. Pract. Exp.* **2021**, *51*, 550–570. [CrossRef]
21. Aslam, N.; Rustam, F.; Lee, E.; Washington, P.B.; Ashraf, I. Sentiment analysis and emotion detection on cryptocurrency related Tweets using ensemble LSTM-GRU Model. *IEEE Access* **2022**, *10*, 39313–39324. [CrossRef]
22. Aslam, N.; Xia, K.; Rustam, F.; Lee, E.; Ashraf, I. Self voting classification model for online meeting app review sentiment analysis and topic modeling. *PeerJ Comput. Sci.* **2022**, *8*, e1141. [CrossRef] [PubMed]
23. Araujo, A.F.; Gôlo, M.P.; Marcacini, R.M. Opinion mining for app reviews: An analysis of textual representation and predictive models. *Autom. Softw. Eng.* **2022**, *29*, 1–30. [CrossRef]
24. Aljedaani, W.; Mkaouer, M.W.; Ludi, S.; Javed, Y. Automatic classification of accessibility user reviews in android apps. In Proceedings of the 2022 7th international conference on data science and machine learning applications (CDMA), Riyadh, Saudi Arabia, 1–3 March 2022; IEEE: Piscataway, NJ, USA, 2022, pp. 133–138.
25. Naeem, M.Z.; Rustam, F.; Mehmood, A.; Ashraf, I.; Choi, G.S. Classification of movie reviews using term frequency-inverse document frequency and optimized machine learning algorithms. *PeerJ Comput. Sci.* **2022**, *8*, e914. [CrossRef]
26. Rustam, F.; Mehmood, A.; Ahmad, M.; Ullah, S.; Khan, D.M.; Choi, G.S. Classification of shopify app user reviews using novel multi text features. *IEEE Access* **2020**, *8*, 30234–30244. [CrossRef]
27. Khalid, M.; Ashraf, I.; Mehmood, A.; Ullah, S.; Ahmad, M.; Choi, G.S. GBSVM: Sentiment classification from unstructured reviews using ensemble classifier. *Appl. Sci.* **2020**, *10*, 2788. [CrossRef]
28. Umer, M.; Ashraf, I.; Mehmood, A.; Ullah, S.; Choi, G.S. Predicting numeric ratings for google apps using text features and ensemble learning. *ETRI J.* **2021**, *43*, 95–108. [CrossRef]
29. Rehan, M.S.; Rustam, F.; Ullah, S.; Hussain, S.; Mehmood, A.; Choi, G.S. Employees reviews classification and evaluation (ERCE) model using supervised machine learning approaches. *J. Ambient Intell. Humaniz. Comput.* **2022**, *13*, 3119–3136. [CrossRef]
30. Al Kilani, N.; Tailakh, R.; Hanani, A. Automatic classification of apps reviews for requirement engineering: Exploring the customers need from healthcare applications. In Proceedings of the 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS), Granada, Spain, 22–25 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 541–548.
31. Srisopha, K.; Phonsom, C.; Lin, K.; Boehm, B. Same app, different countries: A preliminary user reviews study on most downloaded ios apps. In Proceedings of the 2019 IEEE International Conference on Software Maintenance and Evolution (ICSME), Cleveland, OH, USA, 29 September–4 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 76–80.
32. Hossain, M.S.; Rahman, M.F. Sentiment analysis and review rating prediction of the users of Bangladeshi Shopping Apps. In *Developing Relationships, Personalization, and Data Herald in Marketing 5.0*; IGI Global: Pennsylvania, PA USA, 2022; pp. 33–56.
33. Bello, A.; Ng, S.C.; Leung, M.F. A BERT Framework to Sentiment Analysis of Tweets. *Sensors* **2023**, *23*, 506. [CrossRef]
34. Catelli, R.; Pelosi, S.; Esposito, M. Lexicon-based vs. Bert-based sentiment analysis: A comparative study in Italian. *Electronics* **2022**, *11*, 374. [CrossRef]
35. Patel, A.; Oza, P.; Agrawal, S. Sentiment Analysis of Customer Feedback and Reviews for Airline Services using Language Representation Model. *Procedia Comput. Sci.* **2023**, *218*, 2459–2467. [CrossRef]
36. Mujahid, M.; Kanwal, K.; Rustam, F.; Aljadani, W.; Ashraf, I. Arabic ChatGPT Tweets Classification using RoBERTa and BERT Ensemble Model. *Acm Trans. Asian-Low-Resour. Lang. Inf. Process.* **2023**. [CrossRef]
37. Bonifazi, G.; Cauteruccio, F.; Corradini, E.; Marchetti, M.; Sciarretta, L.; Ursino, D.; Virgili, L. A Space-Time Framework for Sentiment Scope Analysis in Social Media. *Big Data Cogn. Comput.* **2022**, *6*, 130. [CrossRef]
38. Bonifazi, G.; Corradini, E.; Ursino, D.; Virgili, L. Modeling, Evaluating, and Applying the eWoM Power of Reddit Posts. *Big Data Cogn. Comput.* **2023**, *7*, 47. [CrossRef]
39. Messaoud, M.B.; Jenhani, I.; Jemaa, N.B.; Mkaouer, M.W. A multi-label active learning approach for mobile app user review classification. In Proceedings of the Knowledge Science, Engineering and Management: 12th International Conference, KSEM 2019, Athens, Greece, 28–30 August 2019; Proceedings, Part I 12; Springer: Berlin/Heidelberg, Germany, 2019; pp. 805–816.
40. Fuad, A.; Al-Yahya, M. Analysis and classification of mobile apps using topic modeling: A case study on Google Play Arabic apps. *Complexity* **2021**, *2021*, 1–12. [CrossRef]
41. Venkatakrishnan, S.; Kaushik, A.; Verma, J.K. Sentiment analysis on google play store data using deep learning. In *Applications of Machine Learning*; Springer: Singapore, 2020; pp. 15–30.
42. Alam, S.; Yao, N. The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. *Comput. Math. Organ. Theory* **2019**, *25*, 319–335. [CrossRef]
43. Vijayarani, S.; Ilamathi, M.J.; Nithya, M. Preprocessing techniques for text mining-an overview. *Int. J. Comput. Sci. Commun. Netw.* **2015**, *5*, 7–16.
44. R, S.; Mujahid, M.; Rustam, F.; Mallampati, B.; Chunduri, V.; de la Torre Díez, I.; Ashraf, I. Bidirectional encoder representations from transformers and deep learning model for analyzing smartphone-related tweets. *PeerJ Comput. Sci.* **2023**, *9*, e1432. [CrossRef]
45. Kadhim, A.I. An evaluation of preprocessing techniques for text classification. *Int. J. Comput. Sci. Inf. Secur.* **2018**, *16*, 22–32.
46. Loria, S. Textblob Documentation. Release 0.15. 2018. Volume 2. Available online: https://buildmedia.readthedocs.org/media/pdf/textblob/latest/textblob.pdf (accessed on 23 May 2023).

47. Borg, A.; Boldt, M. Using VADER sentiment and SVM for predicting customer response sentiment. *Expert Syst. Appl.* **2020**, *162*, 113746. [CrossRef]
48. Karamibekr, M.; Ghorbani, A.A. Sentiment analysis of social issues. In Proceedings of the 2012 International Conference on Social Informatics, Alexandria, VA, USA, 14–16 December 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 215–221.
49. Kumar, Y.; Koul, A.; Singla, R.; Ijaz, M.F. Artificial intelligence in disease diagnosis: A systematic literature review, synthesizing framework and future research agenda. *J. Ambient. Intell. Humaniz. Comput.* **2023**, *14*, 8459–8486 [CrossRef]
50. Shafique, R.; Aljedaani, W.; Rustam, F.; Lee, E.; Mehmood, A.; Choi, G.S. Role of Artificial Intelligence in Online Education: A Systematic Mapping Study. *IEEE Access* **2023**, *11*, 52570–52584. [CrossRef]
51. George, A.; Ravindran, A.; Mendieta, M.; Tabkhi, H. Mez: An adaptive messaging system for latency-sensitive multi-camera machine vision at the iot edge. *IEEE Access* **2021**, *9*, 21457–21473. [CrossRef]
52. Ravindran, A.; George, A. An edge datastore architecture for Latency-Critical distributed machine vision applications. In Proceedings of the USENIX Workshop on Hot Topics in Edge Computing (HotEdge 18), Boston, MA, USA, 10 July 2018.
53. Kadhim, A.I. Survey on supervised machine learning techniques for automatic text classification. *Artif. Intell. Rev.* **2019**, *52*, 273–292. [CrossRef]
54. Chen, H.; Wu, L.; Chen, J.; Lu, W.; Ding, J. A comparative study of automated legal text classification using random forests and deep learning. *Inf. Process. Manag.* **2022**, *59*, 102798. [CrossRef]
55. Schröder, C.; Niekler, A. A survey of active learning for text classification using deep neural networks. *arXiv* **2020**, arXiv:2008.07267.
56. Prabhat, A.; Khullar, V. Sentiment classification on big data using Naïve Bayes and logistic regression. In Proceedings of the 2017 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 5–7 January 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–5.
57. Valencia, F.; Gómez-Espinosa, A.; Valdés-Aguirre, B. Price movement prediction of cryptocurrencies using sentiment analysis and machine learning. *Entropy* **2019**, *21*, 589. [CrossRef] [PubMed]
58. Zharmagambetov, A.S.; Pak, A.A. Sentiment analysis of a document using deep learning approach and decision trees. In Proceedings of the 2015 Twelve International Conference on Electronics Computer and Computation (ICECCO), Almaty, Kazakhstan, 27–30 September 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 1–4.
59. Shah, K.; Patel, H.; Sanghvi, D.; Shah, M. A comparative analysis of logistic regression, random forest and KNN models for the text classification. *Augment. Hum. Res.* **2020**, *5*, 12. [CrossRef]
60. Tiwari, D.; Singh, N. Ensemble approach for twitter sentiment analysis. *IJ Inf. Technol. Comput. Sci.* **2019**, *8*, 20–26. [CrossRef]
61. Arya, V.; Mishra, A.K.M.; González-Briones, A. Analysis of sentiments on the onset of COVID-19 using machine learning techniques. *ADCAIJ Adv. Distrib. Comput. Artif. Intell. J.* **2022**, *11*, 45–63. [CrossRef]
62. Severyn, A.; Moschitti, A. Unitn: Training deep convolutional neural network for twitter sentiment classification. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Denver, CO, USA, 4–5 June 2015; pp. 464–469.
63. Seo, S.; Kim, C.; Kim, H.; Mo, K.; Kang, P. Comparative study of deep learning-based sentiment classification. *IEEE Access* **2020**, *8*, 6861–6875. [CrossRef]
64. Nowak, J.; Taspinar, A.; Scherer, R. LSTM recurrent neural networks for short text and sentiment classification. In Proceedings of the Artificial Intelligence and Soft Computing: 16th International Conference, ICAISC 2017, Zakopane, Poland, 11–15 June 2017; Proceedings, Part II 16; Springer: Cham, Switzerland, 2017; pp. 553–562.
65. Mujahid, M.; Rustam, F.; Alasim, F.; Siddique, M.; Ashraf, I. What people think about fast food: Opinions analysis and LDA modeling on fast food restaurants using unstructured tweets. *PeerJ Comput. Sci.* **2023**, *9*, e1193. [CrossRef]
66. Tang, D.; Qin, B.; Liu, T. Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1422–1432.
67. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
68. Tenney, I.; Das, D.; Pavlick, E. BERT rediscovers the classical NLP pipeline. *arXiv* **2019**, arXiv:1905.05950.
69. González-Carvajal, S.; Garrido-Merchán, E.C. Comparing BERT against traditional machine learning text classification. *arXiv* **2020**, arXiv:2005.13012.
70. Cruz, J.C.B.; Cheng, C. Establishing baselines for text classification in low-resource languages. *arXiv* **2020**, arXiv:2005.02068.
71. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5753–5763.
72. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
73. Amaar, A.; Aljedaani, W.; Rustam, F.; Ullah, S.; Rupapara, V.; Ludi, S. Detection of fake job postings by utilizing machine learning and natural language processing approaches. *Neural Process. Lett.* **2022**, *54*, 2219–2247 [CrossRef]
74. Jelodar, H.; Wang, Y.; Yuan, C.; Feng, X.; Jiang, X.; Li, Y.; Zhao, L. Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimed. Tools Appl.* **2019**, *78*, 15169–15211. [CrossRef]
75. Wadhwa, S.; Babber, K. Performance comparison of classifiers on twitter sentimental analysis. *Eur. J. Eng. Sci. Technol.* **2021**, *4*, 15–24. [CrossRef]

76. SemEvel2013 Dataset. Available online: https://www.kaggle.com/datasets/azzouza2018/semevaldatadets?select=semeval-2013-train-all.csv (accessed on 23 May 2023).
77. Rustam, F.; Ashraf, I.; Mehmood, A.; Ullah, S.; Choi, G.S. Tweets classification on the base of sentiments for US airline companies. *Entropy* **2019**, *21*, 1078. [CrossRef]
78. Sentiment Viz: Tweet Sentiment Visualization. Available online: https://www.csc2.ncsu.edu/faculty/healey/tweet_viz/tweet_app/ (accessed on 23 May 2023).

information

MDPI

Article

Computing the Sound–Sense Harmony: A Case Study of William Shakespeare's Sonnets and Francis Webb's Most Popular Poems

Rodolfo Delmonte

Department of Linguistics and Comparative Cultural Studies, Ca Foscari University of Venice, 30123 Venice, Italy; delmont@unive.it

Abstract: Poetic devices implicitly work towards inducing the reader to associate intended and expressed meaning to the sounds of the poem. In turn, sounds may be organized a priori into categories and assigned presumed meaning as suggested by traditional literary studies. To compute the degree of harmony and disharmony, I have automatically extracted the sound grids of all the sonnets by William Shakespeare and have combined them with the themes expressed by their contents. In a first experiment, sounds have been associated with lexically and semantically based sentiment analysis, obtaining an 80% of agreement. In a second experiment, sentiment analysis has been substituted by Appraisal Theory, thus obtaining a more fine-grained interpretation that combines dis-harmony with irony. The computation for Francis Webb is based on his most popular 100 poems and combines automatic semantically and lexically based sentiment analysis with sound grids. The results produce visual maps that clearly separate poems into three clusters: negative harmony, positive harmony and disharmony, where the latter instantiates the need by the poet to encompass the opposites in a desperate attempt to reconcile them. Shakespeare and Webb have been chosen to prove the applicability of the method proposed in general contexts of poetry, exhibiting the widest possible gap at all linguistic and poetic levels.

Keywords: specialized NLP system for poetry; automatic poetic analysis; visualization of linguistic and poetic content; Sound–Sense matching algorithm; phonetic and phonological analysis; automatic lexical and semantic sentiment analysis; computing irony; appraisal theory framework

Citation: Delmonte, R. Computing the Sound–Sense Harmony: A Case Study of William Shakespeare's Sonnets and Francis Webb's Most Popular Poems. *Information* **2023**, *14*, 576. https://doi.org/10.3390/info14100576

Academic Editors: Peter Revesz and Tudor Groza

Received: 7 July 2023
Revised: 21 September 2023
Accepted: 5 October 2023
Published: 20 October 2023

1. Introduction

In this article, I will propose a totally new technique to assess and appreciate poetry, the ***Algorithm for Sound and Sense Harmony (henceforth ASSH)***. The main tenet of this paper is the existence of a hidden and systematic plan by important poets like Shakespeare and Webb to organize rhyming structures in accordance with a principle of overall ASSH. What is meant here by "Sound Harmony" is the presence of rhymes whose sound—the stressed vowel that is dominant—belongs to the four sound classes that may comprise all vowel sounds, phonologically speaking, i.e., low, mid, high-front, high-back, or part of them. In addition, the "Sound Harmony" is composed with Sense to make up the ASSH, where the choice of sounds reflects the contents of the poem, as it may be represented by main topics, intended meaning and overall sentiment. The same argument is presented for the presence of the three main classes of consonants, i.e., continuants, sonorants, obstruents and their partition into voiced vs. unvoiced. The choice to favor the presence of one class vs. another is to be interpreted as a way to highlight sense-related choices of words that will either accompany or contrast with Sounds. In particular, we associate different mood—following traditional judgements—to vowels and consonants according to their class, as follows:

1. Low and mid vowels evoke a sense of brightness, peace and serenity;

2. High, front and back vowels evoke a sense of surprise, seriousness, rigor and gravity;
3. Obstruent and unvoiced consonants evoke a sense of harshness and severity;
4. Sonorant and continuant consonants evoke a sense of pleasure, softness and lightness.

Classes 1 and 4 will be regarded in the same area of positive thinking, while classes 2 and 3 will more naturally be accompanied by negative sentiment. Of course, it may be the case that crossed matches with classes belonging to opposite types will take place more or less frequently, indicating the need to reconcile opposite feelings in the same poem. This is what happens in both Shakespeare's and Webb's poems, as will be shown in the sections below.

It is important to highlight the role of sounds in poetry, which is paramount for the creation of poetic and rhetoric devices. Rhyme, alliterations, assonances and consonances may contribute secondary and, in some cases, primary additional meaning by allowing words which are not otherwise syntactically or semantically related to share part if not all of their meaning by means of metaphors and other similar devices. Thus, most of the difficult work of every poet is devoted to the choice of the appropriate word to use for rhyming purposes, mainly, but also for the other important devices mentioned above.

In the case of Shakespeare, for the majority of the sonnets, he took care of choosing words for the rhymes contributing sounds to the four varieties, thus producing a highly varied sound harmony. We will discuss this in the sections below, paying attention to associate choice of one class vs. another, with choice of specific themes or words. This important feature of the sonnets has never been noticed by literary critics in the past. Reasons for this apparent lack of attention may be imputed to the existence of two seemingly hindering factors: a former factor is the use of words which had a double pronunciation at the time, as for instance LOVE which could be pronounced as MOVE in addition to its current pronunciation. The latter factor regards the existence of a high—in comparison with other poets of the same Elizabethan period—percentage of a variable we call Rhyme Repetition Rate (TripleR), which indicates the use of the same "head" word—i.e., the rhyming word that precedes the alternate rhyme scheme—or sometimes the same couple of words.

The use of mood and related colours associated with sound in poetry has a long tradition. Rimbaud composed a poem devoted to "Vowels", where colours were associated with each of the main five vowels. Roman Jakobson [1,2] and Mazzeo [3] wrote extensively about the connection between sound and colour in a number of papers. Fónagy [4] wrote an article in which he explicitly connected the use of certain types of consonant sounds associated with certain moods: unvoiced and obstruent consonants are associated with aggressive mood; sonorants with tender moods. Macdermott [5] identified a specific quality associated with "dark" vowels, i.e., back vowels, that of being linked with dark colours, mystic obscurity, hatred and struggle. As a result, we are using darker colours to highlight back and front vowels as opposed to low and middle vowels, the latter with light colours. The same applies to representing unvoiced and obstruent consonants as opposed to voiced and sonorants. But as Tsur (see [6], p. 15) notes, this sound–colour association with mood or attitude has no real significance without a link to semantics.

In one of the visual outputs produced by our system, SPARSAR—presented in a section below, the Semantic Relational View, we are using dark colours for *concrete* referents vs. *abstract* ones [7] with lighter colours; and dark colours also for *negatively* marked words as opposed to *positively* marked ones with lighter colours. The same strategy applies to other poetic maps: this technique has certainly the good quality of highlighting opposing differences at some level of abstraction. Our approach is not comparable to work by Saif Mohammad [8], where colours are associated with words on the basis of what their mental image may suggest to the mind of annotators hired via Mechanical Turk (Amazon Mechanical Turk (MTurk) is a crowdsourcing marketplace that makes it easier for individuals and businesses to outsource their processes and jobs). The resource only contains word–colour association for some 12,000 entries over the 27 K items listed. It is, however, comparable to a long list of other attempts at depicting phonetic differences in poems as will be discussed further on. With this experiment, I intend to verify the number of poems in Webb's corpus in which it is possible to establish a relationship between semantic content in terms of negative vs. positive sense—usually referred to with one word as "the sentiment"—and the sound produced by syllables in particular, stressed ones. We adopt a lexical approach, mainly using the database of 40 K entries made available by Brysbaert et al. 2014.

Thus, I will match the negative sentiment expressed by the words' sense with sad-sounding rhymes and poetic devices as a whole, and the opposite for positive sentiment by scoring and computing the ratios. I repeat here below the way in which I organized vowel and consonant sounds:

- Low, middle, high-front, high-back

Where I identify the two classes low and middle as promoting positive feelings, and the two high as inducing negative ones. As to the consonants, I organized the sounds into three main classes and two types:

- *Obstruents* (plosives, affricates), *continuants* (fricatives), *sonorants* (liquids, vibrants, approximants) plus the distinction into
- Voiced vs. unvoiced.

In this case, the ratios are computed dividing the sum of continuants and sonorants by the number of obstruents; and the second parameter will be the ratio obtained by dividing number of voiced by unvoiced. Whenever the value of the ratios is above 1, positive results are obtained; the contrary applies whenever values are below 1. In this way, counting results is immediate and very effective.

The Result section of the paper has a first rather lengthy subsection dedicated to the problem of rhyming structure which in the Sonnets constitutes the basic framework onto which all the subsequent reasoning is founded. Another subsection is dedicated to associating rhyming schemes with different themes as they have evolved in time. We dedicate a subsection to explaining the importance of the lexical approach in organizing the rules for the system SPARSAR, which derives the final vowel and consonant grids that allow us to make the first comparison. The lexical and semantic approach to deriving the sentiment of each sonnet operates a first subdivision of harmonic and disharmonic sonnets into negatively vs. positively marked sonnets. Measuring correlations reveals a constant contrasting attitude induced by the sound–sense agreement, which we interpret as an underlying hidden intention to produce some form of ironic mood in Shakespeare's sonnets.

Detecting irony requires a much deeper and accurate analysis of the semantic and the pragmatics of the sonnets. We proceed into two separate but conjoined ways: producing a gold standard of the sonnets and then manually annotating each sonnet using the highly sophisticated labeling system proposed by the Appraisal Theory Framework, ATF that we introduce briefly in Section 3.2.4. Matching the empirical approach and the automatic analysis confirms the overall underlying hypothesis: the sound–sense disharmony has a fundamental task, that of suggesting an underlying ironic attitude which is at the heart of all the sonnets. ATF makes available a more fine-grained approach which takes non-literal language into due account, thus improving on the previous method of sentiment-based analysis (see Martin et al. [9] and Toboada et al. [10]).

2. Materials and Methods

In this section, I will present the system SPARSAR and the pipeline of modules that allow it to carry out the complex analysis reported above.

2.1. SPARSAR—A System for Poetry Analysis and Reading

SPARSAR [11] produces a deep analysis of each poem at different levels: it works at the sentence level at first, then at the verse level and finally at the stanza level (see Figure 1 below). The structure of the system is organized as follows: the input text is processed at first at a syntactic and semantic level and grammatical functions are evaluated. Then, the poem is translated into a phonetic form, preserving its visual structure and its subdivision into verses and stanzas. Phonetically translated words are associated with mean duration values taking into account position in the word and stress. At the end of the analysis of the poem, the system can measure the following parameters: mean verse length in terms of msec. and in number of feet. The latter is derived by a verse representation into metrical structure. Another important component of the analysis of rhythm is constituted by the algorithm that measures and evaluates rhyme schemes at the stanza level and then the overall rhyming structure at the poem level. In addition, the system has access to a restricted list of typical pragmatically marked phrases and expressions that are used to convey specific discourse function and speech acts, and need specialized intonational contours.

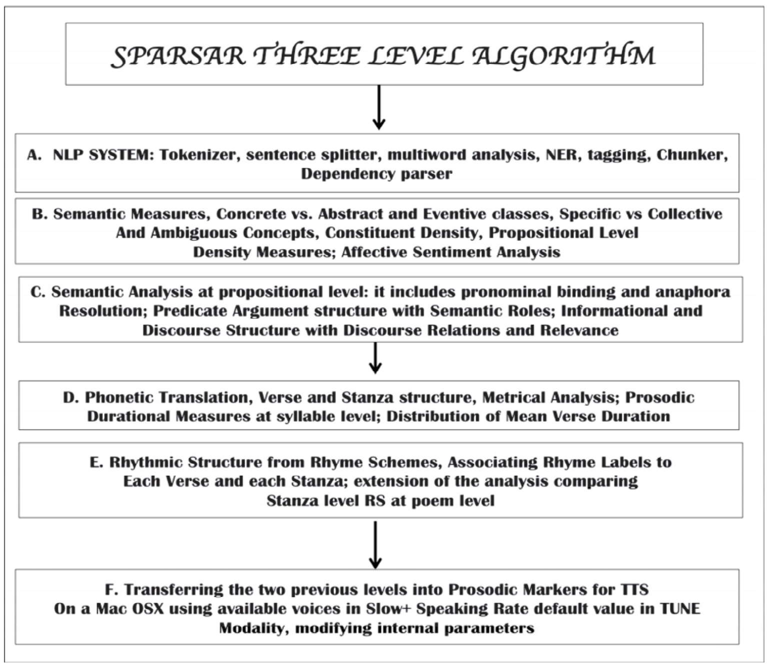


Figure 1. Architecture of *SPARSAR* with main pipeline organized into three levels.

We use the word "expressivity" [12], referring to the following levels of intervention of syntactic–semantic and pragmatic knowledge, which include the following:

- Syntactic heads which are quantified expressions;
- Syntactic heads which are preverbal subjects;
- Syntactic constituents that starts and ends an interrogative or an exclamative sentence;
- Distinguish realis from irrealis mood;
- Distinguish deontic modality including imperative, hortative, optative, deliberative, jussive, precative, prohibitive, propositive, volitive, desiderative, imprecative, directive and necessitative, etc.;
- Distinguish epistemic modality including assumptive, deductive, dubitative, alethic, inferential, speculative, etc.;
- Any sentence or phrase which is recognized as a formulaic or frozen expression with specific pragmatic content;
- Subordinate clauses with inverted linear order, distinguishing causal from hypotheticals and purpose complex sentences;
- Distinguishing parentheticals from appositives and unrestricted relatives;

- Discourse Structure to distinguish satellite and dependent clauses from the main clause;
- Discourse structure to check for discourse moves—up, down and parallel;
- Discourse relations to tell foreground relations from backgrounds;
- Topic structure to tell the introduction of a new topic or simply a change at relational level.

Current TTS only takes into account information coming from punctuation and, in some cases, from tagging. This hampers the possibility to capture the great majority of structures listed above. In addition, they do not adequately consider ambiguity of punctuation: for instance, the comma is a highly ambiguous punctuation mark with a whole set of different functions which are associated with specific intonational contours, and require semantic- and discourse-level knowledge to disentangle ambiguity. In general, punctuation marks like question and exclamative marks, are always used to modify the prosody of the previous word, which on the contrary is clearly insufficient to reproduce such pragmatically marked utterances and would encompass the whole sentence from its beginning word.

2.2. The Modules for Syntax and Semantics

The system uses a modified version of *VENSES*, a semantically oriented NLP pipeline [13]. It is accompanied by a module that works at sentence level and produces a whole set of analyses at quantitative, syntactic and semantic levels. As regards syntax, the system makes available chunks and dependency structures. Then, the system introduces semantics both in the version of a classifier and by isolating the verbal complex in order to verify propositional properties, like presence of negation, to compute factuality from a crosscheck with modality, aspectuality—that is derived from the lexica—and tense. On the other hand, the classifier has two different tasks: separating concrete from abstract nouns, and identifying highly ambiguous from singleton concepts (from number of possible meanings from WordNet and other similar repositories). Eventually, the system carries out a sentiment analysis of the poem, thus contributing a three-way classification: neutral, negative, and positive that can be used as a powerful tool for prosodically related purposes.

Semantics in our case not only refers to predicate–argument structure, negation scope, quantified structures, anaphora resolution and other similar items. It essentially refers to a propositional-level analysis, which is the basis for discourse structure and discourse semantics contained in discourse relations. It also paves the way for a deep sentiment or affective analysis of every utterance, which alone can take into account the various contributions that may come from syntactic structures like NPs and Aps, where affectively marked words may be contained. Their contribution needs to be computed in a strictly compositional manner with respect to the meaning associated with the main verb, where negation may be lexically expressed or simply lexically incorporated in the verb meaning itself.

In Figure 1 above the architecture of the deep system for semantic and pragmatic processing, in which phonetics are shown, prosodics and NLP are deeply interwoven. The system does low-level analyses before semantic modules are activated, that is tokenization, sentence splitting, and multiword creation from a large lexical database. Then, chunking and syntactic constituency parsing is conducted using a rule-based recursive transition network: the parser works in a cascaded recursive way to include higher syntactic structures up to the sentence and complex sentence level. These structures are then passed to the first semantic mapping algorithm that looks for subcategorization frames in the lexica freely made available for English, including a proprietor lexicon of some 10 K entries, with most frequent verbs, adjectives and nouns, also containing a detailed classification of all grammatical or function words. This mapping is performed following LFG principles [14,15], where c-structure is mapped onto f-structure, thus obeying uniqueness, completeness and coherence. The output of this mapping is a rich dependency structure, which contains information related to implicit arguments as well, i.e., subjects of infinitivals, participials and gerundives. LFG representation also has a semantic role associated with each grammatical function, which is used to identify the syntactic head lemma uniquely in the sentence. When fully coherent and complete predicate argument structures have been built, pronominal binding and anaphora resolution algorithms are fired. The coreferential processed are activated at the semantic level. Discourse-level computation is conducted at the propositional level by building a vector of features associated with the main verb of each clause. They include information about tense, aspect, negation, adverbial modifiers, and modality. These features are then filtered through a set of rules which have the task to classify a proposition as either objective/subjective, factual/nonfactual, foreground/background. In addition, every lexical predicate is evaluated with respect to a class of discourse relations. Eventually, discourse structure is built, according to criteria of clause dependency where a clause can be classified either as coordinate or subordinate.

2.3. The Modules for Phonetic and Prosodic Analysis

The second set of modules is a rule-based system that converts graphemes of each poem into phonetic characters; it divides words into stressed/unstressed syllables and computes rhyming schemes at the line and stanza level. To this end, it uses grapheme-to-phoneme translations made available by different sources, amounting to some 500 K entries, and include the CMU dictionary (Freely downloadable from http://www.speech.cs.cmu.edu/cgi-bin/cmudict accessed on 6 July 2023), MRC Psycholinguistic Database (Freely downloadable from https://websites.psychology.uwa.edu.au/school/mrcdatabase/uwa_mrc.htm accessed on 6 July 2023), Celex Database [16], plus a proprietor database made of some 20,000 entries. Out-of-vocabulary words are computed by means of a prosodic parser implemented in a previous project [17], containing a big pronunciation dictionary which covers 170,000 entries, approximately. Besides the need to cover the majority of grapheme-to-phoneme conversions through the use of appropriate dictionaries, the remaining problems to be solved are related to ambiguous homographs like “import” (verb) and “import” (noun), and are treated on the basis of their lexical category derived from previous tagging. Eventually, there is always a certain number of out-of-vocabulary words (OOVW). The simplest case is constituted by differences in spelling determined by British vs. American pronunciation. This is taken care of by a dictionary of graphemic correspondences. However, whenever the word is not found, the system proceeds by morphological decomposition, splitting at first the word from its prefix and if that still does not work, its derivational suffix. As a last resource, an orthographically based version of the same dictionary is used to try and match the longest possible string in coincidence with the current OOVW. Then, the remaining portion of the word is dealt with by guessing its morphological nature, and if that fails, a grapheme-to-phoneme parser is used. Some of the OOVWs that have been reconstructed by means of the recovery strategy explained above are wayfarer, gangrened, krog, copperplate, splendor, filmy, seraphic, seraphine, and unstarred.

Other words we had to reconstruct are shrive, slipstream, fossicking, unplotted, corpuscle, thither, wraiths, etc. In some cases, the problem that made the system fail was the presence of a syllable which was not available in our database of syllable durations, *VESD* [17]. This problem has been coped with by manually inserting the missing syllable and by computing its duration from the component phonemes, or from the closest similar syllable available in the database. We only had to add 12 new syllables for a set of approximately 1000 poems that the system computed.

The system has no limitation on type of poetic and rhetoric devices; however, it is dependent on language: Italian line verse requires a certain number of beats and metric accents which are different from the ones contained in an English iambic pentameter. Rules implemented can demote or promote word-stress on a certain syllable depending on the selected language, line-level syllable length and contextual information. This includes knowledge about a word being part of a dependency structure either as dependent or as head.

As R. Tsur [18] comments in his introduction to his book, iambic pentameter has to be treated as an abstract pattern and no strict boundary can be established. The majority of famous English poets of the past, while using iambic pentameter, have introduced violations, which in some cases—as for Milton's Paradise Lost—constitute the majority of verse patterns. Instead, the prosodic nature of the English language needs to be addressed, at first. English is a stress-timed language as opposed to Spanish or Italian which are syllable-timed languages. As a consequence, what really matters in the evaluation of iambic pentameters is the existence of a certain number of beats—5 in normal cases, but also 4 in deviant ones. Unstressed syllables can number higher, as for instance in the case of exceptional feminine rhyme or double rhyme, which consists of a foot made of a stressed and an unstressed syllable (very common in Italian) ending the line—this is also used by Greene et al. [19] to loosen the strict iambic model. These variations are made to derive from elementary two-syllable feet, the iamb, the trochee, the spondee, and the pyrrich. According to the author, these variations are not casual, they are all motivated by the higher syntactic–semantic structure of the phrase. So, there can be variations as long as they are constrained by a meaningful phrase structure.

In our system, in order to allow for variations in the metrical structure of any line, we operate on the basis of syntactic dependency and have a stress demotion rule to decide whether to demote stress on the basis of contextual information. The rule states that word stress can be demoted in dependents in adjacency with their head in case they are monosyllabic words. In addition, we also have a promotion rule that promotes function words which require word stress. This applies typically to ambiguously tagged words, like "there", which can be used as an expletive pronoun in preverbal position, and be unstressed; but, it can also be used as locative adverb, in that case in postverbal position, and be stressed. For all these ambiguous cases, but also for homographs not homophones, tagging and syntactic information is paramount.

Our rule system tries to avoid stress clashes and prohibits sequences of three stressed/three unstressed syllables unless the line syntactic–semantic structure allow it to be interpreted otherwise. Generally speaking, prepositions and auxiliary verbs may be promoted; articles and pronouns never. An important feature of English vs. Italian is length of words in terms of syllables. As may be easily gathered, English words have a high percentage of one-syllable words when compared to Italian which, on the contrary, has a high percentage of 3/4-syllable words.

2.4. Computing Metrical Structure and Rhyming Scheme

Any poem can be characterized by its rhythm which is also revealing of the poet's peculiar style. In turn, the poem's rhythm is based mainly on two elements: meter, that is distribution of stressed and unstressed syllables in the verse, presence of rhyming and other poetic devices like alliteration, assonance, consonance, enjambments, etc., which contribute to poetic form at the stanza level. This level is combined then with syntax and semantics to produce the adequate breath groups and consequent subdivision: these will usually coincide with line-stop words, but they may continue to the following line by means of enjambments.

As discussed above, see Figure 1, the analysis starts by translating every poem into its phonetic form. After processing the whole poem on a line-by-line basis and having produced all phonemic transcription, the system looks for poetic devices. Here, assonances, consonances, alliterations and rhymes are analysed and then evaluated. Here, metrical structure is computed, that is the alternation of beats: this is performed by considering all function or grammatical words which are monosyllabic as unstressed. In particular, "0" is associated with all unstressed syllables, and a value of "1" to all stressed syllables, thus including both primary and secondary stressed syllables. Syllable building is a discovery process starting from longest possible phone sequences to shortest one. This is performed heuristically trying to match pseudo syllables with the syllable list. Matching may fail and will then result in a new syllable which has not been previously met. The assumption is that any syllable inventory will be deficient, and will never be sufficient to cover the whole spectrum of syllables available in the English language. For this reason, a certain number of phonological rules has been introduced in order to account for any new syllable that may appear. Also, syntactic information is taken advantage of, which is computed separately to highlight chunks' heads as produced by the bottomup parser. In that case, stressed syllables take maximum duration values. Dependent words, on the contrary, are "demoted" and take minimum duration values.

Metrical structure is used to evaluate its distribution in the poem by means of statistical measures. As a final consideration, we discovered that even in the same poem, it is not always possible to find that all lines have an identical number of syllables, identical number of metrical feet and identical metrical verse structure. If we consider the sequence "01" as representing the typical iambic foot, and the iambic pentameter as the typical verse metre of English poetry, there is no poem strictly respecting it in our analyses. On the contrary, we found trochees, "10", dactyls, "100", anapests, "001"and spondees, "11". At the end of the computation, the system is used to measure two important indices: "mean verse length" and "mean verse length in no. of feet", that is, mean metrical structure.

Additional measures that we are able to produce are related to rhyming devices. Since we consider it important to take into account structural internal rhyming schemes and their persistence in the poem, the algorithm makes available additional data derived from two additional components: word repetition and rhyme repetition at the stanza level. Sometimes, "refrain" may also apply, that is, the repetition of an entire line of verse. Rhyming schemes together with metrical length are the strongest parameters to consider when assessing similarity between two poems.

Eventually, the internal structure of metrical devices used by the poet can be reconstructed: in some cases, stanza repetition at the poem level may also apply. To create the rhyming scheme, couples of rhyming lines are searched by trying a match recursively of each final phonetic word with the following ones, starting from the closest to the one that is further apart. Each time, both rhyming words and their distance are registered. In the following pass, the actual final line numbers are reconstructed and then an indexed list of couples, line number–rhyming line for all the lines is produced, including stanza boundaries. Eventually, alphabetic labels are assigned to each rhyming verse starting from A to Z. A simple alphabetic incremental mechanism updates the rhyme label. This may go beyond the limits of the alphabet itself and in that case, double letters are used.

2.5. From Sentiment Analysis to the Deep Pragmatic Approach by ATF

We based a first approach to detecting sound–sense harmony on sentiment analysis, which in our case encompasses both a lexical and a semantic analysis at the propositional level. More generally speaking, computational research on sentiment analysis has been based on the use of shallow features with a binary choice to train statistical model [20] that, when optimized for a particular task, will produce acceptable performance. However, generalizing the model to new texts is a hard task and, in addition, the sonnets contain a lot of nonliteral language. The other common approach used to detect irony, in the majority of the cases, is based on polarity detection. Sentiment analysis [21,22] is in fact an indiscriminate labeling of texts either on a lexicon basis or on a supervised feature basis, where in both cases, it is just a binary—ternary or graded—decision that has to be made. This is certainly not explanatory of the phenomenon and will not help in understanding what it is that causes humorous reactions to the reading of an ironic piece of text. It certainly is of no help in deciding which phrases, clauses or just multiwords or simply words, contribute to create the ironic meaning (see [23]).

Shakespeare's Sonnets are renowned for being full of ironic content [24,25] and for their ambiguity, thus sometimes reverting the overall interpretation of the sonnet. Lexical ambiguity, i.e., a word with several meanings, emanates from the way in which the author uses words that can be interpreted in more ways not only because they are inherently polysemous, but because sometimes the additional meaning they evoke can sometimes be derived on the basis of the sound, i.e., homophone (see "eye" and "I" in sonnet 152). The sonnets are also full of metaphors which many times require contextualising the content to the historical Elizabethan life and society. Furthermore, there is an abundance of words related to specific language domains in the sonnets. For instance, there are words related to the language of economy, war, nature and to the discoveries of the modern age, and each of these words may be used as a metaphor of love. Many of the sonnets are organized around a conceptual contrast, an opposition that runs parallel and then diverges, sometimes with the use of the rhetorical figure of the chiasmus. It is just this contrast that generates irony, sometimes satire, sarcasm, and even parody. Irony may be considered in turn as what one means using language that normally signifies the opposite, typically for humorous or emphatic effect; and a state of affairs or an event that seems contrary to what one expects and is amusing as a result. As to sarcasm, this may be regarded the use of irony to mock or convey contempt. Parody is obtained by using the words or thoughts of a person but adapting them to a ridiculously inappropriate subject. There are several types of irony, though we select verbal irony which, in the strict sense, is saying the opposite of what you mean for outcome, and it depends on the extra-linguistic context [26]. As a result, satire and irony are slightly overlapping but constitute two separate techniques; eventually, sarcasm can be regarded as a specialization or a subset of irony. It is important to remark that in many cases, these linguistic structures may require the use of non-literal or figurative language, i.e., the use of metaphors.

Joining sentiment, irony and sound as they could have been heard by Elizabethan audiences is what makes the Sonnets so special even today, and our paper succeeds in clarifying the peculiarities of the at the same time deep and shallow combination of factors intertwined to produce the final glamorous result that every sonnet does also today.

3. Results

This section will present results of the analysis of Shakespeare's sonnets at first and then of Webb's poems highlighting all cases of harmony and disharmony with relation to theme and meaning intended in the poem.

3.1. Sound Harmony in the Sonnets

We postulate the existence of a hidden plan in Shakespeare's poetic approach, to abide to a harmonic principle that requires all varieties of sound classes to be present and to relate by virtue of a sound–meaning correspondence, to thematic and meaning development in the sonnet. To discover such a plan, we analysed the phonetic representation of the rhyming words of all sonnets using SPARSAR—the system that analyzes automatically any poem, see below—and then organized the results of all vowel sounds into the four classes mentioned above. We did the same with consonants and consonant clusters in order to obtain a sound grid that is complete and retains as much complexity as possible of each poem and compared it with sense-related analyses.

However, in order to produce such a result, almost 500 phonologically ambiguous rhyming words had to be checked and transformed into the pronunciation current in the XVIth century when Early Modern English was still existent. This will be explained in a section below. It is also important to remind that the sonnets contain some 800 contractions and some 50 metrical adjustments which require the addition of an end of word syllable. After all these corrections, we obtained a sound map which clearly testifies to the intention of preserving a sound–sense harmony in the overall poetic scheme of the sonnets.

We may state as a general principle that the sound–sense harmony is respected whenever there is a full agreement between the sound grid and the mood associated with the meaning of the words. We assume then that there exists a sound–meaning correspondence by which different emotions or sentiments may be associated with each class. And of course, different results will be obtained by subtracting one class from the set, as we will comment below.

3.1.1. Periods and Themes in the Sonnets

The sonnets have been written in the short period that goes from 1592 to 1608, but we do not know precisely when. The majority of critics have divided them up into two main subperiods: a first one from 1592 to 1597 encompassing Sonnets from 1 to 126 and a second subperiod from 1598 to 1608 that includes Sonnets 127 to 154 (see Melchiori [9]). In addition, the sonnets have been traditionally subdivided into five main cycles or themes (Melchiori: Introduction): from 1 to 17, the reproduction sonnets, progeny, in which the poet spurs the young man to marry; from 18 to 51, immortality of poetry, the temptation of the friend by the lady, friend is guilty, and the absence of the loved one; from 52 to 96, poetry and memory, beauty and poetic rivalry; from 97 to 126, memory, the mistakes of the poet; and the last one from 127 to 152, the theme of the dark lady and unfaithfulness.

In Michael Schoenfeldt's Introduction to his edited book [27], we find a similar subdivision: Sonnets 1–126 are addressed to a beautiful young man, while Sonnets 127–152 are directed to a dark lady, and there are many other thematic and narrative sequences like 1–17 mentioned above (ibid. iii).

In the study of inversion made by Ingham and Ingham [28] on all of Shakespeare's plays, the authors reported three separate historical periods characterized by different frequencies in the use of subject inversion (VS) compared with canonical order (SV) on a total number of 951 clause structures:

1. A first period that goes from 1592 to 1597, where we have the majority of the cases of VS (214 over 421 total cases).
2. A second period that goes from 1598 to 1603, where the number of cases is reduced by half, but the proportion remains the same (109 over 213 total cases). A third period that goes from 1604 to 1608, where the proportion of cases is reverted (95 over 317 total cases) and VS cases are the minority.

The main themes of the sonnets are well-known: from 1 to 126 they are stories about a handsome young man, or rival poet; from 127 to 152 the sonnets concern a mysterious "dark" lady the poet and his companion love. The last two poems are adaptations from classical Greek poems. In the first sequence, the poet tries to convince his companion to marry and have children who will ensure immortality. Aside from love, the poem and poetry will "defeat" death. In the second sequence, both the poet and his companion have become obsessed with the dark lady, the lexicon used is sensual and the tone distressing. These themes are at their highest in the best sonnets indicated above. So, we would expect these sonnets to exhibit properties related to popularity that set them apart from the rest.

We decided to look into the "themes" matter more deeply and discovered that the immortality theme is in fact present through the lexical field constituted by the keyword DEATH. We thus collected all words related to this main keyword and they are the following ones, omitting all derivations, i.e., plurals for nouns, third person, past tense and gerundive forms for verbs:

BURY, DEAD, DEATH, DECEASE, DECAY, DIE, DISGRACE, DOOM, ENTOMBED, GRAVE, GRIEF, GRIEV ANCE, GRIEVE, SCYTHE, SEPULCHRE, TOMB, and WASTE

Which we connected to SAD, SADNESS, UNHAPPYNESS, and WRINKLE. We ended up by counting 64 sonnets containing this lexical field which can be safely regarded as the most frequent theme of all. We then looked for the opposite meanings, the ones related to LIFE, HAPPY, HAPPYNESS, PLEASURE, PLEASE, MEMORY, POSTERITY, and ETERNITY. In this case, 28 sonnets are the ones mentioning these themes. So, overall, we individuated 92 sonnets addressing emotionally related strong themes. When we combine the two contrasting themes, death/eternity, sadness/memory, we come up with the following 19 sonnets:

1, 3, 6, 8, 15, 16, 25, 28, 32, 43, 48, 55, 63, 77, 81, 92, 97, 128, 147

3.1.2. Measuring All Vowel Classes

We show in the Table 1. below general statistics of the distribution of stressed vowel sounds in rhyming words of all the sonnets. We included also diphthongs, considering the stressed portion as the relevant sound. The expected result is that the phonological class of high-back is the one less present in the sonnets, followed by high-front and low. Rhyming words with the middle stressed vowel are the ones with the highest frequency.

Table 1. Distribution of sounds of end-of-line rhyming words divided into four phonological classes.

Phon. Class	High-Front	Mid	Low	High-Back	Total
No. Class	119	159	142	111	531
StrVowDiph	493	861	587	314	2155

Here below are some examples of the classification of stressed vowels of rhyming words in the first three sonnets:

Sonnet 1: FRONT—increase, decease, spring, niggarding, be, thee;

BACK—fuel, cruel;
LOW—die, memory, eyes, lies;
MIDDLE—ornament, content;

Sonnet 2: BACK—use, excuse, old, cold;

MIDDLE—field, held, days, praise;
LOW—lies,eyes, mine, thine, brow, now;

Sonnet 3: HIGH—thee, see, husbandry, posterity, be, thee;

BACK—womb, tomb, viewest, renewest;
LOW—another, mother, prime, time.

In Table 2, we show the presence of the four classes in each sonnet, confirming our starting hypothesis about the intention to maintain a sound harmony in each sonnet: as can be easily gathered, 140 sonnets over 154 have rhymes with sounds belonging to more than two classes.

Table 2. Subdivision of the sonnets by number of classes.

No. Classes	4-Class	3-Class	2-Class	1-Class	Total
No. Sonnets	77	64	12	1	154

There is one sonnet with only one class and it is sonnet 146; then, there are 13 sonnets with 2 classes of sounds: 8, 9, 64, 71, 79, 81, 87, 90, 92, 96, 124, and 149. These sonnets contain rhyming pairs with low and middle sounds, except for three sonnets: sonnet 71 which contains high-back and middle sounds; sonnet 9 which contains high-front and low sounds; and sonnet 96 containing high-front and middle sounds. The themes developed in these sonnets fit perfectly into the rhyming sound class chosen. Let us consider sonnet VIII which is all devoted to music and string instruments which require more than one string to produce their sound, thus suggesting the need to find a companion and get married. Consider the line "the true concord of well tunèd sounds," where hints to the need that sounds should be "well" tuned. Sonnet 81 celebrates the poet and his verse which shall survive when death will come. Sonnet 92 is in fact pessimistic in the possibility that love will last "for the term of life" and no betrayal will ensue. As to sonnet 146, it is a mixture of two seemingly different themes: a criticism of extravagant display or rich clothing of wealth by writers of the time, or perhaps his mistress and trying to convince her to change her ways for eternal salvation. Some critics regard this as the most profoundly religious or meditative sonnet. But, the feeling of the lover renouncing something brings back his mistress and the feeling of being powerless against her chastity, so that religious life becomes a desirable aim. In this sense, death can also be depicted as desirable.

It is important to notice the overall strategy of choice of sound in relation to meaning, in the rhyming devices used, for instance, in sonnet 147 (all sonnets are taken from the online version made available at https://www.shakespeares-sonnets.com/ accessed on 6 July 2023):

My reason, the physician to my love,
Angry that his prescriptions are not kept,
Hath left me, and i desperate now approve
Desire is death, which physic did except.

The interesting fact in this case is that the appearance of a back high sound like |U| would match with the appearance of the saddest word, DEATH in the same stanza. In other words, the magistral use of rhyming sounds goes hand in hand with the themes and meaning developed in the sonnet.

Interesting to note how the rhyming sound evolves in the Sonnets taking sonnet 107 as an example: from SAD sounds (back and high), to MID and CLOSE to LOW and OPEN in the third stanza, to end with a repetition of MID sounds in the couplet:

Not mine own fears, nor the prophetic soul
Of the wide world dreaming on things to come,
Can yet the lease of my true love control,
Supposed as forfeit to a confin'd doom.
The mortal moon hath her eclipse endur'd,
And the sad augurs mock their own presage;
Incertainties now crown themselves assur'd,
And peace proclaims olives of endless age.
Now with the drops of this most balmy time,
My love looks fresh, and death to me subscribes,
Since, spite of him, I will live in this poor rime,
While he insults o'er dull and speechless tribes:
And thou in this shalt find thy monument,
When tyrants' crests and tombs of brass are spent.

In Sonnet 145, the overall feeling of sadness is transferred in the rhyming sounds: in the first stanza, the correct EME pronunciation requires |come| to be pronounced as |doom|, CUM/DUM a high-back sound which is then be repeated in the final couplet where "sav'd my life" appears. Here, important echoes of the |U| sound appear in the couplet with end-of-line words THREW and YOU.

. . .. . .
Straight in her heart did mercy come,
Chiding that tongue that ever sweet
Was us'd in giving gentle doom;
. . ..
From heaven to hell is flown away.
'I hate', from hate away she threw,
And sav'd my life, saying 'not you'.

We saw above the subdivision into classes; however, it does not tell us how the four phonological classes are distributed in the sonnets.

The resulting sound image coming from rhyme repetitions is eventually highlighted by the frequency of occurrence of same stressed vowel as shown in Table 3. In this table, we separated vowel sounds into three classes, high, middle, and low, to allow a better overall evaluation.

Table 3. Total count for vowel, final consonants and sonorant sounds organized into classes for all Shakespearean sonnets.

N.	Un/StressVow/ Con	Following Vowel/ Consonant	Freq Occ	High	Middle	Low	Consonant
1	ay	d, er, f, l, m, n, r, t, v, z	109			109	
2	ey	d, jh, k, l, m, n, s, t, v, z	81		81		
3	n_	d, iy, jh, s, t, z	80				80
4	r_	ay1, d, ey1, iy, iy1, k, n, ow, ow1, s, t, th, uw1, z	68				68
5	eh	d, jh, k, l, n, r, s, t, th	68		68		
6	ih	d, l, m, n, ng, r, s, t, v	51	51			
7	ao	d, l, n, ng, r, s, t, th, z	40		40		
8	iy	d, f, ih, k, l, m, n, p, s, t, v, z	45	45			
9	s	iy, st, t	38				38
10	uw	d, m, n, s, t, th, v, z	47	47			
11	ah	d, l, n, s, t, z	34			34	
12	ow	k, l, n, p, t, th, z	21	25			
13	t	er, ey1, iy, s, st	21				21
14	ah	d, k, m, n, ng	17			17	
15	aa	n, r, t	16			16	
16	ae	ch, d, k, ng, s, v	14			14	
17	d_z		13				13
18	er	ay1, d, iy, z	11		11		
		Total final sounds	778	168	200	190	220

Eventually, we come up with 61 more frequent heads with occurrences up to four and a total of 778 repeated vowel and consonant line-ending sounds. We now consider the remaining 288 rhyming pairs organized into "head" and "dependent", i.e., the preceding end of the line's rhyming word and the one in the corresponding alternate/adjacent end of line.

A direct consequence of the level of rhyming pair repetition rate is the sound image created in each sonnet. We assume that a high level of repetition will create a sort of echo from one sonnet to the next and a continuation effect, but it will also contribute a sense of familiarity. We decided to verify what would be the overall sound effect created by the total number of rhyming pairs analysed. Thanks to SPARSAR modules for phonetic transcription and poetic devices detection discussed elsewhere [29], we managed to recover all correct rhyming pairs and their phonetic forms. We report the results in the tables below.

The resulting sound image coming from rhyme repetitions is eventually highlighted by the frequency of occurrence of same stressed vowel as shown in the two tables below. We separated vowel sounds into three classes, high, middle, and low, to allow for an easy overall evaluation. If we consider all vowel sounds, there appears to be a highly balanced use of rhyming pairs with stressed low vowels being the more frequent. Not so if we consider diphthongs—we always consider the stressed vowel in both rising and falling diphthongs.

3.1.3. Distributing Vowel and Diphthong Classes into Thematic Periods

Win Table 4 below, we collected all stressed vowels and diphthongs for the five periods or phases into which the Sonnets collection can be divided up and found interesting variations: Period 1 has only 17 sonnets and 238 stressed rhyming words; Period 2 has 34 sonnets and 476 rhyming words; Period 3 has the majority, 45 sonnets and 630 rhyming words; Period 4 has 30 sonnets and 420 words; and Period 5 has the remaining 28 sonnets and 398 rhyming words.

Table 4. (**a**) Distribution of stressed rhyming vowels in five phases. (**b**) Weighted values of the distribution of stressed rhyming vowels in five Phases.

(a)

	Low	**Middle**	**High**	**Total**
Period 1	40	42	57	139
Period 2	105	68	102	275
Period 3	111	105	136	352
Period 4	59	79	122	260
Period 5	66	60	99	225
Totals	381	354	516	1251

(b)

	Low	**Middle**	**High**	**Total**
Period 1	2.3529	2.4706	3.3529	8.1765
Period 2	3.0882	2	3	8.0882
Period 3	2.4667	2.3334	3.0223	7.8223
Period 4	1.9667	2.6334	4.0667	8.6667
Period 5	2.3571	2.1429	3.5357	8.0357
Totals	30.529%	28.365%	41.106%	100%

In Table 4a we computed absolute values for each vowel class distributed in the five periods and what can be preliminarily noted is the high number of "high" vowels and the lower number of the two other classes. In Table 4b, we produced weighted measures in order to take into account differences in number of sonnets considered which, as a result, will produce a disparity in the total number of occurrences. Frequency values for each vowel class are now a ratio of the number of sonnets per phase, the same with total values.

In this case, we can easily see that high vowels are always the class which had the most occurrences and Periods 4 and 5 are the ones with the highest number—which, however, needs to be divided by two subclasses, front and back. The low vowel class is the one with higher percentage, and in Period 2, low vowels have their highest value when compared to the other Periods. The opposite takes place in Period 4, where High vowels are at their highest also compared with the other Periods and low vowels are at their lowest also compared to other Periods. We may note that, overall, the highest number of stressed vowels belongs to Phase 4, whereas the lowest number to Phase 3. Overall, the majority of stressed vowels belongs to the phonetic class of high vowels followed by low and then middle.

We must now consider diphthongs and verify whether the same picture applies. Diphthongs, as annotated in the CMU dictionary, do not contain any high stressed nuclear vowel, because the choice was to separate high vowels in all those cases. So, we are left with five diphthongs: two low, AW and AY; and three middle, EY, OW, and OY.

As can be easily gathered from absolute total values, middle diphthongs constitute by far the majority. In Table 5 below is their distribution in the five phases, and as we did in Table 4, we show at first absolute values and then in section (b) weighted values:

Table 5. (**a**) Distribution of stressed diphthongs in the sonnets divided in 5 phases. (**b**) Weighted valued of the distribution of stressed diphthongs in the sonnets in 5 phases.

(a)			
	Low	**Middle**	**Total**
Phase 1	50	46	96
Phase 2	78	103	181
Phase 3	112	154	266
Phase 4	81	72	153
Phase 5	65	85	150
Totals	386	460	846

(b)			
	Low	**Middle**	**Total**
Phase 1	2.9412	2.7059	5.6471
Phase 2	2.2941	3.0294	5.3235
Phase 3	2.4889	3.4223	5.9112
Phase 4	2.7	2.4	5.1
Phase 5	2.3214	3.357	5.3571
Totals	45.626%	54.373%	100%

Both Phases 1 and 4 show a decrease of middle vs. low diphthongs, while the remaining three phases behave in the opposite manner: more middle than low diphthongs. The total distribution indicates Phase 3 as the highest number of diphthongs and Phase 4 as the lowest, just the opposite of the previous distribution. General totals show a distribution of middle vs. low diphthongs which is strongly in favour of middle ones. This is just the opposite of what we found in previous counts, and in part then compensates with the lack of high diphthongs.

Eventually, in Table 6 the overall sound image is determined by a strong presence of middle sounds, followed by low sounds and eventually high sounds.

Table 6. Sound image of the sonnets.

	Low	**Middle**	**High**	**Total**
Vowels	381	354	567	1312
Diphthongs	386	460		854
Total	767	814	567	2166

3.2. Rhyming and Rhythm: The Sonnets and Poetic Devices

3.2.1. Contractions vs. Rhyme Schemes

Contractions are present in a great number in the sonnets. Computing them requires reconstructing their original complete corresponding word form in order to be able to match it to the lexicon or simply derive the lemma through morphological processing. This is essentially due to the fact that they are not predictable and must be analysed individually. Each type of contraction has a different manner to reconstruct the basis wordform. In order to understand and reconstruct it correctly, each contraction must go through recovering of the lemma. We have found 821 contractions in the collection, where 255 are cases of genitive's, and 167 are cases of past tense/participle 'd. The remaining cases are organised as follows:

- SUFFIXES attached at word end, for example (’s, ‘d, ’n, ‘st, ’t, (putt’st));
- PREFIXES elided at word beginning, for example (‘fore, ‘gainst, ’tis, ‘twixt, ‘greeing);
- INFIXES made by consonant elision inside the word (o’er, ne’er, bett’ring, whate’er, sland’ring, whoe’er, o’ercharg’d, ‘rous).

Now, consider a contracted word like “sland’ring”: as said before, at first the complete wordform must be reconstructed in order to use it for recovering the lemma and using the grammatical category for syntax and semantics. However, when computing the metrical structure of each line, the phonetic translation should be made on the contracted word, which does not exist in any dictionary neither in the form “slandring” nor in the form “sland-ring”. What we carry out is finding the phonetic transcription, if already existent, in the dictionaries, and then subtracting the phoneme that has been omitted, creating in this way a new word. This is okay until we come to the module where metrical counts are made on the basis of the number of syllables. But, here again, the phonetic form derived from the complete word is not easy to accommodate. There are two possible subdivisions of the phonetic form s_l_ae_n_d_r_ih_ng (in ARPAbet characters): syllable 1: s_l_ae_n_d_; syllable 2: r_ih_ng. Syllable 1 does not correspond to the subdivision for the complete word which would be s_l_ae_n_ | d_eh_ | r_ih_ng. Luckily, the syllable exists independently, but this only happens occasionally. In the majority of the cases, the new word form produces syllables which are inexistent and need to be created ad hoc.

3.2.2. Rhythm and Rhyme Violations

In poetry, in particular in the tradition of the sonnets in Elizabethan times, poetic devices play a fundamental role. Sonnets in their Elizabethan variety had a stringent architecture which required the reciter to organize the presentation according to logical structure in the stanza structure, on the one side introducing the main theme, expanding and developing the accompanying subthemes, exploring consequences, finding some remedies to solve the dilemma or save the protagonist. On the other side, the line-by-line structure required the reciter to respect the alternate rhyming patterns which were usually safeguarded by end-stopped lines. Thus, the audience expectations were strongly influenced by any variation related to rhyming and rhythm as represented by the sequence of breath groups and intonational groups. Whenever the rhyming pattern introduced a new unexpected pronunciation—not in other contexts—of a rhyming word, the audience was stunned: say a common word like *love* was pronounced to rhyme with *prove*. The same effect must have been produced with enjambments, whenever lines had to run-on because meaning required the syntactic structure to be reconstructed—as for instance, in lines ending in a head noun which had its prepositional-of modifier in the beginning of the following line. Breath groups and intonational groups had to be recast to suit the unexpected variation, but rhyming had to be preserved. We will explore these aspects of the sonnets thoroughly in this section.

In a previous paper [30], we discussed the problem of (pseudo) rhyme violations as it has been presented in the literature on Shakespeare. In particular, we referred to the presence of more than 100 apparent rhyme violations, that is, rhyming end-of-line words which according to current pronunciation do not allow the rhyming scheme of the stanza to succeed, but it did in the uncertain grammar of Early Modern English. For instance, in sonnet 1, we find two lines 2–4 with the stanza rhyme scheme ABAB, ending with the words die-memory. In this case, the second word *memory* should undergo a phonological transformation and be pronounced “memo’ry”(*memoray*) ending in a diphthong at the end and sounding like “die”/(*dye*). Linguist David Crystal has discussed and reported on this question in many papers and also on a website—http://originalpronunciation.com/ (accessed on 6 July 2023). He collects and comments rhyming words whose pronunciation is different from Modern RP English pronunciation, listing more than 130 such cases in the Sonnets. However, in our opinion, what is missing is a rigorous proposal to cope with the problem of rhyme violation, and the list of transformations contains many mistakes when compared with the full transcription of the sonnets published in [31]. The solution is lexical as we showed in a number of papers [29,30], i.e., variations should be listed in a specific lexicon of violations and the choice determined by an algorithm. Here below is an excerpt of the table, where we indicated the number of the sonnet, the line number, the rhyming word pair, their normal phonetic transcription using ARPAbet and in the last column the adjustment provided by the lexicon as shown in the example reported here below.

Example of lexical treatment of rhyme violation.

Sonnet No.	Line No.	Rhyme Violation	Arpabet Phoneme	Adjusted Phoneme
Sonnet 1	2–4	die-memory	d_ay-m_eh1_m_er_iy	iy→ay

Variants are computed by an algorithm that takes as input the rhyming word and its stressed vowel from the first line in a rhyming pair and compares it with the rhyming word and vowel of the alternate line. Here, as in the following pages, we will use the phonetic alphabet called ARPAbet which is the one of the phonetic dictionary made available by CMU for computational purposes. The phonetic annotation makes use of American English but includes all vowel phonemes of British English: it has 12 vowels, and two semiconsonants. The missing part regards diphthongs: there are eight diphthongs in the chart, but three of them—descending diphthongs—never appear in the CMU dictionary or are treated as a sequence of a semivowel and a stressed vowel—IA (for CLEAR, _ih_), EA (for DOWNSTAIR CAREFUL, eh_), and UA (for ACTUAL, w_ah). In case of failure, the lexicon of Elizabethan variants is searched. The same stressed vowel may undergo a number of different transformations, so it is the lexicon that drives the change, and it is impossible to establish phonological rules at feature level. Some words may be pronounced in two manners according to rhyming constraints; thus, it is the rhyming algorithm that will decide what to do with the lexicon of variants. The lexicon in our case has not been built manually but automatically, by taking into account all rhyming violations and transcribing the pair of words at line end on a file. The algorithm searches couples of words in alternate lines inside the same stanza and in sequence when in the couplet, and whenever the rhyme is not respected, it writes the pair in output. Take for instance the pair LOVE/PROVE, in that order in alternate lines within the same stanza: in this case, it is the first word that has to be pronounced like the second. The order is decided by the lexicon: LOVE is included in the lexicon with the rule for its transformation; PROVE is not. In some other cases, it is the second word that is modified by the first one, as in CRY/JOLLITY; again, the criterion for one vs. the other choice is determined by the lexicon.

In Table 7. below, we list the total number of violations we found subdividing them by five phases as we did before, in order to verify whether the conventions dictated by Early Modern English grammars of the time did eventually impose a standard in the last period, beginning with the XVIIth century. After Total, we indicate the total number of violations found followed by slash and the number of sonnets. The ratio gives a weighted number that can be used to compare different occurrences in the five phases. As can be noted, the highest number of violations are to be found in the first two phases. Then, there is a decrease from Phase II to Phase IV which is eventually followed by a slight increase in Phase V which, however, is lower than what we found in previous phases. The first two phases then have numbers well over the average: the decrease in the following phases testifies to a tendency in Shakespeare's work to fix pronunciation rules in the sonnets as more and more grammarians tried to document what constituted the rules for Early Modern English.

Table 7. Number of rhyme violations x five phases.

	Sonnets Interval	No. Rhyme Violations/ No. Sonnets	Ratio %
Phase I	1–17	22/17	1.2941
Phase II	18–51	40/34	1.1765
Phase III	52–96	34/45	0.7556
Phase IV	97–126	18/30	0.6
Phase V	127–154	23/28	0.8214
Total		137/154	0.8896

We call these (pseudo) rhyming violations because current reciters available on Youtube do not dare use the old pronunciation required and produce a rhyming violation by using Modern English pronunciation. One of these reciters is the famous actor John Gilgoud, who when reading Sonnet 66, correctly pronounces DESERT with its original meaning, but then in Sonnet 116 produces three violations when rhyming pairs required transformations that were clearly mandatory in Early Modern English, and they are |love| to be pronounced with the vowel of |remove| in lines 2/4, |come| to be pronounced with the vowel of |doom| in lines 10/12, and |loved| to be pronounced with the vowel of |proved| in the couplet. How do we know that these words should be pronounced in that manner and not in the opposite way—say |remove| as |love|, |doom| as |come| and |proved| as |loved|, as is being asserted by Ben Crystal son of David? There are three criteria that determine the way in which words should rhyme: the first one is the rhyming constraints which were so stringent at the time owing to the fact that poetry was only recited and not read on books. Okay, then, there are rhyming constraints but how do they work, in which direction? The direction is determined by two factors: the first one is determined by universal phonological principles, as for instance the one the governs phonological variations of vowel sounds—in the vowel shift of verbs or nouns due to morphological changes—which systematically changed "low" and "mid" features into "high" features and not vice versa [32]. The other factor is simply lexical: i.e., not all words will be subject to a transformation in that period. As a result, some words had double pronunciation. This was extensively documented in books and articles published at the time and written by famous poets like Ben Jonson and a great number of grammarians of the XVI and XVII century. All this information is made available by the famous historical phonologist Wilhelm Vietor of the XIX century in a book published at first in 1889 (2 (we use 1909 Vol 2. edition that can be freely visualized at: https://books.google.it/books?id=rhEQAwAAQBAJ&printsec=frontcover&hl=it&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false accessed on 6 July 2023), by the title "A Shakespeare Phonology" which we have adopted as our reference. Variants are then lexically determined. Some words involved in the transformation are listed below using ARPAbet as the phonetic alphabet in the excerpt taken from the lexicon. As can be easily noticed, variants are related also to stress position, but also to consonant sounds.

Lexicon 1.
shks(despised,d_ih2_s_p_ay1_s_t,ay1,ay1)
shks(dignity,d_ih2_g_n_ah_t_iy1,iy1,ay1).
shks(gravity,g_r_ae2_v_ah_t_iy1,iy1,ay1).
shks(history,hh_ih2_s_t_er_iy1,iy1,ay1).
shks(injuries,ih2_n_jh_er_iy1_z,iy1,iy1).
shks(jealousy,jh_eh2_l_ah_s_iy1,iy1,ay1).
shks(jollity,jh_aa2_l_t_iy1,iy1,ay1).
shks(majesty,m_ae2_jh_ah_s_t_iy1,iy1,ay1).
shks(memory,m_eh2_m_er_iy1,iy1,ay1).

shks(nothing,n_ah1_t_ih_ng,ah1,ow1).

It is now clear that variants need to interact with information coming from the rhyming algorithm that alone can judge whether the given word, usually at line end—but the word can also be elsewhere—has to undergo the transformation or not. The lexicon in our case has not been built manually but automatically, by taking into account all rhyming violations and transcribing the pair of words at line end on a file. The algorithm searches couples of words in alternate lines inside the same stanza and whenever the rhyme is not respected, it writes the pair in output. Take for instance the pair LOVE/PROVE, in that order, in alternate lines within the same stanza: in this case, it is the first word that has to be pronounced like the second. The order is decided by the lexicon: LOVE is included in the lexicon with the rule for its transformation, PROVE is not. In some other cases, it is the second word that is modified by the first one, as in CRY/JOLLITY, again the criterion for one vs. the other choice is determined by the lexicon. Thus, the system SPARSAR has a lexicon of possible transformations which are checked by an algorithm that whenever a violation is found, it is searched for the word to be modified and alters the phonetic description. In case both words of the rhyming pair are in the lexicon, the type of variation to be selected is determined by the overall sound map of the sonnet: Shakespeare produced a careful sound harmony in the choice of rhyming pairs including four or at least three sound classes.

Commenting on David Crystal's Point of View

Since the rhyming scheme is a fundamental issue for establishing sound harmony, the problem constituted by rhyming violations needs a deeper inspection. David Crystal makes available on his website the full phonetic transcription of the sonnets. However, as said above, these transcriptions contain many mistakes. There are two vague explanations Crystal finds to support his transcriptions in his OP (Old Pronunciation) and the first is a tautology: the "pronunciation system has changed since the 16th century": this is what he calls "a phonological perspective" (ibid.:298). In Section 2, entitled "Phonological rhymes", he writes

"Far more plausible is to take on board a phonological perspective, recognizing that the reason for rhymes fail to work today is because the pronunciation system has changed since the 16th century. . . . a novel and illuminating auditory experience, and introduced audiences to rhymes and puns which modern English totally obscures. The same happens when the sonnets are rendered in OP. In sonnet 154, the vowel of "warmed" echoes that of "disarmed", "remedy" echoes "by", the final syllable of "perpetual" is stressed and rhymes with "thrall", and the vowel of "prove" is short and rhymes with "love".

And further on (ibid:299):

"Ben Jonson. . . wrote an "English Grammar" in which he gives details about how letters should be pronounced. How do we know that "prove" rhymed with "love"? This is what he says about letter "O" in Chapter 4: "It naturally soundeth. . .. In the short time more flat, and akind to "u;" as "cosen", "dosen", "mother", "brother", "love", "prove" ". And in another section, he brings together "love, glove" and "move". This is not to deny, of course, that other pronunciations existed at the time. . .. "Love" may actually have had a long vowel in some regional dialects, as suggested by John Hard (a Devonshire man) in 1570 (and think of the lengthening we sometimes hear from singers today, who croon "I lurve you"). But the overriding impression from the orthoepists is that the vowel in "love" was short. It is an important point, because this word alone affects the reading of 19 sonnets. . .."

The second one is the need to respect puns (ibid. 298) which work in OP but not in modern English and, finally, the idiosyncratic spellings in the First Folio and Quarto and the description of contemporary orthoepists, who often give real detail about how pronunciations were in those days. There are no phonological rules, not even a uniform criterion that underlies the variations. The first reason was expressed as follows at the beginning of the paper: "The pronunciation of certain words has changed between Early

Modern English and today, so that these lines (referring to sonnet 154 lines) would have rhymed in Shakespeare's time". The list of pronunciation variations in the Supplementary Material of his paper [33] is messy and confusing but what is more important is that it also contains many mistakes, and we will comment on the first 10 items below.

First of all, the new rhyming transformation of "loved" is not mentioned in the Supplementary Material where according to Crystal "a complete" list should have appeared (ibid.:299). But the most disturbing fact is the recital performed by Ben Crystal (his son and actor in the Globe Theater), which is courageously made publicly available on Youtube (at https://www.youtube.com/watch?v=gPlpphT7n9s accessed on 6 July 2023). We are given a reading of Sonnet 116 which is illuminating of the type of OP Crystal is talking about (see time point 6:12 of total 10:21). The reading in fact does not start there but further on in the last stanza. The first contradictory assertion is just here, in the first stanza where lines B should rhyme and LOVE should be made to rhyme with REMOVE (as it is suggested in the Supplementary Material). The question is that in sonnet 154, the same rhyming pair in the same order LOVE—>REMOVE is transcribed with the opposite pronunciation. In the same paper, he asserts that "the vowel of PROVE is short and rhymes with LOVE" (ibid.:298) referring to the couplet of Sonnet 154 which we assume should be also applied to the B rhyming pair in sonnet 116 and not give us lav/rimav, but rather luv/rimuv. Here, an important additional series of alliteration would be fired if we adopt this pronunciation which in fact is the rule all over the Sonnets: TRUE would rhyme with LOVE and REMOVE/R. But also, further on as we will see, LOVE will rhyme with FOOL and DOOM.

On p.296,

Let me not to the marriage of true minds
Admit impediments, love is not love
Which alters when it alteration finds,
Or bends with the remover to remove.

The recital starts in third stanza, continuing with the couplet.

Love's not Time's fool, though rosy lips and cheeks
Within his bending sickle's compass come;
Love alters not with his brief hours and weeks,
But bears it out even to the edge of doom:
If this be error and upon me proved,
I never writ, nor no man ever loved.

In the Supplementary Material, we find another mistake or contradiction, where Crystal wrongly transcribes "doom" to rhyme with "come" (came/dam) rather than the opposite (cum/dum) and "loved" to rhyme with "proved" (pravd/lavd) which again should be the opposite, (pruvd/luvd). Here, as elsewhere, for instance in Sonnet 55, DOOM rhymes with ROOM in the correct order, ROOM/DOOM, and with the correct sound. Again, let us consider Crystal's wrongly reporting in the Supplementary Material the rhyming pair LOVE/APPROVE as rhyming in the opposite manner, i.e., LOVE is being pronounced as APPROVE which is just the contrary in the transcription; APPROVE is being pronounced as LOVE with a short open-mid back sounds. In Crystal's words,

"There are 19 instances in the sonnets where "love" is made to rhyme with "prove", "move", and their derived forms. And when we look at the whole sequence, we find a remarkable 142 rhyme pairs that clash (13% of all lines). Moreover, these are found in 96 sonnets. In sum: only a third of the sonnets rhyme perfectly in modern English. And in 18 instances, it is the final couples which fails to work, leaving a particularly bad taste in the ear."

This is how he explains the list of the Supplementary Material:

. . .a complete list is given in the Supplementary Material to this paper. The list indicates a rhyming pair where the first element is the one to be transformed because otherwise violating the rhyme. For instance MEMORY = DIE (1) must be interpreted as follows: pronounce "memory" with the same vowel of "die" in modern RP pronunciation to be found in sonnet 1.

It is important to note that the first element in most cases appears as the SECOND rhyming word in the pair, but in some other cases as the first word of the pair. But then, we find a long list of mistakes if we compare the expected pronunciation encoded in the Supplementary Material with the complete transcription of the sonnets made available by David Crystal in a pdf file in the same website, where results are turned upside down. For instance, LOVED = PROVED (116) has been implicitly turned into PROVED = LOVED, that is the transcription of the stressed vowel of "proved" is the same as the one of "loved" and not the opposite. More mistakes in the list can be found where words like TOMB and DOOM are wrongly listed in the opposite manner. In particular, DOOM is made to rhyme with the vowel of COME and not the oppositee; also, TOMB is made to rhyme with COME and DUMB reverting in both cases the order of the rhyming pair and of the transformation. The phonetic transcription file confirms the mistakes: in the related sonnets we find the same short mid-front vowel instead of a short U, dumb/tomb both in sonnet 83 and 101. In all of these cases, the head (the rhyming word of the first line) should be made to rhyme with the dependent (the rhyming word of the second line) as it happens in Sonnet 1 with MEMORY/DIE and in the great majority of cases. So, two elements must be taken into account: the order of the two words of the rhyming pair and then the commanding word, i.e., the word that governs the transformation. In the case of MEMORY/DIE, DIE is the head or the commanding word of the transformation, and comes first in the stanza, whereas MEMORY is the dependent word and comes as second line of the rhyming pair. We list below only the wrong cases and comment the type of mistake made, i.e., either as reverted order, the first element of the pair comes before and it should be read as second; reverted order, the first element is in fact the one deciding the type of vowel to be used; else the order is correct, but the pronunciation chosen is wrong. To comment on the wrong pronunciation required by the rhyme we sometimes use the pronunciation indicated by Vietor in his book, and the phonetic transcription of all the sonnets Crystal made in his pdf file.

There are more mistakes in the Supplementary Material, here are some of them:

anon/alone	75	-should be alone/anon (Vietor:70) both the order and the governor are wrong. It should be: pronounce ALONE as ANON with a short or long /o/
are/care	48	-the order should be care/are, but then the mistake is ARE transcribed like CARE [kEUR :r]
are/care	112, 147	-the order is correct but the transcription is wrong as before
are/compare	35	-the order should be compare/are, transcription correct
are/prepare	13	-the order should be prepare/are, transcription wrong: ARE is pronounced like PREPARE [pEUR :r]
are/rare	52	-order correct and in transcription ARE is like RARE [rEUR :r]—but it should be the opposite. RARE should sound like ARE, rare/are even though the line with RARE comes first.
beloved/removed	25	-order correct, but the transcription is wrong: remove is transcribed with the vowel of beloved
brood/blood	19	the order should be blood/brood: the transcription is also wrong BROOD is transcribed like BLOOD. see Vietor:87, whilst [u] in blood, flood, good, wood s. seems to be the usual Elizabethan sound.
dear/there	110	correct order but the pronunciation of DEAR is transcribed wrongly as [di:r] while the one of THERE is [thEUR :re]
doom/come	107,116,145	correct order but the pronunciation should be governed by DOOM, a short or long [u](Vietor:86): transcription of DOOM is instead with the vowel of COME

We solved the problem by creating a lexicon of phonetic transformations and an algorithm that looked at first for a match in the rhyming word pair positioned in alternate lines if in stanza, and in a sequence if in couplet. In case there was no match, the algorithm looks up the second word in the lexicon, and then the first word and chooses the one that is present. In case both are present in the lexicon, the decision is taken according to the position of the rhyming pair in the sonnet with respect to previous rhymes.

3.2.3. Rhyming Constraints and Rhyme Repetition Rate

If on the one side we have rhyme-apparent violations using the EME pronunciation to suit the rhyme scheme of the sonnet, on the other side, the Sonnets show a high "Repetition Rate" as computed on the basis of rhyming words alone. Due to the requirements imposed by the Elizabethan sonnet rhyme scheme, violations are very frequent, but they are not sufficient to allow the poet with the needed quantity of rhyming words. For this reason, it can be surmised that Shakespeare was obliged to use a noticeable amount of identical rhyming word pairs. The level of rhyming repetition is in fact fairly high in the sonnets, if compared with other poets of the same period, as can be gathered from the tables below. This topic has not gone unnoticed, as for instance [34], which indicates repetition of rhyming words as occurring in a limited number of consecutive adjacent sonnets, but does not give an overall picture of the phenomenon. In fact, as will be clear from the data reported below, the level of rhyming repetition is fairly high and reaches 65% of all rhyming pairs. In [34], we also find an attempt at listing all sonnets violating rhyme schemes which according to him amount to 25. However, as can be easily noticed in the list reported in the Supplementary Material, the number of sonnets violating the rhyme scheme is much higher than that.

To enumerate rhyming repetitions, we collected all end-of-line words with their phonetic transcription and joined them in alternate or sequential order as required by the sonnet rhyme scheme 1–3, 2–4, 5–7, 6–8, 9–11, 10–12, 13–14—apart from sonnet 126 with only 12 lines and a scheme in couplets aabbccddeeff, and sonnet 99 with 15 lines. Seven rhyming pairs for a total of 1078, i.e., 154 sonnets multiplied by 14 equal 2156 divided by two—less one 2155. In the tables reported as an Supplementary Material—the tables related to Rhyming Pair Repetition Rate have only been presented in Torino [30] at the conference and have not been published elsewhere—we only consider at first pairs with a frequency occurrence higher than 4, and we group together singular and plural of the same noun, and third person present indicative, d/n past with base form for verbs. We list pairs considering first occurrence as the "head" and following line as the "dependent". Rhyme may be sometimes determined by rules for rhyme violations as is the case with "eye". We include under the same heading all morphologically viable word forms as long as word stress is preserved in the same location, as said above, including derivations. We decided to separate highly frequent rhyming heads in order to verify whether less frequent ones really matter in the sense of modifying the overall sound image of the sonnets. For that purpose, we produce a first sound map below, limited to higher frequency rhyming pairs and only in a separate count we consider less frequent ones, i.e., hapax, trislegomena and dislegomena.

In many cases, the same pair is repeated in inverted order as for instance "thee/me" and "me/thee", "heart/part" and "part/heart", "love/prove" and "prove/love" but also "love/move" and "love/remove" and "approve/love" and "love/approve", "moan/gone" and "foregone/moan", "alone/gone" and "gone/alone", "counterfeit/set" and "unset/counterfeit", "worth/forth" and "forth/worth", "elsewhere/near" and "near/there", etc. "Thee" is made to rhyme with "me", but also with "melancholy", "posterity", "see". "Eye/s" are made to rhyme with almost identical monosyllabic sounding words like "die", "lie", "cries", "lies", "spies"; but also with "alchemy", "gravity", "history", "majesty", and "remedy", which require the conversion of the last syllable into a diphthong /ay/ preceded by the current consonant. Most of the rhyming pairs evoke a semantic or symbolic relation which is asserted or suggested by the context in the surrounding lines of the stanza that contain them. Just consider the pairs listed above where relations are almost explicit. However, as remarked by [34], rhyme repetition inside the same sonnet may have a different goal: linking lines at the beginning of the sonnet to lines at the end as is the case with sonnet 134 and the rhyme pair "free/me" which reappears in the couple in reversed order. Similar results are suggested by repetition of rhyme pair "heart/part" in sonnet 46.

In Table 8. we did the same count with two other famous poets writing poetry in the same century, Sir Philip Sydney and Edmund Spenser. We wanted to verify whether the high level of rhyming pairs repetition might also apply to other poets writing love sonnets. The results show some remarkable differences in the degree of repetitivity. In Table 9, repeated rhyming pairs are compared to unique ones or hapax rhyming pairs in three Elizabethan poets. Percentages reported are a ratio of all occurrences of rhyming pairs. In the first column, types are considered and Sydney overruns Shakespeare and Spenser. When we come to Token repeating rate—i.e., counting all occurrences of each type and summing them up, we still have the same picture. Eventually, unique or unrepeated rhyming pairs are higher in Spenser than in Shakespeare and Sydney.

Table 8. Rhyme repetition rates in three Elizabethan poets.

Author/ Quanti-Ties	Rhyme-Pair Repeat Types	Rhyme-Pair Repeat Token	Hapax or Unique Rhyme-Pairs
Shakespeare	18.02%	65.21%	34.79%
Spenser	17.84%	47.45%	53.55%
Sydney	22.37%	72.08%	27.02%

Table 9. Rhyme repetition word class-frequency distribution for Shakespeare's sonnets.

X Typ	FX Tok	Sum FX	Sum FX + X	% Sum FX + X
28	1	28	28	2.72
17	1	17	45	4.37
14	2	28	73	7.09
12	2	24	97	9.43
10	1	10	107	10.4
9	5	45	152	14.77
8	3	24	176	17.1
7	1	7	183	17.78
6	6	36	219	21.28
5	10	50	269	26.14
4	29	116	385	37.41
3	37	111	496	48.2
2	87	174	670	65.11
1	359	359	1029	100.0

Now, let us consider the distribution of rhyming words into the corpus of the sonnets. As to general frequency data, the Sonnets contain a number of tokens equal to 18,283 with 3085 types, so-called Vocabulary Richness that is used to measure the ability of a writer to use different words in a corpus, corresponds to 16.87%, a high value for that time when compared with other poets. Also, the number of Hapax and Rare Words (indicating the union of Hapax, Dis and TrisLegomena) corresponds to average values for other poets, respectively to 56%, the first type, and 79%, the second one. If we look at similar data for rhyming words, we see that Rare Words cover more than 65% of all as can be gathered from Table 10 below:

Table 10. Quantitative data for six appraisal classes for sonnets with highest contrast.

	Appr.Pos	Appr.Neg	Affct.Pos	Affct.Neg	Judgm.Pos	Judgm.Neg
Sum	56	25	53	77	32	122
Mean	2.533	1.133	2.4	3.466	1.444	5.466
St.Dev.	8.199	3.691	7.732	11.202	4.721	17.611

We report for each word frequency type in column 1—there is only one head word (*thee*) with frequency 28—the corresponding number of tokens in Table 9, followed by the sum of tokens, the incremental sum and the corresponding percentage with respect to total corpus. As can be noticed from the last column, where incremental percent of rhyme-pair words corpus coverage is reported, the total of rare words, i.e., type rhyme-pair with frequency of occurrence lower than 4, is 62.59%, a fairly low value if compared to the measure evaluated on simple type/token ratios. If we look at most important English poets, as documented in a previous paper , we can see that the average value for Rare Words is 77.88%. However, we are here dealing with rhyming words and the comparison may not be so relevant.

3.2.4. The Sound–Sense Harmony Visualized in Charts

As will appear clearly from the charts below, all the data show a contrasting behaviour which will be attested by correlation values. Where sentiment values increase, the corresponding values for vowels and consonants decrease. To allow better perusing of the trends we split the sonnets into separate tables according to whether their sentiment values are positive or negative. The first chart contains the eleven sonnets which received the highest positive sentiment values. All the charts are drawn from the tables of data derived from the analysis files in xml format, which will be made available as supplementary data (please see Figure 2).

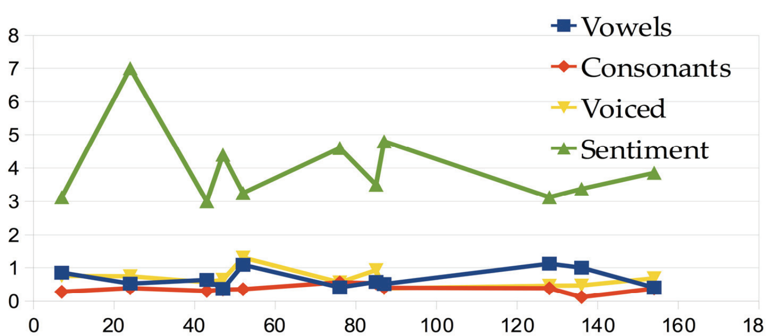


Figure 2. The eleven most positively marked sonnets: 7, 24, 43, 47, 52, 76, 85, 87, 128, 136, 154.

As can be easily noticed, all sound data seem to agree, showing a trend which is very close for the three variables. On the contrary, the sentiment variable has strong peaks and its values are set apart from the sound values. However, the interval of variability for sound variables does remain below or close to 1, thus indicating an opposite trend. In particular, consonants are all below 1, vowels oscillate in three cases, 52, 128, and 136, voiced in two cases, 52 and 85, in this case still below 1 but very close 93% in favour of unvoiced.

We interpret consistently contrasting values as a way to convey ironic, sarcastic and sometimes parodistic meaning. More on this interpretation below. Sonnet 136 is the one that is highly ambiguous and consequently ironic, celebrating the “Will” or simply “will”.

Sonnet 128 is all devoted to music and playing with a wooden instrument which is the target of the ironic vein and the double meaning of words like "tickle". Finally, sonnet 52 is the celebration of the beloved as a "chest" where the rich keep their treasure, and which must be enjoyed "seldom". Sonnet 85 is a celebration of silent thought, and for this theme, it is filled with consonants which are continuants |h,f,th| and are unvoiced, but many words are marked by a sonorant syllable, thus voiced.

We now separate 16 sonnets which have sentiment equal to 1 or slightly lower than 1 but always higher than 92% in favour of positively marked. They are the following: 22, 33, 51, 60, 64, 73, 94, 97, 101, 102, 109, 118, 123, 131, 141, and 150 (please see Figure 3).

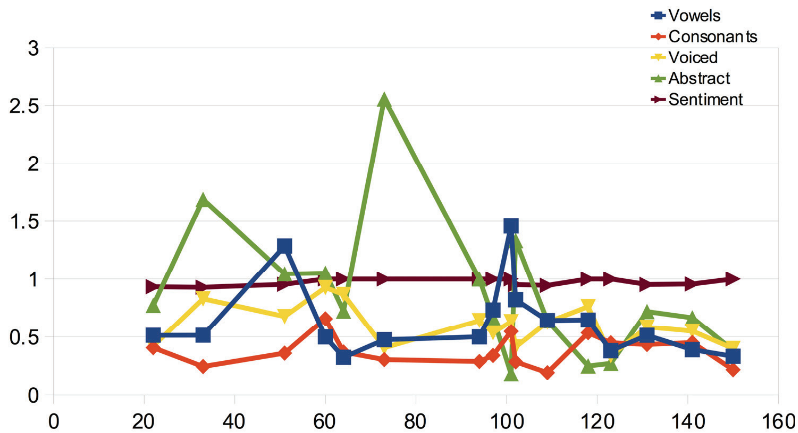


Figure 3. Chart of the 16 borderline sonnets positively marked for sentiment.

In this chart we added the ratio for Abstract/Concrete, which shows a peak for sonnet 73. As the chart clearly shows, the line for Sentiment borders 1, as to the remaining variables, Vowels is the one oscillating most after Abstract. Voiced and Consonants are fairly always aligned apart from sonnet 33 and 102. In both sonnets, the number of "Obstruents" (|b,d,p,t,k,g|) is very low and real consonants are substituted by "Continuants" (|s,sh,th,f,v,h|) both voiced and unvoiced. In the following analysis, for this reason, I will only consider Voicing as the relevant variable for consonants and this will show better agreement in the overall data. Now, we show charts for all negatively marked sonnets using only three variables, starting from Figure 4 below.

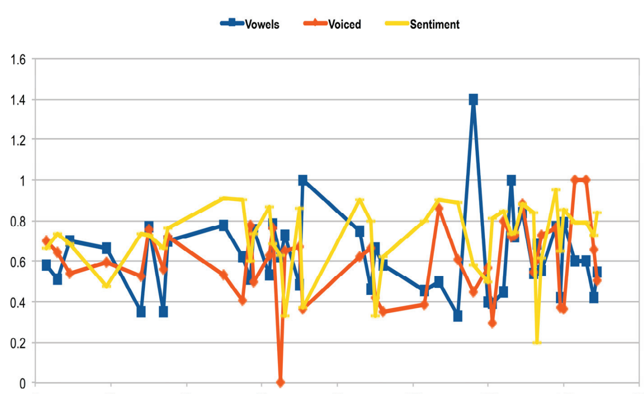


Figure 4. Chart of the 42 negatively marked sonnets: 3, 8, 9, 19, 28, 30, 34, 35, 50, 55, 57, 58, 60, 62, 63, 65, 66, 71, 86, 89, 92, 103, 107, 112, 116, 120, 121, 124, 126, 127, 129, 132, 133, 134, 138, 139, 140, 143, 146, 148, 149.

As can be easily seen, the Sentiment variable is always below 1 but the two remaining variables oscillate up and down, the vowel one oscillating most in the upper portion of the chart, and the voiced one in the lowe portion. In this case, the contrast is even stronger and correlations show a negative trend between Vowels and Sentiment: the one has a decreasing trend while the other has it increasing, apart from a few exceptions, sonnets 30, 35 and 127, which have almost identical values for the three variables. The other correlation between Voicing and Sentiment is positive but very weak: 0.1769.

Correlation between Vowel and Sentiment is positive but very weak; correlation between the Voicing parameter and Sentiment is again negative and very weak at −0.0065037. Thus, results for the 42 sonnets negatively marked by sentiment show that we have negative correlation between vowels and voicing, and vowels and sentiment, but positive correlation between voicing and sentiment. So, it is just the opposite of what we obtain with positively marked sonnets. And finally, in Figure 5. we show the eleven most positively marked sonnets show the same contrasting results.

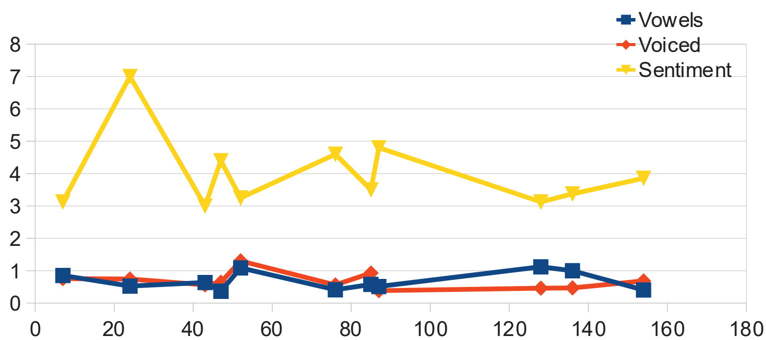


Figure 5. The eleven most positively marked sonnets show the same slightly positive correlation for Vowels–Voicing but very strong negative correlation between Vowels–Sentiment and slightly negative for Voicing/Sentiment at −0.11423482—colours in this case have no meaning.

As to the remaining 85 sonnets positively marked for sentiment, they all have very weak but positive correlations between sound and sense, i.e., below 0.1, respectively, 0.0387 for vowels, 0.05 for consonants, and 0.091 for voicing. The conclusion we may draw is that the sound–sense harmony in Shakespeare's sonnets is represented by a weak extended harmony for those positively marked for sentiment but a strong disharmony for those sonnets negatively marked for sentiment: in particular in all the sonnets we have an inverse correlation, between the two most important variables, Voicing (whether a consonant is a real Obstruent or not) and Sentiment. As said above, voicing includes real obstruents and unvoiced continuants: | p,t,k,s,sh,f,th |. When the pair Voicing/Sentiment assumes a positive correlation value, the other pair Vowel/Sentiment shows the opposite and is negative. Then, we saw the exceptions, in those sonnets which are most positively marked for sentiment, the correlations between Vowel and Sentiment are positive but the correlation between Voicing and Sentiment is negative. Sonnets negatively marked for sentiment have a positive correlation between Voicing and Sentiment, but a negative correlation between Vowel and Sentiment. In other words, the behaviour is just reversed: when meaning is positively marked the sound harmony verges towards a negative feeling. On the contrary, when the meaning is negatively marked the sound harmony verges, bends towards a positive sound harmony. I assume what Shakespeare intended to produce in this way was a cognitive picture of ironic poetic creation.

3.2.5. From Sentiment to Deep Semantic and Pragmatic Analysis with ATF

The final part of the analysis takes us deep into the hidden meaning that the sonnets communicate, i.e., irony. To carry that out, we need to substitute sentiment analysis with a much more semantically consistent framework that could allow us to enter the more complex system of relational meanings that are governed by pragmatics. In this case, neither a word-by-word analysis or propositional-level analysis would be sufficient. We need to capture sequences of words which may have a non-literal meaning and associate appropriate labels: this is what the Appraisal Theory Framework can be useful for.

We have devised a sequence of steps in order to confirm experimentally our intuitions. The preliminary results obtained using sentiment analysis cannot be regarded as fully satisfactory for the simple reason that both the lexical and the semantic approach based on predicate-argument structures are unable to cope with the use of non-literal language. Poetic language is not only ambiguous but it contains metaphors which require abandoning the usual compositional operations for a more complex restructuring sequence of steps.

This has been carefully taken into account when annotating the sonnets by means of Appraisal Theory Framework (henceforth ATF). In our approach, we have followed the so-called incongruity presumption or incongruity-resolution presumption. Theories connected to the incongruity presumption are mostly cognitive-based and related to concepts highlighted, for instance, in [35]. The focus of theorization under this presumption is that in humorous texts, or broadly speaking in any humorous situation, there is an opposition between two alternative dimensions. As a result, we have been looking for contrast in our study of the sonnets, produced by the contents of manual classification. Thus, we have used the Appraisal Framework Theory [36]—which can be regarded as the most scientifically viable linguistic theory for this task, as has already been conducted in the past by other authors (see [12,37] but also [38]), showing its usefulness for detecting irony, considering its ambiguity and its elusive traits.

Thus, we proceeded like this: we produced a gold standard containing strong hints in its classification in terms of humour, by collecting most important literary critics' reviews of the 154 sonnets (the gold standard will be made available as Supplementary Material). To show how the classification has been organized we report here below two examples:

- SONNET 8

SEQUENCE: 1–17 Procreation MAIN THEME: One against many ACTION: Young man urged to reproduce METAPHOR: Through progeny the young man will not be alone NEG.EVAL: The young man seems to be disinterested POS.EVAL: Young man positive aesthetic evaluation CONTRAST: Between one and many

- SONNET 21

SEQUENCE: 18–86 Time and Immortality MAIN THEME: Love ACTION: The Young man must understand the sincerity of poet's love METAPHOR: True love is sincere NEG.EVAL: The young man listens the false praise made by others POS.EVAL: Young Man positive aesthetic evaluation CONTRAST: Between true and fictitious love.

As can be seen, the classification is organized using seven different linguistic components: we indicate SEQUENCE for the thematic sequence into which the sonnet is included; this is followed by MAIN THEME which is the theme the sonnet deals with; ACTION reports the possible action proposed by the poet to the protagonist of the poem; METAPHOR is the main metaphor introduced in the poem sometimes using words from a specialized domain; NEG.EVAL and POS.EVAL stand for Negative Evaluation and Positive Evaluation contained in the poem in relation to the theme and the protagonist(s); finally, CONTRAST is the key to signal presence of opposing concrete or abstract concepts used by Shakespeare to reinforce the arguments purported in the poem. Not all the sonnets were amenable to a pragmatic/linguistic classification. We ended up with 98 sonnets classified over 154, corresponding to a percentage of 63.64%, the rest have been classified as Blank. Many sonnets have received more than one possible pragmatic category. This is due to the difficulty in choosing one category over another. In particular, it has been particularly hard to distinguish irony from satire, and irony from sarcasm. Overall, we ended up with 54 sonnets receiving a double marking over 98. This was also one of the reasons to use ATF: often literary critics were simply hinting at "irony" or "satire", but the annotation gave us a precise measure of the level of contrast present in each of the sonnets regarded generically as "ironic".

The annotation has been organized around only one category, *Attitude*, and its direct subcategories, in order to keep the annotation at a more workable level, and to optimize time and space in the XML annotation. *Attitude* includes different options for expressing positive or negative evaluation, and expresses the author's feelings. The main category is divided into three primary fields with their relative positive or negative polarity, namely:

- *Affect* is every emotional evaluation of things, processes or states of affairs, (e.g., like/dislike); it describes proper feelings and any emotional reaction within the text aimed towards human behaviour/process and phenomena.
- *Judgement* is any kind of ethical evaluation of human behaviour, (e.g., good/bad), and considers the ethical evaluation on people and their behaviours.
- *Appreciation* is every aesthetic or functional evaluation of things, processes and state of affairs (e.g., beautiful/ugly; useful/useless), and represent any aesthetic evaluation of things, both man-made and natural phenomena.

Eventually, we ended up with six different classes: *Affect Positive*, *Affect Negative*, *Judgement Positive*, *Judgement Negative*, *Appreciation Positive*, and *Appreciation Negative*. Overall, in the annotation, there is a total majority of positive polarities with a ratio of 0.511, in comparison to negative annotations with a ratio of 0.488. In short, the whole of the positive poles is 607, and the totality of the negative poles is 579 for a total number of 1186 annotations. *Judgement* is the more interesting category because it allows social moral sanction, which is then split into two subfields, *Social Esteem* and *Social Sanction*—which, however, we decided not to mark. In particular, whereas the positive polarity annotation of *Judgement* extends to *Admiration* and *Praise*, the negative polarity annotation deals with *Criticism* and *Condemnation* or *Social Esteem* and *Social Sanction* (see [38], p. 52). Here below is the list of 77 sonnets manually classified with ATF over 98 matching critics' evaluation.

List of 77 successfully classified sonnets matching critics' evaluation.

1 2 4 5 6 10 12 14 17 18 19 20 21 27 30 32 33 34 35 37 41 42 47 48 50 56 57 61 65 67 68 69 71 72 74 75 77 78 79 81 82 84 87 92 95 97 98 101 102 104 106 108 109 111 113 114 115 116 123 125 126 127 129 134 136 137 139 142 144 145 146 149 151 152 153 154

As a first result, we may notice a very high convergence existing between critics' opinions and the output of manual annotation by *Appraisal* classes: 77 over 98 corresponds to a percentage of 78%. As to the sonnets' structure, *Judgement* is found mainly in the final couplet of the sonnets (for more details, see [3]). As to interpretation criteria, we assumed that the sonnets with the highest contrast could belong to the category of **Sarcasm**. The reason for this is justified by the fact that a high level of *Negative Judgements* accompanied by *Positive Appreciations* or *Affect* is by itself interpretable as the intention to provoke a sarcastic mood. As a final result, there are 44 sonnets that present the highest contrast and are specifically classified according to the six classes above. There is also a group that contains ambiguity sonnets which have been classified with a double class, mainly by **Irony** and **Sarcasm**. As a first remark, in all these sonnets, negative polarity is higher than positive polarity with the exception of sonnet 106. In other words, if we consider this annotation as the one containing the highest levels of *Judgement*, we come to the conclusion that a possible **Sarcasm** reading is mostly associated with presence of *Judgement Negative* and in general with high *Negative* polarity annotations. In Figure 6 below, we show the 44 sonnets classified with Sarcasm.

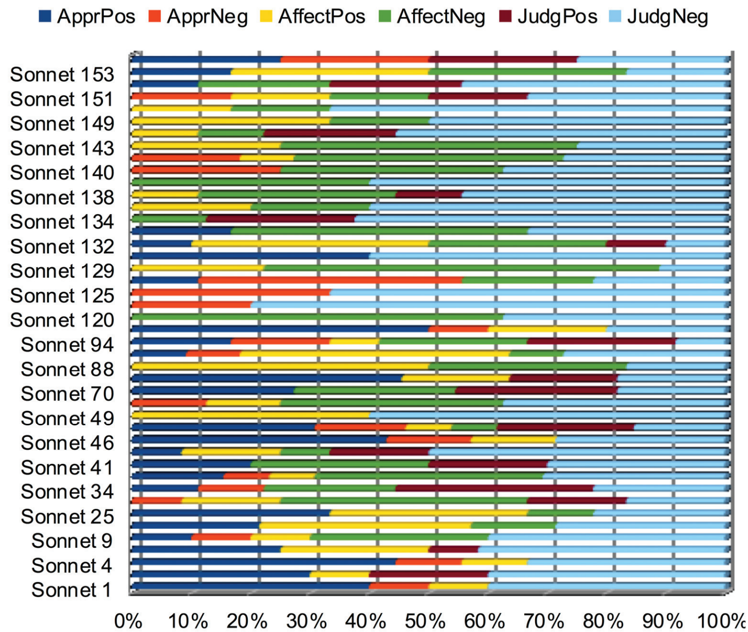


Figure 6. The 44 sonnets classified with Sarcasm with the highest level of Judgements—colours in this case have no meaning.

We associated different colours to make the subdivision into the six classes visually clear. It is possible to note the high number of *Judgements* both *Negative* (in orange) and *Positive* (in pale blue): in case *Judgement Positive* is missing, it is substituted by *Affect Positive* (pale green) or by *Appreciation Positive* (blue). This applies to all 44 sonnets apart from sonnets 120 and 121 where *Judgement Negative* is associated with *Affect Negative* and to *Appreciation Negative*. In other words, if we consider this annotation as the one containing the highest levels of *Judgement*, we come to the conclusion that possible **Sarcasm** reading is mostly associated with presence of *Judgement Negative* and, in general, with high *Negative* polarity annotations. As a first result, we may notice a very high correlation existing between critics' opinions as classified by us with the label highest contrast and the output of manual annotation by *Appraisal* classes.

In Figure 7 we show the group of 50 sonnets classified, mainly or exclusively, with **Irony** and check their compliance with *Appraisal* classes.

As can be easily noticed, the presence of *Judgement Negative* is much lower than in the previous diagram for **Sarcasm**. In fact, in only half of them—25—have annotations for that class; the remaining half introduces two other negative classes: mainly *Affect Negative*, but also *Appreciation Negative*. As to the main *Positive* class, we can see that it is no longer *Judgement Positive*, but *Affect Positive* which is present in 33 sonnets (please see Table 11).

Table 11. Quantitative data for six appraisal classes for sonnets with lowest contrast.

	Appr.Neg	Appr.Pos	Affct.Pos	Affct.Neg	Judgm.Pos	Judgm.Neg
Sum	139	65	64	81	59	37
Mean	5.346	2.5	2.461	3.115	2.269	1.423
St.Dev.	18.82	8.843	8.707	11.009	8.029	5.047

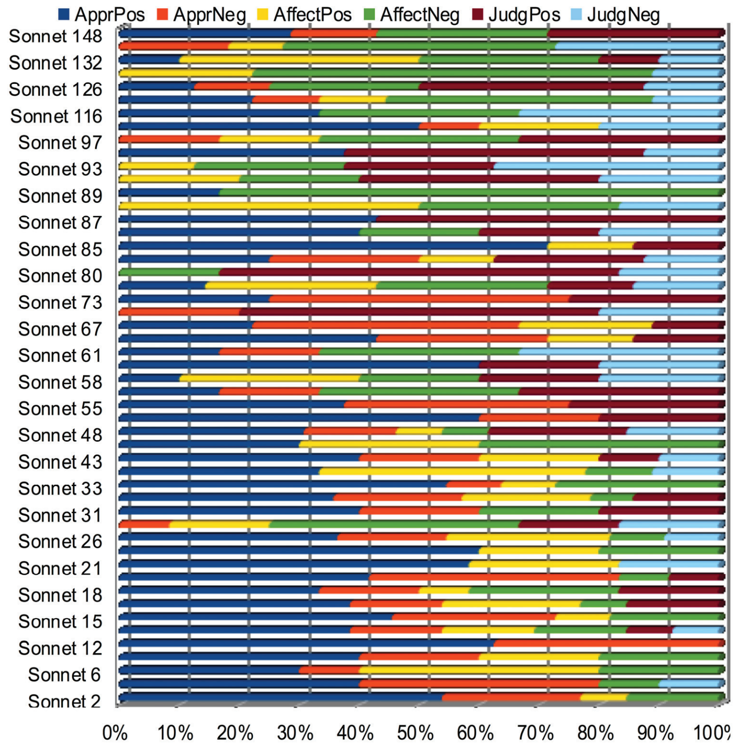


Figure 7. The 50 sonnets classified with Irony, with a lower level of Judgement Negative but higher Affect Negative.

In other words, we can now consider that **Sarcasm** is characterized by a majority of negative evaluations 146/224 while **Irony** is characterized by a majority of *Positive* evaluations 262/183 and that the values are sparse and unequally distributed.

The final figure, Figure 8, concerns the number of sonnets with blank or neutral evaluation by critics which amount to 60. As a rule, this group of sonnets should look different from the two groups we already analysed.

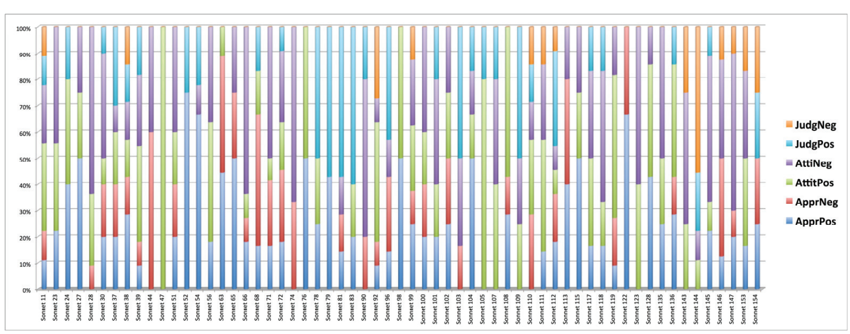


Figure 8. The 60 Sonnets classified by critics as neutral.

As expected, this figure looks fairly different from the previous two. The prevailing colour is pale blue, i.e., *Judgement Positive*; orange, i.e., *Appraisal Negative*, is only occasionally present; and green is perhaps the second prominent colour, i.e., *Affect Positive*. In order to know how much the difference is, we can judge it from the quantities shown in Table 12 below.

Table 12. Quantitative data for six appraisal classes for sonnets with no contrast.

	Appr.Pos	Appr.Neg	Affct.Pos	Affct.Neg	Judgm.Pos	Judgm.Neg
Sum	88	59	89	109	49	8
Mean	3.034	2.034	3.068	3.758	1.689	0.275
St.Dev.	1.268	7.638	11.482	14.052	6.368	1.079

3.2.6. Matching ATF Classes with the Algorithm for Sound–Sense Harmony (ASSH)

The experiment with ATF classes matching critics' evaluation has been fairly successful, but how do these classes gauge with the Sound–Sense harmony? In order to check this, we transferred the data related to vowels and consonants and matched them with ratios of the three main ATF categories: *Appreciation Positive/Negative*, *Affect Positive/Negative*, and *Judgement Positive/Negative*. As in previous computation, all data below 1 will be interpreted as a case of superior *Negative Polarity* and the opposite when data are above 1. To allow a better view of the overall data, we split them into sonnets with contrast to the first group that we show in Figure 9, and sonnets with no contrast to the second group, that we show in Figure 10. This time, however, we used our classification and abandoned the critics' one.

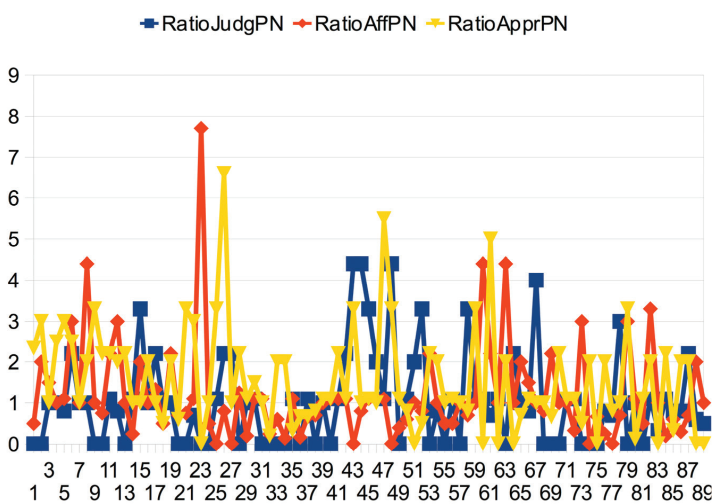


Figure 9. Distribution of 89 sonnets manually classified by ATF with no contrast.

The data in Figure 10 show the distribution of the Sound–sense variable for the three parameters: we did not introduce variables for vowels and voicing which are, however, present in the same table and allow us to evaluate the correlation between ATF and sound, which as can be seen below is negative for both *Judgement* and *Affect*:

1. Correlation between Vowels and *Judgement*: -0.1254;
2. Correlation between Voicing and *Judgement*: -0.1468;
3. Correlation between Vowels and *Affect*: -0.08859;
4. Correlation between Voicing and *Affect*: -0.01346;
5. Correlation between Judgement and *Affect*: -0.1376;
6. Correlation between *Affect* and *Appraisal*: -0.0351.

Correlations of sound data with Appraisal are on the contrary both positive. If we consider now the remaining 65 sonnets which have been classified by ATF with contrast, we obtain a different picture. In this case, we have separated each class and projected them with sound data, Vowels and Voicing in the following three diagrams.

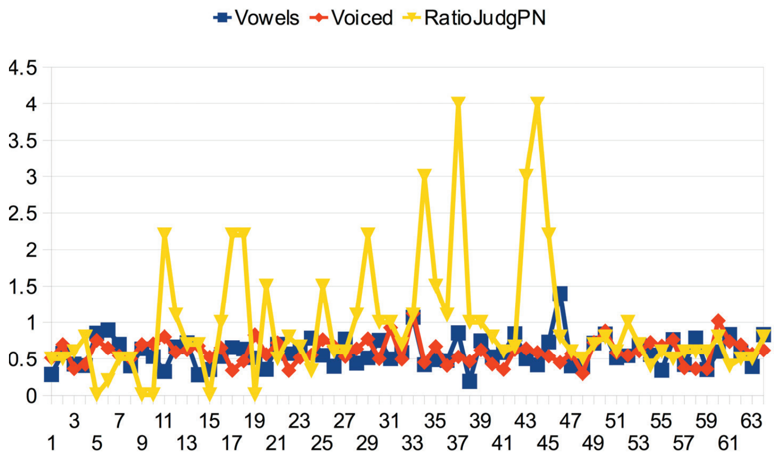


Figure 10. Distribution of 65 sonnets classified as Judgements with contrast and their sound data.

All correlation measures with *Judgements* are negative:

Correlation between Vowels and *Judgements*: −0.0594;
Correlation between Voicing and *Judgements*: −0.0677;
Correlation between *Judgement* and *Affect*: −0.0439;
Correlation between *Judgement* and *Appraisal*: −0.0522.

In Figure 11 below we use again sound data and the second parameter Affect:

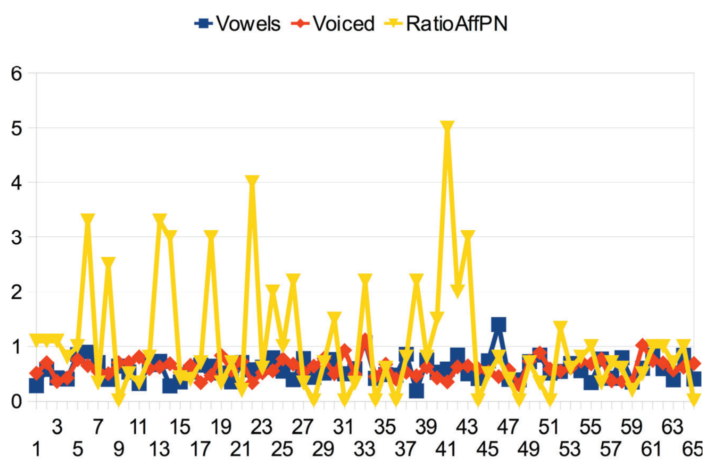


Figure 11. Distribution of 65 sonnets classified by ATF as Affect with contrast and their sound data.

Correlation data for *Affect* are only partly negative:
Correlation between Vowels and *Affect*: 0.09;
Correlation between Voicing and *Affect*: −0.1435;
Correlation between *Affect* and *Appraisal*: 0.2594.

Finally in Figure 12 we project sound data with Appraisal parameters:

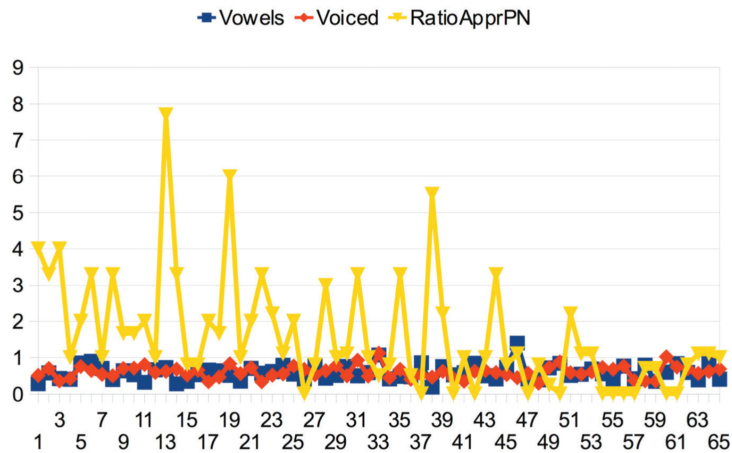


Figure 12. Distribution of 65 sonnets classified by ATF as Appraisal with contrast and their sound data.

Eventually, the correlations for *Appraisal* are also both negative:

Correlation between Vowels and *Appraisal*: −0.2068;
Correlation between Voicing and *Appraisal*: −0.0103.

Now the only positive correlations are the ones shown by *Affect* with Vowels and with *Appraisal*; the remaining correlations are all negative. The subdivision operated now using our manual classification with ATF seems more consistent than the one made before using the critics' evaluation. As a first comment, these data confirm our previous evaluation made on the basis of sentiment analysis, i.e., the sonnets are mainly disharmonic due to Shakespeare's intention to produce ironic effects on the audience. Here below is the list of the 89 sonnets classified by our manual ATF labeling as having no contrast:

List of 89 sonnets manually classified by ATF as having no contrast.

2 6 11 13 14 16 17 20 22 23 24 26 27 30 37 38 39 40 43 44 45 46 47 51 54 56 59 62 63 64 65 66 68 71 72 74 75 76 77 78 79 81 83 84 85 87 89 90 91 96 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 117 118 119 122 123 124 125 126 128 130 135 136 141 145 146 147 148 149 150

List of 65 sonnets manually classified by ATF as having contrast.

1 3 4 5 7 8 9 10 12 15 18 19 21 25 28 29 31 32 33 34 35 36 41 42 48 49 50 53 57 58 60 61 67 69 70 73 80 82 86 88 92 93 94 95 97 116 120 121 127 129 131 132 133 134 137 138 139 140 142 143 144 151 152 153 154

Comparing the "contrast" criterion with the sentiment-based classification is not possible; however, the "contrast" group of sonnets is included in majority by the "negatively" marked sonnets, with the exception of 16 sonnets which are the following ones:

30 55 60 62 63 66 71 103 107 112 124 126 127 146 148 149

What these sonnets have in common is an identical number of *Appraisal Positive/Negative* feature (28), a high number of *Affect Negative* feature (38) vs. *Positive* ones (11), and the relatively lowest number of *Judgement Negative* features (10) vs. *Positive* ones (18). In other words, by decomposing *Negative Polarity* items into three classes, we managed to show the weakness of sentiment analysis, where Negativity is a cover-all class. Overall, the sonnets contain a majority of positively or neutrally evaluated sonnets in both sentiment and appraisal analysis, and a minority of negatively evaluated sonnets: the SSH is, however, mostly a disharmony.

3.3. Sound and Harmony in the Poetry of Francis Webb

In this section, I will presents the results obtained from the analysis of the poetry by Francis Webb, who is regarded by many critics among the best English poets of the last century—and differently from Shakespeare, he never uses ironic attitudes. All the poems I will be using are taken from the Collected Poems edited by Toby Davidson [39].

I will introduce a type of graphical maps highlighting differences using colours associated with sound and sense (see [11]). The representation of the proposed harmony between sense and sound will be cast on the graphical space as follows:

- Class A:

  Negatively harmonic poems, mainly negatively marked poems on the left. Either the sounds or the sentiment are in majority negative, or both the sounds and the sentiment are negative.

- Class C:

  Positively harmonic poems, mainly positively marked poems on the right. Either the sounds or the sentiment are in majority positive, or both the sounds and the sentiment are positive.

- Class B:

  Disharmonic ones in the middle. The sounds and the sentiment have opposite values and either one or the other have values below a given threshold.

In addition to the evaluation of positive/negative values, we consider the two parameters we already computed related to Metrical Length and Rhyming Scheme that we add together and use for its 10% added value to compensate for poetic relevant features. On the basis of poetic devices analyzed by *SPARSAR*, a list of 14 poems is considered as deviant, and they are the following: *A Sunrise, The Gunner, The Explorer's Wife, For My Grandfather, Idyll, Middle Harbour, Politician, To a Poet, The Captain of the Oberon, Palace of Dreams, The Room, Vancouver by Rail, Henry Lawson,* and *Achilles and the Woman*. In Figure 13 we show the first map of sense–sound evaluation where the split of the "deviants" poems appears clearly:

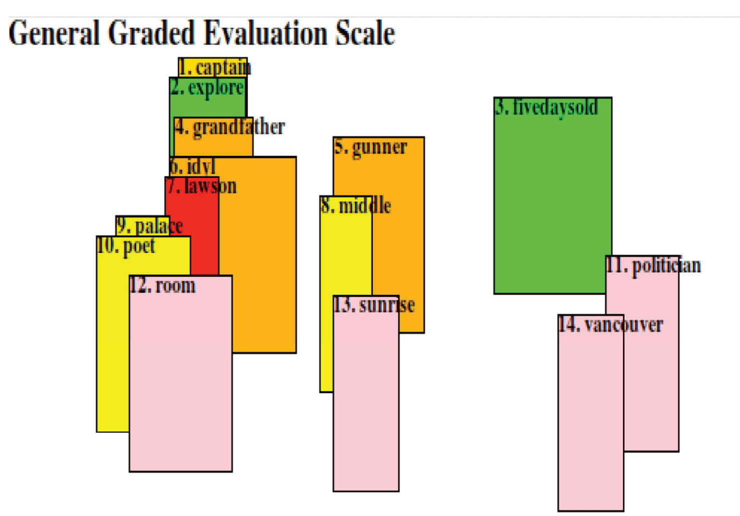


Figure 13. Poems considered as deviants evaluated for their degree of sense/sound harmony.

The poem that best represents balanced positive values is *Five Days Old* and this may be deduced by the presence of the largest box positioned on the right hand side. Overall, the figure shows which poems achieved harmonic values and positions positives on the right and negative on the left sides, and then in the middle disharmonic ones. As clearly appears, "Five Days Old", "Politician" and "Vancouver by Rail" are the three poems computed as endowed with positive harmony, while the remaining poems are either characterized as strongly negative—"Poet", "Palace of Dreams" and "The Room"—or just negative, "The Captain of the Oberon", The Explorer's Wife", "For My Grandfather", "Idyl", and "Henry Lawson". Finally, the last three poems positioned in the centre left are disharmonic, "The Gunner", "Middle Harbour", and "A Sunrise", where disharmonic means that the parameters of sounds are in opposition to those of sense. Slight variations in the position are determined by the contribution of parameters computed from poetic devices as said above. Disharmony as will be discussed further on might be regarded as a choice by the poet with the intended aim to reconcile the opposites in the poem.

The choice of these 14 poems includes poetry written at the beginning of the career, i.e., included in the *Early Poems*—A Sunrise, Palace of Dreams, To a Poet, Idyll, Middle Harbour, and Vancouver by Rail—two poems from *A Drum for Ben Boyd*; Politician, The Captain of the Oberon—five poems from *Leichhardt in Theatre*—The Room, The Explorer's Wife, For My Grandfather, The Gunner, Henry Lawson—and finally, one poem from *Birthday*, Achilles and the Woman, and one poem from *Socrates*, Five Days Old. In what follows, at first, I will show small groups of poems taken from different periods in Webb's poetic production and discuss them separately, rather than conflating them all in a single image. In fact, at the end of this section, I will show a bigger picture where I analysed 87 poems together, resulting in two big figures. Now, I will back to the second experiment where I collected and analyzed the following poems,

Early Poems—Idyll, The Mountains, Vancouver by Rail, A Tip for Saturday, This Runner
Leichhardt in Theatre—Melville at Woods Hole, For Ethel, On First Hearing a Cuckoo
Poems 1950–52—The Runner, Nuriootpa
Birthday—Ball's Head Again, The Song of a New Australian
Socrates—The Yellowhammer
The Ghost of the Cock—Ward Two and the Kookaburra
Unfinished Works—Episode, Untitled

In Figure 14 I show their distribution in the three separate rows:

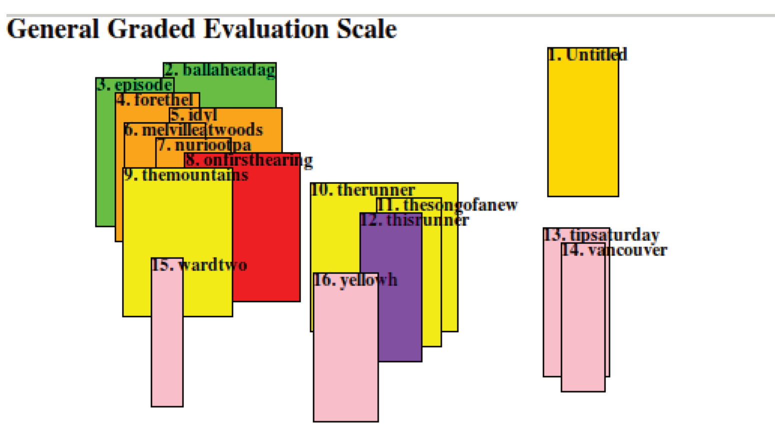


Figure 14. Sixteen poems from different periods of Webb's poetic production computed for their Sense/Sound Harmony.

Here again, it is important to notice the majority of the poems positioned on the left hand side are thus analyzed as possessing negative harmony and only three poems on the right hand side, one of which is the unfinished "Untitled". And then, in the middle, there is a small number of disharmonic poems, or we could call them poems in which there were conflicting forces contributing to the overall meaning intended by the poem. Also take into account the dimension of the box which signals the major or minor contribution of the overall parameters computed as discussed in previous section, of all the linguistic and poetic features contained in the poem, but measured on the basis of their minor or major dispersion using standard deviation. In the following group, I added more poems from later work, which were computer mainly as positive:

Birthday—Hopkins and Foster's Dam

Socrates—A Death at Winson Green, Eyre All Alone, Bells of St Peter Mancroft

The Ghost of the Cock—Around Costessey, Nessun Dorma

Late Poems 1969–73—Lament for St Maria Goretti, St Therese and the Child

As showns before, also in Figure 15 the poems are positioned in three separate rows according to their overall sentiment:

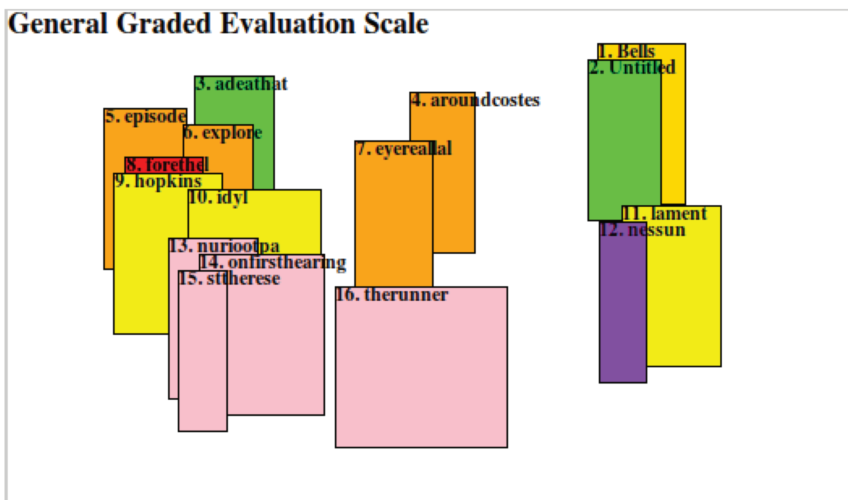


Figure 15. Sixteen poems taken mainly from late poetic production computed for their sense/sound harmony.

In Figure 16, I will now show a bigger picture containing 50 poems, where we can see again the great majority of them being positioned on the left hand side. The positive side is enriched by "Moonlight" from *Early Poems*, and "Song of the Brain" from *Socrates*, and the middle disharmonic list now counts 16 poems.

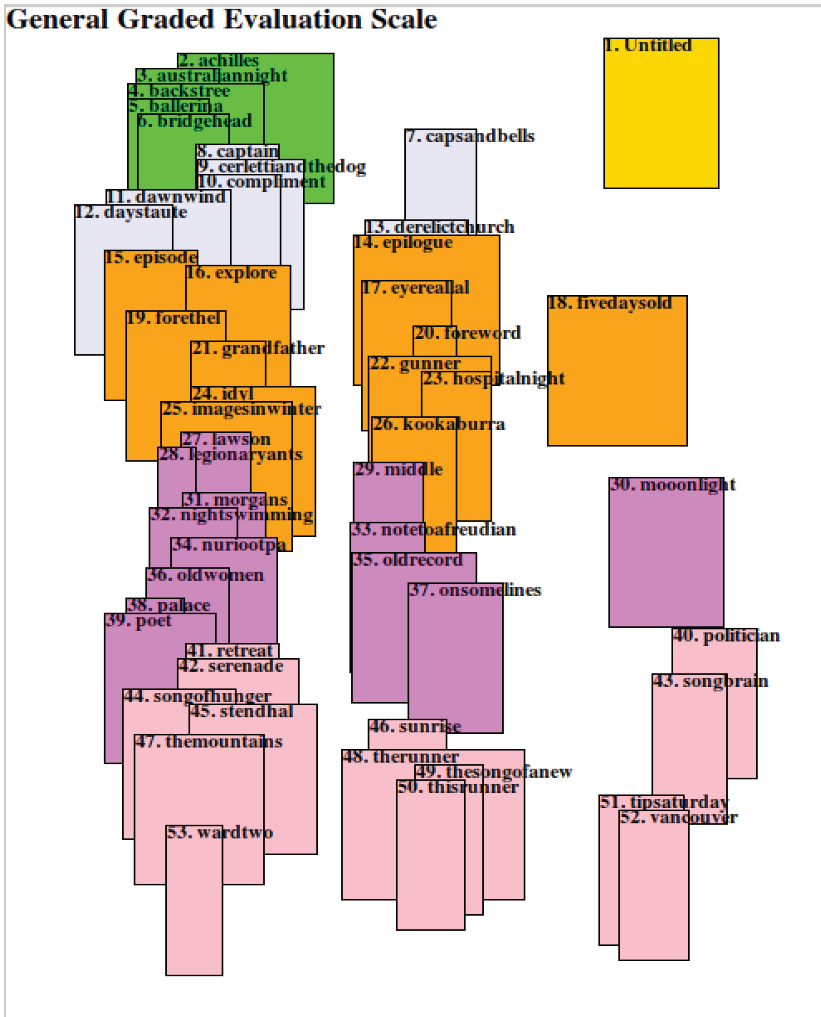


Figure 16. Fifty poems computed by sense/sound harmony.

So, we can safely say that the great majority of Webb's poems contain a negative harmony. This is further confirmed by the following Figure 17, which represents the analysis of 87 poems. I decided not to increase the number of poems up to 130 as was the case with the APSA system simply because otherwise the image becomes too difficult to read and poems' labels will be too cluttered together.

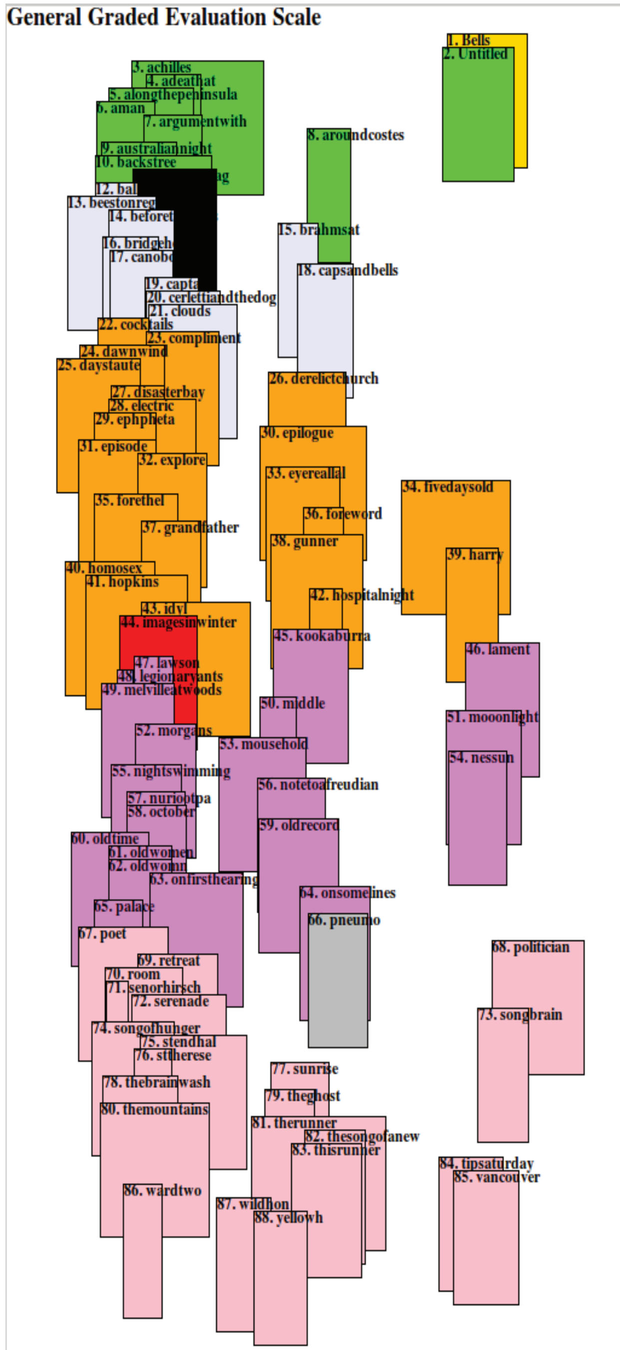


Figure 17. Sound–sense harmony in Webb's 87 poems.

4. Discussion

As now appears more clearly, the sound–sense harmony poses strict requirements on the execution of the overall experiment, which is composed of a first part dedicated to sound harmony, thus deriving the poet's major or minor intention to fill completely the harmonic scheme with the four classes of sounds available. For this first part of the experiment, the paper has been mainly concentrated on Shakespeare's sonnets, which require a much harder level of elaboration in order to complete the sound–sense harmony experiment due to the presence of rhyme violations. As the data presented have extensively shown, sounds in Shakespeare's sonnets are mainly distributed in the four classes and the three main classes; only a few sonnets have two classes and only one sonnet has one single class. The distribution is not casual as discussed above and responds to requirements imposed by the contents. In order to obtain such an important but preliminary result, all rhyming pairs had to undergo a filtering check to evaluate their role in the overall rhyming scheme of the sonnet. In case of rhyme violation, the lexicon would have to be checked and the appropriate phonetic variation inserted.

We have then shown that the sound–sense relation may represent similar but distinct situations: in case of disharmony, we may be in presence of ironic/sarcastic expressions, as happens in Shakespeare's sonnets. This is derived from the data: as shown above, the correlation has a negative trend, meaning that the two main variables—the ones defining the behaviour of the sound patterns, and the other the behaviour of the sense, in this case the sentiment pattern—diverge and move in opposite directions. On the contrary, in the case of Webb's poetry, the contrast—when present—represents his need to encompass the opposites in life and this is testified by the frequent use of oxymora and by his condition of outcast rejected by society. Data for Webb show a great agreement in negatively marked sound–sense harmony and a much reduced agreement for positively marked data. Webb has lived half of his life in psychiatric hospitals rejected by the people who knew him, and was only accepted as a poet.

The use of two sense-related approaches has allowed us to differentiate what sentiment analysis reduced to two parameters. With the Appraisal Theory Framework, we thus managed to better specify the nature of negative sentiment using more fine-grained distinctions derived from the tri-partite subdivision of Attitude into Judgement, Appraisal and Affect. The data confirmed the previous analysis but allowed a further distinction of negatively marked sonnets into sarcastic vs. ironic.

The approach has been proven general enough to encompass poets embodying the widest possible gap from the cultural, linguistic and poetic point of view. Current DNNs are unable to cope with this task which is highly complex. It requires a sequence of carefully wrought processes in order to produce a final evaluation: in particular, the first task that is problematic for AI systems like ChatGPT is an as faithful as possible phonetic transcription of each poem. When asked to produce one such transcription, ChatGPT carried it out using IPA symbols, but as for the ARPAbet version, the result was a disaster. Word stress was assigned correctly only for a 75% of the words. The reason for this situation is very simple: dictionaries for DNN models number over one million distinct word forms and there is no resource available which counts more than 200,000 fully transcribed entries. The solution is to provide rule-based algorithms but we know that DNNs are just the opposite. They are unable to generalize what they might have learnt from a dictionary to new unseen word forms [40]. In addition, transcribing in another language—like Italian—has resulted in a complete failure. And phonetic transcription is just the first step in the pipeline of modules which are responsible for the final evaluation, as the previous section has clarified.

5. Conclusions

In this article, we have proposed a totally new technique to assess and appreciate poetry, the ***algorithm for Sound–Sense harmony (ASSH)***. In order to evaluate poetry, we associated the phonetic image of a poem as derived from stressed syllables of rhyming words with the computed semantic and pragmatic meaning of the clauses contained in the poem. Meaning is represented by so-called "sentiment analysis" in a first approach and then by the "appraisal theory framework" in a second approach, which has offered a more fine-grained picture of the contents of each poem. We tested the technique with the work of two famous poets, Shakespeare—an Elizabethan poet—and Francis Webb, a contemporary poet. The results obtained show the possibility to reclassify ASSH into two subcategories: **disharmony** and **harmony**, where the majority of Shakespeare's sonnets belong to the first and Webb's poetry—and as I assume the majority of current poetry—to the second. Disharmony is characterized by the presence of a marked opposition between classes—both phonetically and semantically; on the contrary, harmony is characterized by a convergence of sound and sense in the two possible nuances, negative and positive.

The data from Shakespeare's sonnets have been analyzed by usual methods with graphic charts; in the case of Webb, a new methodology has been proposed, by projecting on a graphic space the image of a poem based on its parameters, in a three dimensional manner. This is performed by drawing a coloured box representing each poem which can vary its shape according to its relevance, while its position varies according to the overall semantic parameters computed. The position of the box is assigned on one of the three sides into which the graphic space is organized: left for negatively marked harmonic poems, center for disharmonic ones, and right for positively marked harmonic poems. Boxes may vary slightly their position in one of the sides assigned according to their parameters. Differently from the results obtained for Shakespeare's sonnets, Webb's poetry—we tested the system with 100 of the most important poems—is thus characterized by a majority of poems positioned on the left, i.e., possessing negatively marked parameters for SSH. Finally, disharmony has at least two possible interpretations: in the case of Shakespeare, it represents an ironic/sarcastic mood, while in Webb's poetry, it is the result of the internal struggle for psychic survival. The method has thus been shown to be most general and applicable to any type of poetry characterizing the poet's personality by ASSH's deep analysis of the explicit and implicit contents of her/his poetic work.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/info14100576/s1.

Funding: This research received no external funding.

Data Availability Statement: We make available data of the complete analysis of the 154 sonnets and of 100 Webb's poems used in his section. Data is contained within supplementary material.

Acknowledgments: The ATF classification task has been carried out partly by Nicolò Busetto, co-author of a number of papers describing the work done. Thanks to two anonymous reviewers for the stimulating and inspiring comments that allowed me to improve the paper.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Jakobson, R. *Six Lectures on Sound and Meaning*; MIT Press: Cambridge, MA, USA, 1978.
2. Jakobson, R.; Waugh, L. *The Sound Shape of Language*; Indiana University Press: Bloomington, IN, USA, 1978.
3. Mazzeo, M. Les voyelles Colorées: Saussure et la Synesthésie. *Cah. Ferdinand Saussure* **2004**, *57*, 129–143.
4. Fónagy, I. The Functions of Vocal Style. In *Literary Style: A Symposium*; Chatman, S., Ed.; Oxford University Press: Oxford, UK, 1971; pp. 159–174.
5. Macdermott, M.M. *Vowel Sounds in Poetry: Their Music and Tone Colour, Psyche Monographs, No.13*; Kegan Paul: London, UK, 1940.
6. Tsur, R. *What Makes Sound Patterns Expressive: The Poetic Mode of Speech-Perception*; Duke University Press: Durham, NC, USA, 1992.
7. Brysbaert, M.; Warriner, A.B.; Kuperman, V. Concreteness ratings for 40 thousand generally known English word lemmas. *Behav. Res. Methods* **2014**, *46*, 904–911. [CrossRef] [PubMed]
8. Mohammad, S. Even the Abstract have Colour: Consensus in Word Colour Associations, 2011b. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011.
9. Melchiori, G. *Shakespeare's Sonnets*; Adriatica Editrice: Bari, Italy, 1971.
10. Taboada, M.; Grieve, J. Analyzing appraisal automatically. In *AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*; AAAI Press: Washington, DC, USA, 2004; pp. 158–161.
11. Delmonte, R. Visualizing Poetry with SPARSAR—Poetic Maps from Poetic Content. In Proceedings of the NAACL-HLT Fourth Workshop on Computational Linguistics for Literature, Denver, CO, USA, 4 June 2015; pp. 68–78.

12. Montaño, R.; Alías, F.; Ferrer, J. Prosodic analysis of storytelling discourse modes and narrative situations oriented to Text-to-Speech synthesis. In Proceedings of the 8th ISCA Speech Synthesis Workshop, Barcelona, Spain, 31 August–2 September 2013; pp. 171–176.
13. Delmonte, R.; Tonelli, S.; Boniforti, M.A.P.; Bristot, A. VENSES—A Linguistically-Based System for Semantic Evaluation. In *Machine Learning Challenges*; Quiñonero-Candela, J., Dagan, I., Magnini, B., Florence , d'Alché-Buc, F., Eds.; Springer: Berlin, Germany, 2005; pp. 344–371.
14. Bresnan, J. (Ed.) *The Mental Representation of Grammatical Relations*; The MIT Press: Cambridge, MA, USA, 1982.
15. Bresnan, J. (Ed.) *Lexical-Functional Syntax*; Blackwell Publishing: Oxford, UK, 2001.
16. Baayen, R.H.; Piepenbrock, R.; Gulikers, L. *The CELEX Lexical Database (CD-ROM)*; CELEX2 LDC96L14. Web Download; Linguistic Data Consortium: Philadelphia, PA, USA, 1995.
17. Bacalu, C.; Delmonte, R. Prosodic Modeling for Syllable Structures from the VESD—Venice English Syllable Database. In *Atti 9 Convegno GFS-AIA*; Sirea: Venice, Italy, 1999.
18. Tsur, R. *Poetic Rhythm: Structure and Performance: An Empirical Study in Cognitive Poetics*; Sussex Academic Press: Eastbourne, UK, 2012; p. 472.
19. Greene, E.; Bodrumlu, T.; Knight, K. Automatic Analysis of Rhythmic Poetry with Applications to Generation and Translation. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Cambridge, MA, USA, 9–11 October 2011; pp. 524–533.
20. Carvalho, P.; Sarmento, L.; Silva, M.; de Oliveira, E. Clues for detecting irony in user-generated contents: Oh...!! it's so easy;-). In Proceedings of the 1st International CIKM Workshop on Topic-Sentiment Analysis for Mass Opinion, Hong Kong, 6 November 2009; ACM: New York, NY, USA, 2009; pp. 53–56.
21. Kao, J.; Jurafsky, D. A Computational Analysis of Style, Affect, and Imagery in Contemporary Poetry. In Proceedings of the NAACL Workshop on Computational Linguistics for Literature, Montréal, ON, Cananda, 8 June 2012.
22. Kim, S.-M.; Hovy, E. Determining the sentiment of opinions. In Proceedings of the 20th International Conference on Computational Linguistics—COLING, Stroudsburg, PA, USA, 23–27 August 2004; pp. 1367–1373.
23. Reyes, A.; Rosso, P. Mining subjective knowledge from customer reviews: A specific case of irony detection. In Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis WASSA '11, Stroudsburg, PA, USA, 24 June 2011; pp. 118–124.
24. Weiser, D.K. Shakespearean Irony: The 'sonnets'. In *Neuphilologische Mitteilungen*; Modern Language Society: NY, USA, 1983; Volume 84, pp. 456–469. Available online: http://www.jstor.org/stable/43343552 (accessed on 6 July 2023).
25. Weiser, D.K. *Mind in Character—Shakespeare s Speaker in the Sonnets*; The University of Missouri Press: Columbia, MI, USA, 1987.
26. Attardo, S. *Linguistic Theories of Humor*; Mouton de Gruyter: Berlin, Germany; New York, NY, USA, 1994.
27. Schoenfeldt, M. *Cambridge Introduction to Shakespeare's Poetry*; Cambridge University Press: Cambridge, UK, 2010.
28. Ingham, R.; Ingham, M. Chapter 5: Subject-verb inversion and iambic rhythm in Shakespeare's dramatic verse. In *Stylistics and Shakespeare's Language: Transdisciplinary Approaches*; Ravassat, M., Culpeper, J., Eds.; Continuum: London, UK, 2011; pp. 98–118.
29. Delmonte, R. Exploring Shakespeare's Sonnets with SPARSAR. *Linguist. Lit. Stud.* **2016**, *4*, 61–95. Available online: https://www.hrpub.org/journals/jour_archive.php?id=93&iid=772 (accessed on 6 July 2023). [CrossRef]
30. Delmonte, R. Poetry and Speech Synthesis, SPARSAR Recites. In *Ricognizioni—Rivista di Lingue, Letterature e Culture Moderne*; Università Di Torino: Torino, Italy, 2019; Volume 6, pp. 75–95. Available online: http://www.ojs.unito.it/index.php/ricognizioni/article/view/3302 (accessed on 6 July 2023).
31. Crystal, D. Sounding out Shakespeare: Sonnet Rhymes in Original Pronunciation. 2011. Available online: https://www.davidcrystal.com/GBR/Books-and-Articles (accessed on 6 July 2023).
32. Mazarin, A. The Developmental Progression of English Vowel Systems, 1500–1800: Evidence from Grammarians, in Ampersand. 2020. Available online: https://reader.elsevier.com/reader/sd/pii/S2215039020300011?token=BCB9A7B6C95F35D354C940E08CBA968ED124CF160B0AC3EF5FE9146C4B3885E825A878104ED06E127685F139918CCEB6 (accessed on 6 July 2023).
33. Crystal, D. *Think on My Words: Exploring Shakespeare's Language*; Cambridge University Press: Cambridge, UK, 2008.
34. McGuire, P.C. Shakespeare's non-shakespearean sonnets. *Shakespear. Q.* **1987**, *38*, 304–319. [CrossRef]
35. Attardo, S. Irony as relevant inappropriateness. *J. Pragmat.* **2000**, *32*, 793–826. [CrossRef]
36. Martin, J.; White, P.R. *Language of Evaluation, Appraisal in English*; Palgrave Macmillan: London, UK; New York, NY, USA, 2005.
37. Read, J.; Carrol, J. Annotating expressions of appraisal in English. *Lang. Resour. Eval.* **2012**, *46*, 421–447. [CrossRef]
38. Delmonte, R.; Busetto, N. Detecting irony in Shakespeare's sonnets with SPARSAR. In Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy, 13–15 November 2019. Available online: https://dblp.org/rec/bib/conf/clic-it/DelmonteB19 (accessed on 6 July 2023).
39. Webb, F. *Collected Poems*; Davidson, T., Ed.; University of Western Australia Publishing: Crawley, Australia, 2011.
40. Delmonte, R. What's wrong with deep learning for meaning understanding. In Proceedings of the 2nd Italian Workshop on Explainable Artificial Intelligence (XAI.it 2021), Virtual, 1–3 December 2021.

information

Article

Morphosyntactic Annotation in Literary Stylometry

Robert Gorman

Department of Classics and Religious Studies, College of Arts and Sciences, University of Nebraska–Lincoln, Lincoln, NE 68588, USA; rgorman1@unl.edu

Abstract: This article investigates the stylometric usefulness of morphosyntactic annotation. Focusing on the style of literary texts, it argues that including morphosyntactic annotation in analyses of style has at least two important advantages: (1) maintaining a topic agnostic approach and (2) providing input variables that are interpretable in traditional grammatical terms. This study demonstrates how widely available Universal Dependency parsers can generate useful morphological and syntactic data for texts in a range of languages. These data can serve as the basis for input features that are strongly informative about the style of individual novels, as indicated by accuracy in classification tests. The interpretability of such features is demonstrated by a discussion of the weakness of an "authorial" signal as opposed to the clear distinction among individual works.

Keywords: stylometry; Universal Dependencies; authorship attribution

1. Introduction

Stylometry is a discipline that attempts to apply rigorous measurement to the traditional concerns of stylistics. Stylistics involves the identification and evaluation of certain characteristics that may distinguish the language use of individuals, groups of individuals, genres, etc. For humanists, the objects of stylometric study are most frequently literary, historical, or philosophical texts. Recent years have seen much research in the application of stylometrics in the humanities, and this work has produced many advances in the field. However, there remain important weaknesses in the predominant methods in the field. The present study is an attempt to address aspects of these weaknesses: the lack of stylometric input features that produce results that are both (1) topic agnostic and (2) directly interpretable in traditional terms.

Generally, to be of interest to researchers, the characteristics of the "style" of a text must be distinctive enough to allow us to discriminate that text from other relevant material. Thus, from the early days of stylometrics [1], success in classification experiments has served to establish the stylometric value of the input features on which accurate classifications were based. Of course, considerations other than high accuracy must also be considered when evaluating input features. For example, the frequency profiles for a set of common words [2] or common word sequences—word n-grams [3] are generally quite effective for classification, but because these input features may include "lexical" words, they are usually avoided when the style of an individual writer is the focus. Lexical words, also called "content" words, can be strongly influenced by topic, genre, etc., and this influence may confound classification. In such a case, researchers rely upon features considered to be "topic agnostic" since they are not closely and directly dependent on the subject matter of the text in question. Chief among these topic agnostic inputs are "function" words and character n-grams. Unlike lexical words, function words (for example, prepositions, conjunctions, determiners, etc.) belong to a small, closed set. In spite of this fact, function words as a group are used more often than content words [4]. In addition, function words more closely reflect syntactic structure than semantic content. Thus, we can reasonably assume that function words are relatively free of confounding effects. Some function words may, however, be more closely dependent on genre or topic than others. Gendered

Citation: Gorman, R. Morphosyntactic Annotation in Literary Stylometry. *Information* **2024**, *15*, 211. https://doi.org/10.3390/info15040211

Academic Editor: Horacio Saggion

Received: 30 January 2024
Revised: 14 March 2024
Accepted: 15 March 2024
Published: 9 April 2024

pronouns, for example, are for this reason sometimes removed from studies of function words [5]. On the other hand, while function words often allow for accurate classification and therefore clearly capture something distinctive about many texts, it is difficult to translate the frequency profile of a set of prepositions, conjunctions, etc., into a detailed understanding of the style of a text.

Character n-grams, which recently have become quite popular in textual studies, share, but to a more extreme degree, the advantages and disadvantages of function words. Consisting of character sequences without regard to their position in a word, their order in a sentence, etc., character n-grams represent a text at a sub-lexical level although, because spaces between words are usually counted as "characters", rough information about word boundaries is reflected in this input. For this reason, they are generally free of the criticism that they are closely dependent on external factors such as topic or intended audience (for reservations, see [6]). However, it should be obvious that the frequency distribution of randomized sequences of letters is practically uninterpretable in terms more of traditional approaches to style, and character n-grams are therefore uninteresting from that perspective. Thus, approaches to stylometry are closely connected to the ongoing debate in machine learning and related fields about the relative advantage of choosing, on the one hand, heuristic input features that may be difficult to interpret and, on the other hand, input that represents a symbolic structure such as syntax. (For a recent examination of the topic with a bibliography, see [7]).

Another example of a non-morphosyntactic computational analysis of texts is front-back vowel harmony testing [8,9], which tests whether there is a tendency of having words to have only front vowels or only back vowels. Front-back vowel harmony is so characteristic of certain languages that this feature can be detected even if these languages are written in an undeciphered syllabic script [8].

This paper is an introduction to the stylometric and stylistic value of the morphosyntactic information provided in the annotations of the Universal Dependency treebanks. First, using the standard criterion of text/authorship attribution, it will demonstrate that morphosyntactic input features can successfully discriminate among texts without the identification of any vocabulary items. This result indicates that these features can be effective while being topic agnostic. Second, it will show that many morphosyntactic input features can be interpreted in a relatively straightforward way that is consistent with terms and concepts long used in the precomputational study of literary style.

Advances in the field of interpretable machine learning have provided important tools for expanding the usefulness of ML by making results easier to understand, even with "black-box" algorithms (my thanks go to the anonymous reviewer for emphasizing this point). It nonetheless remains an advantage, at least when attempting to persuade researchers in literary fields of the validity of computational approaches, to select input variables that are explainable by referring to traditional stylistics. The academic study of literary style has its roots in the traditional disciplines of Poetics and Rhetoric [10]. Both approaches agree that among the most important parts of a description of the style of a text or corpus are analyses of diction and word arrangement.

Diction is essentially word choice or vocabulary. This traditional focus is also central to our investigations, in that information about every word in every text analyzed is included in our data. At the same time, morphological annotation allows us to abstract away from individual vocabulary choices. Each word is included in our input features not as a token of a particular lexical item but rather as a representation of the relevant morphosyntactic categories (part-of-speech, singular or plural, subject or direct object, etc.). Thus, in accordance with the traditional importance of diction in stylistic research, words remain the basic unit of analysis in this study, but in a way that seeks to be topic agnostic and avoid the confounding effects often introduced by a consideration of vocabulary.

The traditional importance of word arrangement to stylistic research in the humanities is also reflected in the input features chosen for this study. Information about the syntactic annotation of every word in the corpus is reflected in the input features. Syntactic annotation explicitly encodes the relationship between words, and therefore, its use as a dimension of analysis can be seen as a natural expansion of a traditional approach.

Thinking of the morphosyntactical features used in these experiments as computational "enhancements" of the traditional pillars of literary style, diction and arrangement, will, it is hoped, promote a broad understanding of the approach. Interpretability should also be increased by our use of traditional terminology. Terms used for morphological categories and values are known to any serious researcher in literary style. The widespread adoption of dependency grammar is, admittedly, relatively recent, but generally, the protocols of dependency grammar are closely related to the traditional concepts and terminology of the humanistic study of language and literature.

Thus, a stylometric analysis based on the features presented here is well suited to contribute to a thoroughgoing investigation of the style of a work or author in a way that should be interpretable to a wide range of readers. In addition, in the course of our discussion, it will become clear that this morphosyntactic approach is effective, with minimal adjustments, across a range of languages. This quality is beneficial in a field where English texts have been the predominant source of, and testing ground for, stylometric methods [11]).

The organization of the remainder of this paper is as follows. First, in Section 2 the various corpora are described and a step-by-step construction of effective input features from morphosyntactic annotation is described. Section 3 explains the classification experiment used to demonstrate the stylometric value of the input features. In Section 4, the results of the classification are briefly discussed. In Section 5 morphosyntactic frequencies are used as the basis of an investigation into the interrelations between the "local" characteristics of individual novels and the more general "authorial" signature. A short conclusion rounds off the article.

2. Corpora and Morphosyntactic Input Features

The input features used in this study are derived from texts annotated according to the framework used in the Universal Dependencies Treebank Collection [12,13]. The Universal Dependencies (UD) project is an open community effort that has been growing rapidly in recent years. The project has given impetus to the development and publication of software implementing pipelines for tokenization, tagging, lemmatization and dependency parsing of texts in a wide range of languages. These invaluable programs—called UDPipes—cover a wide range of languages and are available for the R and Python environments as well as through a convenient web interface (https://lindat.mff.cuni.cz/services/udpipe/, accessed 1 January 2024).

Because the focus of this paper is the advantages of morphosyntactic features as stylometric tools for the humanities, corpora consisting of a selection of novels have been chosen. While much recent stylometric work has concentrated on social media texts and the like, this material is less central to the interests of humanists than more traditional literary writing. Our corpus includes novels in English, French, German, and Polish. A more diverse set of languages would have been preferable, but such works meeting the requirements of our study were not readily available. The design of our investigation calls for literary works that are similar in genre and chronology. As many authors as possible should be represented, and for each author, the set should include three separate novels. These criteria could be met by combining reference corpora freely available at github.com. The English, German, and Polish novels were made available by the Computation Statistics Group (https://github.com/computationalstylistics, accessed 1 January 2024). The French novels were a resource provided by *Computerphilologie Uni Würzburg* (https://github.com/cophi-wue/refcor, accessed 1 January 2024). Because we will compare the performance of morphosyntactic input variables for works in the different languages, all corpora were limited to the size dictated by the smallest set (German). As a result, each language corpus contains 15 authors, each represented by three different works. In order to facilitate comparisons between the languages, only the first 20,000 tokens (excluding punctuation) of each work in each corpus are considered.

After the collection of a suitable set of texts, the next step is to generate the basic annotations from which the input features will be assembled. This processing is carried out with the appropriate UDPipes through the "udpipe" package for the R Software Environment [14] (R version 4.2.1). Raw text (.txt files) provided to the udpipe produces output in the CONLL-U format. An example of this output is given below (Tables 1–4).

Table 1. "Shallow" annotation output by UDPipe. Sentence: "It gives us the basis for several deductions" (Doyle, *The Hound of the Baskervilles*, 1901).

Token	Lemma	Upos	Feats
It	it	PRON	Case = Nom | Gender = Neut | Number = Sing | Person = 3 | PronType = Prs
gives	give	VERB	Mood = Ind | Number = Sing | Person = 3 | Tense = Pres | VerbForm = Fin
us	we	PRON	Case = Acc | Number = Plur | Person = 1 | PronType = Prs
the	the	DET	Definite = Def | PronType = Art
basis	basis	NOUN	Number = Sing
for	for	ADP	NA
several	several	ADJ	Degree = Pos
deductions	deduction	NOUN	Number = Plur

Table 2. "Deep" annotation output by UDPipe. Sentence: "It gives us the basis for several deductions" (Doyle, *The Hound of the Baskervilles*, 1901).

Head_Token_Id	Dep_Rel
2	nsubj
0	root
2	iobj
5	det
2	obj
8	case
8	amod
5	nmod

Table 3. "Shallow" annotation by UDPipe. Sentence: "There, however, stood only a single bowl" (Spyri, *Heidi*, 1880).

Token	Lemma	Upos	Feats
Da	Da	ADV	NA
stand	stehen	VERB	Mood = Ind | Number = Sing | Person = 3 | Tense = Past | VerbForm = Fin
aber	aber	ADV	NA
nur	nur	ADV	NA
ein	ein	DET	Case = Nom | Gender = Neut | Number = Sing | PronType = Art
einziges	einzig	ADJ	Degree = Pos | Gender = Neut | Number = Sing
Schüsselchen	Schüsselchen	NOUN	Gender = Neut | Number = Sing | Person = 3

Table 4. "Deep" annotation by UDPipe. Sentence: "There, however, stood only a single bowl" (Spyri, *Heidi*, 1880).

Head_Token_Id	Dep_Rel
2	advmod
0	root
2	advmod
5	advmod
7	det
7	amod
2	nsubj

For each token, the analysis gives the form as it appears in the text and its lemma. This information is not used in the method described here since our goal is to examine the discriminative power of morphosyntactic features. In addition, as noted above, general vocabulary may be largely dependent on genre or subject matter and may confound analysis. It is worth noting that the elimination of word forms and lemmas from consideration simplifies preprocessing and, to some degree, compensates for the time required to extract input features from the parsed text. Minimal clean-up of the .txt file is required; chapter titles and the like can be left in the document without affecting the results of the classification.

Leaving aside the form and lemma, the remaining columns in the UDPipe output shown in Tables 1–4 are essential to our method. The "upos" column contains the UD part-of-speech tags for each word. The "feats" column gives the morphological analysis. Morphology information has the form "TYPE = VALUE, with multiple features separated by a bar symbol (TYPE1 = VALUE | TYPE2 = VALUE). Consider, for example, the morphological data supplied for the word *us* in Table 1. The upos column assigns *us* to "pronoun" as its part of speech. The "feats" column then gives the following information: Case = Acc | Number = Plur | Person = 1 | PronType = Prs. This annotation can be read as follows. The grammatical case of *us* is accusative; its grammatical number is plural; it refers to the speaker, so it is considered grammatically a first-person word; lastly, *us* belongs to the pronoun subtype "personal".

A comparison of the "feats" column in Table 1 with that in Table 3 reflects an important typological difference among languages. Languages can vary significantly in their morphological complexity. For example, English nouns (*basis* and *deductions* in Table 1) are generally annotated only for grammatical number, while English adjectives (*several* in Table 1) show only grammatical degree (i.e., positive, comparative, and superlative). In contrast, German nouns (*Schüsselchen* "bowl" in Table 3) and adjectives (*einziges* "single") are considered to have grammatical gender (and case) as well as number. In addition to the natural differences between languages, complications can be introduced by the parser. For example, the UDPipe version used in this study ("german-hdt-ud-2.5-191206.udpipe" Wijffels 2019) does not assign a case to every instance of a noun or adjective. Instead, explicit annotation of case is generally restricted to words in which different cases are indicated morphologically: for example, occurrences of *Kindes* ("child's") are marked as genitive of the noun *Kind* ("child"); and occurrences of *bösem* are marked as dative of the adjective *böse* ("bad").

Parts of speech and morphology constitute what we may call "shallow" syntactic features. These features reflect some syntactical structures, but do not represent them directly. In contrast, the "head_token_id" and "dep_rel" columns are a direct representation of syntactic organization. The head token is the item that is the immediate syntactic "parent" of a given token. The "dep_rel" reports the dependency-type label as specified in the UD annotation guidelines. The dependency relation specifies the type of grammatical structure between the parent and target. From these columns, we can calculate the grammatical structure of an entire sentence, as visualized in a dependency tree such as the one shown below (Figure 1).

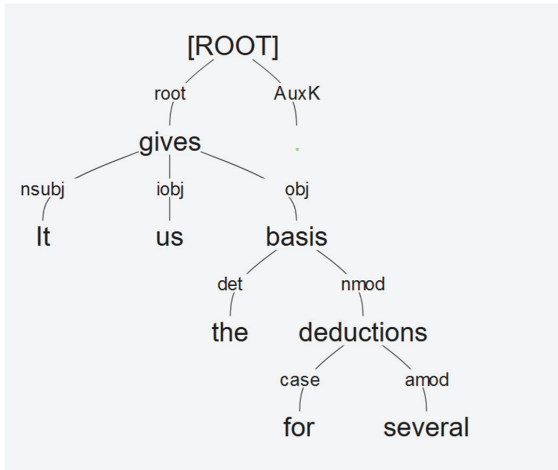


Figure 1. Universal Dependency tree for "It gives us the basis for several deductions".

The syntactic "path" from the sentence root to each "leaf" token is given by the combination of head id and dependency relationship. The syntactic function of each word is clearly and specifically defined by these two values. For example, the word *basis* is the *obj* of the word *gives. obj* is the UD label for what is traditionally called the "direct object" of a verb (a list of syntax labels along with examples can be found on the UD website: https://universaldependencies.org/en/dep/index.html, accessed 1 January 2024). The word *several* is labeled as *amod* of *deductions*. *amod* indicates an adjectival modifier. The word *deductions* itself is an *nmod* of *basis*. In UD annotation, *nmod* means "nominal modifier", a noun or noun phrase directly dependent on and specifying another noun (or noun phrase). For example, the prepositional phrase in "toys for children".

It is important to recognize the special importance of the "head_token_id" annotation. Because its values specify the configuration of the dependency tree for each sentence, head token information can also be used to add structural/syntactic "depth" to the "shallow" morphological data. For example, examined against the background of the dependency tree, the German word *Schüsselchen* ("bowl") is no longer just a neuter noun, but a neuter noun that is dependent on a past tense verb, or a neuter noun that is dependent on the main verb, etc. Thus, the head token annotation allows us to consider the "syntactic sequence" of words, a hierarchically ordered analogue to the chronologically ordered sequence encoded in traditional n-grams.

The input features in our study are composed primarily of the three kinds of information discussed above: (1) morphological annotation; (2) syntactic information; and (3) morphosyntactic "n-grams" containing combinations of morphological and syntactic data from words that are hierarchically contiguous in the dependency tree of a sentence.

When constructing input features from morphosyntactic annotation, it is important to design the features in a way that preserves information while avoiding sparsity. We can achieve this goal by incorporating, in the input features for a single word, a series of combinations of individual morphosyntactic values. In many languages, a morphological analysis of a word may be relatively complex. For example, the UD annotation for the German word *stand* (Table 3) indicates that its morphology may be identified as a *past indicative third-person singular finite verb.* Naturally, as the number of more or less independent values in a given complex annotation increases, the frequency of that set of values will correspondingly decrease. Thus, while 12.5% of the words in Spyri's *Heidi* are annotated with the part of speech *verb*, only 4.2% are marked with a combination of the *verb* annotation and the tense annotation *past*. If we take syntactic function into consideration, only 1.6% of words are a past tense verb whose relationship is annotated as *root* (i.e., the main verb of a sentence). Such a sharp tendency toward sparsity will rapidly compromise the effectiveness of morphosyntactic data. To avoid this effect, we distribute the full morphological and syntactic annotation for each word into a set of combinations made from its assigned grammatical values. In this way, for example, each verb is associated with an input feature giving its tense, another giving its mood, another its person, etc. Then, all binary combinations of types are generated (e.g., tense and mood, tense and person, or mood and person). The same is carried out for ternary combinations. The result is a framework for organizing input features that is satisfactorily informative while maintaining an acceptable level of sparsity.

In addition to encoding the morphosyntactic information for each word in the text, we take advantage of the opportunity afforded by the head token annotation to enrich the data, as mentioned above. For each word that is not annotated as the *root* of a sentence, we include input features constructed from the morphosyntactic annotation of that word's dependency "parent". For example, the input features for *deductions* in Table 1 would include combinations made from *basis* as well as *deductions* itself. These syntactically ordered n-grams bring a measure of structural depth to otherwise shallow "surface" morphological information.

In addition to the morphosyntactic categories discussed so far, we have also included a small additional group in the feature set. Natural language may be conceptualized as a hierarchical structure (as illustrated in dependency treebanks) projected onto a linear order, the chronological sequence of words in texts or speech. Word order, as well as word hierarchy, can represent crucial stylometric information. We capture some of this linear information by adding two values to the annotations provided by UDPipe: dependency distance (DD) and dependency direction (DDir).

DD is the distance in the linear order of a sentence between a given word and its parent word, measured by the number of words. More precisely, DD can be thought of as the absolute value of the difference between the linear index of a word (its position in the linear sequence) and the linear index of the word's parent. Thus, in our example sentence, "It gives us the basis for several deductions", the DD of the word *us* is 1: the index of *us* = 3, the index of parent word *gives* = 2; hence, $3 - 2 = 1$. As treebanks of many languages become widely available, research on DD is becoming more important. DD has been suggested as a proxy for sentence complexity [15–17] and as an explanation for aspects of word order. It is therefore reasonable to include DD among our input features on the assumption that it represents something important about the style of a text. A second addition to our set of categories is dependency direction (DDir). This category is quite simple. A value is assigned to each word (except for sentence roots) indicating whether it comes before or after its parent word in the linear order of the sentence. Word order has long been a staple of analyses of stylistics, so it naturally finds a place in a stylometric study (computational studies based on treebank data for DDir tend to be focused on typological questions rather than stylistics ones [18]).

The restriction of our input features to unary, binary, and ternary combinations of annotation categories is an attempt to balance the desire to include the widest range of possibly useful stylometric data with the need to avoid a sparse set of inputs. Nevertheless, additional culling of the input features is necessary. The limitation of combinations to more than three elements still allows for over 16,000. And each of these combinations is a type, not a variable. Each component of a given type may represent more than one value; the ternary combination gender–number–case may take one of 24 different value combinations in German (3 genders $\times$ 2 numbers $\times$ 4 cases). Thus, even our restricted set of combinations, when populated with the appropriate values, would be computationally unfeasible. We have addressed this problem with a naïve approach. Since we cannot know in advance which combinations may be most distinctive for authors and texts, we have selected among them based on frequency alone. For each combination length, only those type–value pairs which occur in approximately 5% of the tokens in the corpus have been included as input variables for classification. The process of populating feature types with their values is computationally slow for combinations of more than two elements, so we have used a smaller sample corpus for each language. Thus, the 5% cut-off is an approximation. A separate set of variables has been identified in this way for each language. Because UDPipe produces different types of morphological annotation, and because syntactic annotation, although it largely consists of the same relationship labels, has different frequency distributions in various languages, the same selection procedure with the same 5% cut-off results in a different quantity of features for each language. Details are given in Table 5.

Table 5. Number of input features by number of type–value components in each feature.

	Unary	Binary	Ternary	Total
English	55	231	367	653
French	59	337	629	1025
German	63	325	673	1061
Polish	65	337	735	1137

The nature of these features may be difficult for the reader to visualize from a description alone. The examples given below in Section 5 should provide illustration.

3. Classification

The purpose of this study is to examine the effectiveness of morphosyntactic input features as stylometric markers of literary texts. In particular, we test the usefulness of the annotation produced by the UDPipe applications. As noted above, classification experiments are a standard means to evaluate the worth of different sets of input features. It is to be expected that the various steps implemented by UDPipe involve a greater or lesser degree of error. Since the information/noise ratio worsens for shorter input texts [19], the first round of classification tests will be performed using a range of shorter "texts" sampled from our corpora. The sample sizes are 2000, 1000, and 500 words. For the purpose of sampling, each text was treated as a "bag of words". Each token was—naively—treated as independent of all others; no further account was taken of the context of an individual token in sentence, paragraph or any other unit of composition.

Recent years have seen the rapid development of many sophisticated classification algorithms. Deep learning approaches are appearing frequently in stylometric studies (for example, [20,21]). However, in spite of the accuracy achieved by some of these approaches, they are often uninterpretable; it is unclear exactly how the algorithm arrived at a particular classification, or even just what elements of a text were considered [22]. This is not a satisfactory outcome for stylometrics in a literary or historiographical context. In such fields, understanding and explaining the style of a text or author is often the principal goal.

In an effort to combine good accuracy and a high level of interpretability, we have chosen logistic regression as the approach for this study. Logistic regression has long been used extensively in many fields and is well understood. It is a straightforward approach to identify the contribution of each input feature to the predictions produced by this method. An additional advantage is that logistic regression is able to function well in the presence of many co-linear variables. Morphosyntactic data are by nature highly inter-dependent, and this may present a problem for some approaches. In this study, regression was implemented through the LiblineaR package for the R Project for Statistical Computing [23,24]. This package offers a range of linear methods; we selected the L-2 regularization option for logistic regression.

The first experiment was designed to discover if morphosyntactic features could distinguish among the individual novels in the corpora. For each input sample text size in each language, 80% of the data were used for training the classifier and the remaining 20% were set aside for testing. Inclusion of a segment in the training or testing set was random. For example, to test 2000 word samples, each 20,000-word text in a corpus was split randomly into ten samples, eight of which were used for training and two were set aside for testing. Each training step of the classifier was therefore based on 360 samples (8 per novel for 45 novels). The procedure for other sample sizes was analogous. To validate the results of the classification testing, we used Monte Carlo sub-sampling [25] applied at two levels. As a rule, the populating of the segments with randomly selected tokens was carried out ten times. For each of these partitionings to create text segments, 50 additional random partitionings into a training set and a test set were made.

4. Results

We would expect the stylometric "signature" of individual novels to be very strong. This expectation is based on the (over-)simplifying assumption that a single literary work has a unitary style, arising from a shared theme, time of composition, etc). It should therefore not be surprising that morphosyntactic attribution with separate classes for each novel is quite successful. The results are given in Table 6, which gives the mean accuracy rate (correct "guesses"/total "guesses") for the 500 iterations in the top row of each cell, with the accuracy range reported below the mean. There are 45 classes in the data set for each language. All classifications were multi-class (one-versus-rest approach).

Table 6. Results of classification by individual novel (45 classes).

	500-Word Samples	**1000-Word Samples**	**2000-Word Samples**
English	90.6% (90.1–91.3%)	97.1% (96.4–98.1%)	99.4% (98.6–99.9%)
French	93.8% (93.1–94.6%)	96.9% (94.4–98.9%)	98.9% (97.7–100%)
German	96.3% (95.8–96.6%)	99.1% (98.8–99.3%)	99.8% (99.2–100%)
Polish	98.3% (96.8–98.9%)	99.5% (98.3–100%)	100% (100–100%)

Clearly, the works in each corpus are sharply distinguishable at the morphosyntactic level. Unfortunately, there is little published research to which these results may usefully be compared. Generally, recent stylometric research has a quasi-forensic tendency, focused on the ability to "prove" authorship of particular texts. In such cases, there is no reason to examine the discriminability of the individual works of an author. In contrast, our interest is in the *descriptive* value of stylometric measures as applied to works as well as authors. Our assumption in this study is that input features that both discriminate texts clearly and are understandable in terms of traditional stylistics may serve as the basis of valuable stylometric descriptions. Our results indicate that discriminability is high even with the relatively small 500-word samples; this success can be taken as an indication that a good deal of stylistic information is in fact conveyed by the features that we have proposed. We will examine some of the most important of these distinguishing features in the next section. It is worth mentioning here that the same procedure (albeit with different input features for each corpus) works quite well for each language tested. In fact, it is apparent from the 500-word samples that the morphosyntactic signal is somewhat stronger the more morphologically complex the language is. This complexity is reflected in the number of features as reported in Table 5: a sharper distinction seems to exist between works in Polish, which has 1137 total input features, compared to between works in English (653 total features).

5. Interpretability

In this section, we explore how the morphosyntactic input features presented here can be interpreted in a relatively straightforward manner using traditional grammatical terms. This advantage is not associated with more popular inputs such as character n-grams, which can achieve high accuracy in a classification test but do not lend themselves to clear interpretation.

In order to better illustrate the interpretability of our feature set, we will carry out our discussion against the background of an important open problem in stylometrics. This problem concerns the relationship between a stable authorial "signature" and the variability that all authors can be expected to display among their individual works. We have seen above (Table 6) that each novel in our four corpora has its own strong stylometric "signal" that allows it to be uniquely identified. Thus, all 2000-word samples in our classification were assigned to the correct work with an aggregated mean accuracy greater than 98%.

Matters are different if, instead of isolating the morphosyntactic characteristics that distinguish particular novels from each other, we try to abstract from the particular works the more general "style" of each author. Table 7 presents the results of such an experiment. Once again, data were randomly partitioned into samples of 2000, 1000, and 500 words. This time, however, classification followed the standard "leave-one-out" method. For each training iteration of the logistic regression classifier, all samples from one novel were withheld from the training data. The classes for attribution were the 15 authors in each corpus; the target class was modeled on the basis of the two novels by the relevant author remaining in the training set. In Table 7, the mean accuracy rate (correct "guesses"/total "guesses") for the 450 classification attempts is given in the top row of each cell, and the accuracy range is reported below the mean (the data for each novel were partitioned ten times into 2000-word samples; from each partitioning, 45 leave-one-out models were trained and the held-out set of samples was classified). There are 15 classes in the data set.

Table 7. Results of leave-one-out classification by author (15 classes).

	500-Word Samples	**1000-Word Samples**	**2000-Word Samples**
English	51.0% (49.6–53.5%)	56.9% (55.6–58.2%)	62.9% (60.2–64.8%)
French	54.0% (51.8–55.3%)	55.1% (54.0–56.1%)	57.1% (55.1–59.3%)
German	61.2% (59.9–63.2%)	63.0% (60.3–65.1%)	62.9% (59.7–64.0%)
Polish	44.8% (43.4–45.8%)	42.8% (42.4–44.7%)	41.5% (40.2–43.7%)

The sharp decrease in classification accuracy is striking. Presumably, an explanation is to be found in the greatly increased difficulty of the problem. The results of the most closely comparable previous studies point to the same conclusion. Maciej Eder has published three important studies on authorship attribution [19,26,27] in which the corpora are similar to our own. The accuracy of Eder's experiments is consistent with our results. For example, Eder (2010) classifies samples of various sizes drawn from 63 English novels; for samples of around 1000 words, accuracy falls between 40% and 50%. A more precise comparison is unfortunately not possible. All three of Eder's works present their results in graphs rather than tables. Thus, only rough estimates for the accuracy of a given sample size are possible. Most of Eder's data are based on the most frequent words. For a corpus of 66 German novels, samples ranging from 500 to 2000 words seem to yield accuracy scores from 30% to 60%. Evidently, the low accuracy of our authorship attribution tests (as compared to novel-by-novel classification) is not anomalous. Furthermore, it does not seem likely that the combination of input features and classifier that was quite good at identifying individual novels would become uninformative about the authorship of those same works. The field of stylometry, at least as it pertains to the realm of literary writing, relies on the assumption that each author displays a number of linguistic "peculiarities" which remain stable for some significant length of time. Of course, this more or less stable authorial "signature" only exists as it is manifested in their individual writings, and these writings naturally vary in a host of ways. While we assume that the morphosyntactic dimension of authorial style is less affected by the "local" variation among texts than other aspects of style may be, true independence is of course impossible. It is essentially inconceivable that the author's personal linguistic "signature" could be completely separable from and unaffected by certain "external" factors—a novel's plot, setting or characters, for example—when it is only in the treatment of these factors that style comes into existence.

A comparison of Tables 6 and 7 show that, to speak loosely, the authorial signal as reflected by the morphosyntactic input features is only about half as strong as that produced by the combination of author and the "local" characteristics of the individual novel. This difference in results is the context against which we will examine the interpretability of our feature set. In particular, we will choose a single text, *Oliver Twist*, an 1839 work by Charles Dickens, on which to focus our discussion. We will briefly examine input features that allow this novel to be distinguished from the other 44 works in the corpus. Then, we will do the same for features that group *Oliver Twist* together with the other two Dickens works in the corpus while at the same time distinguishing "Dickens" as a class separate from the other 14 authors represented.

There are several simple ways to identify input features that are highly discriminative with a "one-layer" classifier such as logistic regression. For example, we could select those features to which the classifier assigned the most extreme weights (positive or negative). Similarly, we could guide selection by looking at the product of the model weight and the frequency of the feature, since it is on this basis that the algorithm assigns the probability for any class. However, to avoid complicating this discussion, we will limit our focus to the frequency of occurrence of the features. In particular, we examine the standardized value of input frequencies and choose those with the largest z-scores. Because morphosyntactic values are naturally interdependent, each input feature selected represents a group of correlated grammatical phenomena. For example, in many languages, only verbs are considered to have tense. Therefore, if a word is annotated as "tense = past", the part-of-speech annotation is redundant. Table 8 presents the selection of features "preferred" by *Oliver Twist*, as compared to the remainder of the corpus.

Table 8. Selection of input features "preferred" in *Oliver Twist*.

#	Feature	Frequency, *Oliver*	Frequency (Mean of Corpus)	Frequency Rank	Z-Score
1A	Number is singular, parent precedes	0.176	0.148	1	2.98
2A	Parent's own parent follows	0.124	0.104	1	2.59
3A	Parent is singular, parent's DD = 2	0.071	0.061	3	2.04
4A	Article, parent is singular noun	0.079	0.064	5	1.72
5A	Parent's dependency label is "object"	0.065	0.058	3	1.43

A few examples will help to illustrate the phenomena underlying these values. The first feature is grammatically transparent. This sentence from *Oliver Twist* has two examples: "They talked of **hope** and **comfort**". The two bold-faced nouns are annotated with feature #1; obviously singular, they are preceded by their dependency parent, *talked.* Although this dependency—a noun upon a verb—is the most frequent structure annotated with feature

#1 (a rate of 0.403 in *Oliver Twist*), a noun dependent on a preceding noun is also common (0.195), as in the following: "... an unwonted allowance of **beer**...". Here, the singular *beer* is dependent on *allowance.* One should also be aware that on rare occasions (0.018), the word annotated with feature #1 is itself a verb instead of the more usual noun (0.871) or pronoun (0.107): "I never knew how bad she **was**...". In this sequence, *was* is considered the head word of the indirect question clause *how bad she was;* UD grammar considers that the clause is dependent on the verb *knew*, which precedes it.

When interpreting a feature such as #1, which reflects more than a single grammatical value, it is a good idea to establish the relative contribution made by the components to the combined frequency. For example, both elements of feature #1 are more frequent in *Oliver Twist* than in the remainder of the corpus. Words annotated with *singular*: *OT* = 0.3171 and corpus = 0.308; words annotated with *parent precedes*: *OT* = 0.3394 and corpus = 0.3158. At the same time, the frequency of the combination of the two components (*OT* 0.17) is much higher than would be expected based on the parts (0.107). Thus, in a study of the style of the Dickens work, both feature #1 and its parts would be worthy of further analysis.

Feature #2 reflects a deep level of sentence structure since a word's annotation is based on its dependency parent and "grandparent". For example, in the sentence "If he **could** have known that... perhaps he would have cried the louder", *known*, the head word of the conditional clause, is the parent of the annotated word *could; cried*, the main verb of the sentence, is the parent of *known.* Since *cried* follows *known*, feature #2 is appropriately applied to *could*.

Feature #3, based simply on the dependency distance of the parent of the annotated word, should need no illustration. It is worth noting that, as for feature #1, both components of feature #3 are elements "preferred" by *Oliver Twist.* The frequency of words annotated with "parent is singular" is *OT* = 0.401 and corpus = 0.376; for "DD of parent is 2", *OT* = 0.134 and corpus = 0.129. Based on the individual frequencies, the expected rate of occurrence for the combination is 0.0537 for *Oliver Twist.* The actual rate is almost one-third larger.

Feature #4 is simple but is an example of the importance that very elementary syntactic structures can have in drawing stylometric distinctions. Essentially, this feature indicates the number of singular nouns that are modified by an article, either definite (*the)* or indefinite (*a/an*). Again, it is informative to analyze a compound feature according to its components. In this instance, *Oliver Twist* displays a preference for singular nouns (*OT*: 0.1534; corpus: 0.1404), but much of the distinctive force of feature #4 cannot be explained in terms of the frequencies of singular nouns. When feature #4 is controlled for the number of such words, the ratio of articles per singular noun is *OT* = 0.5182 and corpus = 0.4558. Clearly, a high frequency of articles is a stylistic characteristic of *Oliver Twist*.

Feature #5 is grammatically more complex. To interpret it, one must be familiar with two important aspects of dependency grammar: (1) verb valency and (2) functional syntax. According to dependency grammar in general, sentences are structures "built" according to the "requirements" of its component verbs. The primary requirement is the valency of a verb. Simply, "valency" in the appropriate sense is the number of dependents that are necessary to make a verb syntactically and semantically "correct". Such necessary components are called "arguments" of a given verb. For example, consideration of the sentence "Caesar died in Rome" shows that *die* here has a valency of one. If we subtract the dependencies, we produce "*died in Rome" and "Caesar died". Only the second is acceptable and indicates that *died* has a valency of one. A bivalent verb can be seen in "Brutus killed Caesar in Rome". *Kill* requires both *Brutus* and *Caesar*, but not *in Rome.* The argument of a monovalent verb is called the verb's *subject*; for bivalent verbs such as *kill*, one dependency is labeled as the verb's *subject*, the other as its *object*.

"Functional syntax" refers to the theory according to which dependencies are labeled primarily according to the role that they play with respect to their parent word. Less importance is given to the internal characteristics of the dependency. Consider the sentence "She put money in the bank". The verb *put* is trivalent since it requires a subject (*she*), an object (*money*), and a third expression (*in the bank*) indicating a place/goal. This third expression in the case of trivalent verbs is also often called an object (or second object). The primacy of function is evident here in that the fact that *in the bank* is a prepositional phrase does not affect its dependency label. Its internal structure is irrelevant. Compare the sentence "She put money there". The dependency relationships in this version would be the same as in the first example. Although *there* in the second sentence is an adverb, it, like *in the bank*, is correctly annotated as object. The two expressions serve the same function with respect to *put*, and therefore receive the same relationship annotation.

The functional emphasis shown by dependency grammar reduces the number of dependency relation labels and, at the same time, groups a range of internally different phenomena in the same category. A few examples may help to clarify the phenomena that are reflected by feature #5. Under the label "object", UDPipe includes primarily nouns and adjectives. An example of a noun object is "Give it a **little** gruel...", where *little* is the annotated word and *gruel* its parent; *gruel* is the second argument (here a direct object) of *give.* One of Dickens's most famous sentences provides an example of an adjective in the *object* function: "'Please, sir', replied Oliver, 'I want **some** more'". Here, *more* is the direct object of *want*, and *some* (the annotated word) is a dependent modifier of the adjective *more*.

The word characterized by feature #5, in distinction to its *object* dependency parent, can also represent varying grammatical phenomena. To take only verbs, we find examples like "I shall take a early opportunity of **mentioning** it ...". The annotated *mentioning* is a verb in its gerund form and is dependent on *opportunity*, the direct object of *take*. A different phenomenon is represented by "Bumble wiped from his forehead the perspiration which his walk had **engendered**...". The annotated *engendered* is the verb of the relative clause describing (and therefore dependent on) *perspiration*. UD grammar considers the verb the head of a relative clause, and therefore, *engendered* is the direct dependent of *perspiration*, which in turn is the direct object of the main verb *wiped.* Yet another difference is apparent in "The boy had no friends to **care** for...", where the annotated *care* is part of an explanatory infinitive structure which specifies the meaning of *friends*, the direct object of *had*.

This brief discussion of the dependency relationship *object* should make clear that the dependency labels are the most complicated annotation in the morphosyntactic data set created by UDPipe. However, since their complexity arises primarily from the grouping together of different grammatical "types" according to their grammatical "function", the interpretation of the relevant input features is time-consuming rather than conceptually difficult.

In addition to input features that are strongly "preferred" in *Oliver Twist*, there are others that are sharply "avoided". We will look only at three of the most important, as given in Table 9.

Table 9. Input features "avoided" in *Oliver Twist*.

#	Feature	Frequency, *Oliver*	Frequency (Mean of Corpus)	Frequency Rank	Z-Score
1B	Parent is an infinitive verb	0.076	0.10	43	−1.71
2B	Personal pronoun	0.085	0.114	43	−1.66
3B	Plural	0.041	0.051	42	−1.22

Feature #1B represents dependencies of the infinitive form of the verb. The English infinitive is morphologically the same as the dictionary lemma. It primarily occurs in one of two configurations. An infinitive can be "introduced" by the particle *to*, as in the following examples: "I have come out myself **to** take **him there**"; and "... the parish would like **him to** learn a right pleasant **trade** ...". It is apparent that *to* plus the infinitive has a wide range of syntactic functions. In the first example, *to take* expresses the purpose for which the action of the main verb was undertaken. The infinitive phrase can be deleted from the sentence without making it ungrammatical. In contrast, the syntax (and semantics) of *like* in the second example requires an object; *to learn* performs that necessary function and cannot be omitted without producing incorrect grammar. Note that in both example sentences, the infinitives have three words directly dependent on them (marked in bold). All of these words, then, would be correctly coded with feature 1B.

The second common configuration for the English infinitive is to be "introduced" by certain modal and auxiliary verbs. Examples occur in the following passages: "**Do I** understand that he **asked** for more. . .?"; ". . . [I]t **may** be as you **say**. . .". In the first example, the infinitive is *understand* which forms a verb phrase with the auxiliary *do.* In the second, the infinitive is *be*, a usage sometimes called complementary since the infinitive is necessary to complete the structure implied by a modal (here *may*). The bold-faced words in each example again indicate direct dependents of the infinitives, as required for feature 1B.

From these examples and the brief discussion, it should be clear that English infinitives in their most frequent structures *cannot* appear without at least one direct dependent. According to the rules of UD, the particle *to* is considered a dependent of the infinitive it precedes. Likewise, auxiliaries and modals such as those in our second set of examples are annotated as immediate dependents of the infinitives with which they are associated. It will not be surprising, then, to learn that, on average, each infinitive has more direct dependents (*OT*: 2.516; corpus: 2.795) than finite verbs (*OT*: 1.752; corpus: 1.628). At the same time, these numbers indicate a sharp difference between *Oliver Twist* and the rest of the corpus with respect to the complexity of infinitive clauses. The increase in the average number of dependencies from finite verbs to infinitives is much smaller than we might expect, given that infinitives "automatically" come with at least one dependent word: dependency per word increases by 1.167 for the corpus, but just by 0.764 for *Oliver Twist.* Thus, measured by the number of dependencies, infinitive structures in *OT* are less complex than we would expect based on finite structures in the same novel.

The grammatical categories reflected in features 2B and 3B are self-evident and require no examples. We only note that the relative avoidance of personal pronouns (*I*, *you*, *she*, *he*, *it*, etc.) in *Oliver Twist* is no doubt associated with the same novel's relative preference for common and proper nouns: *OT* = 0.245 and corpus = 0.218. Relevant for the interpretation of feature 3B is the fact that the frequency of all words annotated with grammatical number—nouns, pronouns, verbs and a few determiners (*this*/*these*, etc.)—is lower for *Oliver Twist* than for the remainder of the corpus (*OT* = 0.358; corpus = 0.364). This difference, however, only partly explains *OT's* relative avoidance of feature 3B. In fact, the distribution of grammatical number within this subset of relevant parts of speech leans strongly toward the singular as compared to the rest of the corpus (*OT*: singular = 0.884 and plural = 0.115; corpus: singular = 0.857 and plural = 0.142).

As noted above, comparison of the classification results in Tables 6 and 7 reveals that for every corpus tested, the signal identifying each individual novel is much more discernable than the authorial signal. While truly understanding this phenomenon—the coexistence of "local" variability and "authorial" style—will no doubt require many years of intensive study, stylometrics at the morphosyntactic level can offer valuable data bearing on this issue.

A detailed discussion is beyond the limits of this paper, but a single straightforward example will serve as a useful illustration. The accuracy values given in Table 7 are averages that encompass a great deal of variation. The authorial signal for some writers in each corpus was very weak; other authors were comparatively quite easy to distinguish. To take the corpus of English language novels, the most distinguishable author, with the highest mean accuracy of classification, was E. M. Forster. Subsamples of Forster's novels were attributed to the author with an accuracy of about 85%. (Forster's works in the corpus are *Where Angels Fear to Tread* (1903), *A Room with a View* (1908) and *Howards End* (1910)). At the other extreme, accuracy for the works of Vernon Lee was generally less than 1%. (Vernon Lee was a *nom de plume* for the writer Violet Paget. The relevant works in the corpus are *The Countess of Albany* (1884), *Miss Brown* (1884) and *Penelope Brandling* (1903).) The authorial signal for Charles Dickens, whose *Oliver Twist* was the focus of our discussion of input features, was squarely in the middle at about 50% (in addition to *Oliver Twist* (1839), the corpus also contains Dickens's *Bleak House* (1853) and *Great Expectations* (1861)).

As one might expect, the algorithm's success at discriminating authorial signals is to some degree correlated with stylometric consistency within the works of each author. In other words, authors whose works show greater variability in the values of the input features tend to be more difficult to correctly attribute an authorial signature to. Table 10 gives a distribution summary for the "intra-author" standard deviations of each morphosyntactic input feature.

Table 10. Summary of standard deviations of input features for selected authors.

	Min.	1st Quart.	Median	Mean	3rd Quart.	Max.
Forster	0.00005	0.0014	0.0027	0.0037	0.0050	0.0174
Dickens	0.00027	0.0040	0.0066	0.0072	0.0099	0.0248
Lee	0.00037	0.0088	0.0130	0.0149	0.0185	0.0733

Each value in Table 10 is based on the standard deviation of the three works of each author for each of the 653 input features used in the English corpus. It is evident that the works of Lee are much less consistent with each other than the works of authors with better classification results. Lee's weak signal is not unexpected, given that the mean standard deviation in her works is more than four times larger than Forster's, while Lee's median value approaches five times Forster's!

That inconsistency within a class is associated with a high noise-to-information ratio and with difficulty in discrimination, which will surprise few people familiar with classification experiments. On the other hand, extensive use of morphosyntactic annotation is rare in stylometric studies, and the data reported in Table 10 suggest that such information would indeed be useful in exploring stylometric variation for an individual author. In addition, because it preserves a relatively high amount of information even in short texts (see the 500-word samples in Tables 6 and 7), a morphosyntactic approach may even be effective in describing stylometric variability within a single work or a single chapter of a work.

We will conclude our investigation into morphosyntactic stylometry by returning to the work of Charles Dickens. Taken together, the works of Dickens in our corpus produce, as mentioned above, a moderate authorial signal. A few of the important input features through which the logistic regression model distinguishes "Dickens" from the other authors in the corpus are given in Table 11.

Table 11. Selected input features preferred or avoided by the class "Dickens".

#	Feature	Mean Frequency of Dickens's Works in Corpus	Mean Frequency of Remaining Corpus	Intra-Author Standard Deviation	Z-Score (Dickens's Works in Corpus)
1C	Parent precedes	0.336	0.314	0.0047	1.66
2C	Parent is a verb with DD > 2	0.136	0.107	0.0091	1.58
3C	Parent is a verb and head of an adverbial clause	0.069	0.055	0.0065	1.41
4C	Adjective	0.064	0.071	0.0056	−0.93
5C	Parent is sentence root	0.183	0.218	0.0082	−0.98

In Table 11, the intra-author standard deviation indicates that for each selected feature, the frequencies in the three works of Dickens in the corpus are quite close to each other. With regard to the z-scores, a distinction is noticeable between these and the scores in

Tables 6 and 7. The values in Table 11 indicate a selection of features with smaller differences between the class of interest (here "Dickens") and the corpus mean. This is not an accident of selection, since relatively large z-scores are less frequent for the mean of Dickens's three relevant works than for *Oliver Twist.* For example, for *OT*, 52 input features had a z-score with a magnitude greater than 1.5 (positive or negative). The corresponding count for the three-work mean is 9.

To turn to the details of the input features, feature 3C needs no further elaboration; any word that comes after its dependency parent in the linear order of the sentence is encoded with this feature. Feature 2C should likewise be self-explanatory at this point. The frequency of feature 4C is based on a simple count of parts of speech: a relative avoidance of adjectives is a shared characteristic of the three Dickens texts.

Features 3C and 5C contain types of dependency relationships and therefore require some explanation. In the UD annotation scheme, a sharp distinction is made between words that function as arguments to a verb (see above) and words that do not. Words that are not arguments are optional in the sense that they can be omitted without rendering the sentence ungrammatical (or nonsensical). *Adverbial* is the label for the most important class of "optional" words. If the word with this function is a verb, it is labeled as an *adverbial clause.* Two examples will point to the many possible ways that a verb can function as an adverbial for another clause: (1) "If **he could have** known that he was an **orphan**, ... perhaps he would have cried the louder"; (2) "But he hadn't, **because nobody had** taught **him**". In sentence 1, *known* is the head verb of the conditional clause that is subordinate to the main verb *cried.* In sentence 2, *taught* is the head of a causal clause, itself dependent on *hadn't.* In both sentences, words annotated with "parent is head of an adverbial clause" are highlighted in bold.

The last of our exemplary "Dickensian" input features, feature 5C, indicates the frequency of sentence main clauses. Generally, the root of a sentence is a label given to the main verb, but a peculiarity of UD in this regard may be illustrated by the following example: "**Boys is wery** obstinit...". In equational structures such as this one, where the subject is "linked" with a predicate nominal (here *obstinit*) by a copula verb (*be* and similar verbs), UD grammar considers the predicate nominal, and not the verb, to be the head of the clause. Thus, the bold-faced words in the example are annotated as dependents of the adjective. As a result of this protocol, a not insignificant portion of sentence *roots* in the UD scheme are nouns, pronouns and adjectives.

The final step in our discussion of the "Dickensian" authorial signal is to give a very few examples of input features that weaken that signal. In particular, this is a selection of features for which the values are relatively diverse across the three Dickens novels in the corpus. Details are given in Table 12.

Table 12. Selected input features where frequency variability weakens the "Dickens" signal. *BH* = *Bleak House*, *GE* = *Great Expectations* and *OT* = *Oliver Twist*.

#	Feature	Frequency (Dickens)	Frequency (Remainder of Corpus)	Intra-Author s.d.	Frequency Rank
1D	Personal pronoun	*BH*: 0.117 *GE*: 0.134 *OT*: 0.085	0.113	0.020	*BH*: 20 *GE*: 7 *OT*: 43
2D	Parent is singular	*BH*: 0.384 *GE*: 0.355 *OT*: 0.401	0.376	0.0188	*BH*: 20 *GE*: 35 *OT*: 8
3D	Parent is past indicative verb	*BH*: 0.137 *GE*: 0.182 *OT*: 0.150	0.136	0.0188	*BH*: 24 *GE*: 3 *OT*: 13
4D	Parent is verb	*BH*: 0.414 *GE*: 0.449 *OT*: 0.409	0.403	0.0179	*BH*: 19 *GE*: 3 *OT*: 6

The morphosyntactic phenomena underlying these features should by now be clear and examples unnecessary. In fact, we have already looked closely at feature 1D, which appeared as feature 2B in Table 9. There, the avoidance of personal pronouns was identified as a distinguishing characteristic of *Oliver Twist.* Here, we see that this characteristic is not shared by the other two relevant works, in each of which personal pronouns are more frequent than in the corpus mean. Thus, unsurprisingly, the same morphosyntactic feature can be informative at one level of classification (e.g., novel by novel) and simultaneously increase noise at another (e.g., author by author).

The features in Table 12 disrupt the authorial signal for "Dickens" because of how much the frequencies vary among the three Dickens novels in the corpus. Comparison of the "author-internal" standard deviations in Tables 11 and 12 shows that the dispersal of the features in Table 12 is 1.9 to 4.2 times greater than in Table 11. Works with such a range of frequencies for any given input feature will hinder the detection of an authorial signal. At the same time, it is important to realize that a tight "grouping" of frequencies for a feature is not in itself enough to make that feature informative for classification. For example, there are many morphosyntactic input features in our set for which the "Dickensian" standard deviation is quite small but whose frequencies are very close to the corpus mean. Two such features are "part-of-speech is verb" and "part-of-speech is a preposition and parent is a noun". For the first of these, the mean frequency for the three Dickens novels is 0.134 (sd = 0.0043) and the corpus mean is 0.130. For the second, a feature that essentially reflects the number of prepositional phrases in the texts, the frequency for Dickens's works is 0.079 (sd = 0.0018), while the corpus mean is 0.078. Features with frequencies in this pattern are generally not valuable for logistic regression. From a stylometric or stylistic point of view, however, it may be just as interesting to know where Dickens's morphosyntactic characteristics adhere to the norm as where they depart sharply from it.

6. Conclusions

This paper has presented arguments for the potential value of morphosyntactic annotation for stylometric analysis. It has demonstrated that UDPipe parsers currently available for many languages produce annotations whose inevitable errors do not seriously undermine the stylometric usefulness of this information, judged by accuracy in classification experiments. Based on the assumption that morphosyntactic characteristics are not closely dependent on the specific subject matter of the target texts, the input features described in this study are to a significant degree topic agnostic. Further, this work has explored the advantage offered by morphosyntax in terms of stylometric interpretability. The input features used here are, for the most part, made up of grammatical concepts likely to be familiar to anyone seriously investigating the style of a literary work or author. Admittedly, the concepts underlying dependency grammar may be new to many investigators, but the syntax of natural languages is itself a complex structure. Dependency grammar, assuming no "hidden" structures, reflects this complexity in a fairly straightforward way. In view of the demonstrated advantages of morphosyntactic information, it seems clear that it should have a larger role in stylometric scholarship.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Mosteller, F.; Wallace, D.L. Inference in an Authorship Problem. *J. Am. Stat. Assoc.* **1963**, *58*, 275–309. [CrossRef]

2. Koppel, M.; Schler, J.; Bonchek-Dokow, E. Measuring Differentiability: Unmasking Pseudonymous Authors. *J. Mach. Learn. Res.* **2007**, *8*, 1261–1276.
3. Coyotl-Morales, R.M.; Villaseñor-Pineda, L.; Montes-y-Gómez, M.; Rosso, P. Authorship Attribution Using Word Sequences. In *Progress in Pattern Recognition, Image Analysis and Applications*; Martínez-Trinidad, J.F., Carrasco Ochoa, J.A., Kittler, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 844–853.
4. Rochon, E.; Saffran, E.M.; Berndt, R.S.; Schwartz, M.F. Quantitative Analysis of Aphasic Sentence Production: Further Development and New Data. *Brain Lang.* **2000**, *72*, 193–218. [CrossRef]
5. Kestemont, M. Function Words in Authorship Attribution from Black Magic to Theory. In Proceedings of the 3rd Workshop on Computational Linguistics for Literature, Gothenburg, Sweden, 27 April 2014; pp. 59–66.
6. Koppel, M.; Schler, J.; Argamon, S. Computational Methods in Authorship Attribution. *J. Am. Soc. Inf. Sci. Technol.* **2009**, *60*, 9–26. [CrossRef]
7. Ranaldi, L.; Pucci, G. Knowing Knowledge: Epistemological Study of Knowledge in Transformers. *Appl. Sci.* **2023**, *13*, 677. [CrossRef]
8. Revesz, P.Z. A vowel harmony testing algorithm to aid in ancient script decipherment. In Proceedings of the 24th International Conf. on Circuits, Systems, Communications and Computers, Chania, Greece, 19–22 July 2020; IEEE Press: New York, NY, USA, 2020; pp. 35–38.
9. VanOrsdale, J.; Chauhan, J.; Potlapally, S.V.; Chanamolu, S.; Kasara, S.P.R.; Revesz, P.Z. Measuring vowel harmony within Hungarian, the Indus Valley Script language, Spanish and Turkish using ERGM. In Proceedings of the 26th International Database Engineered Application Symposium, Budapest, Hungary, 13 September 2022; pp. 171–174.
10. Burke, M. Stylistics: From classical rhetoric to cognitive neuroscience. In *The Routledge Handbook of Stylistics*; Burke, M., Ed.; Routledge handbooks in English Language Studies; Routledge, Taylor & Francis Group: London, UK; New York, NY, USA, 2014.
11. Eder, M.; Górski, R.L. Stylistic Fingerprints, POS-tags, and Inflected Languages: A Case Study in Polish. *J. Quant. Linguist.* **2022**, *30*, 86–103. [CrossRef]
12. Nivre, J.; De Marneffe, M.C.; Ginter, F.; Goldberg, Y.; Hajic, J.; Manning, C.D.; McDonald, R.; Petrov, S.; Pyysalo, S.; Silveira, N.; et al. Universal dependencies v1: A multilingual treebank collection. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia, 23–28 May 2016; pp. 1659–1666.
13. De Marneffe, M.C.; Manning, C.D.; Nivre, J.; Zeman, D. Universal dependencies. *Comput. Linguist.* **2021**, *47*, 255–308. [CrossRef]
14. Wijffels, J. Udpipe: Tokenization, Parts of Speech Tagging, Lemmatization and Dependency Parsing with the 'UDPipe' 'NLP' Toolkit. 2019. Available online: https://CRAN.R-project.org/package=udpipe (accessed on 1 January 2024).
15. Lui, H. Dependency distance as a metric of language comprehension difficulty. *J. Cogn. Sci.* **2008**, *9*, 159–191.
16. Chen, R.; Deng, S.; Liu, H. Syntactic complexity of different test types: From the perspective of dependency distance both linearly and hierarchically. *J. Quant. Linguist.* **2021**, *29*, 510–540. [CrossRef]
17. Ferrer-i-Cancho, R.; Gómez-Rodríguez, C.; Esteban, J.L.; Alemany-Puig, L. Optimality of syntactic dependency systems. *Phys. Rev. E* **2022**, *105*, 014308. [CrossRef]
18. Chen, X.; Gerdes, K. Classifying languages by dependency structure: Typologies of delexicalized universal dependency treebanks. In Proceedings of the Fourth International Conference On Dependency Linguistics, Pisa, Italy, 18–20 September 2017; pp. 54–63.
19. Eder, M. Short Samples in Authorship Attribution: A New Approach. In Proceedings of the Digital Humanities 2017, Conference Abstracts, 8–11 August 2017; McGill University: Montreal, QC, Canada, 2017; pp. 221–224. Available online: https://dh2017.adho.org/abstracts/341/341.pdf (accessed on 1 January 2024).
20. Hay, J.; Doan, B.L.; Popineau, F.; Elhara, O.A. Representation Learning of Writing Style. In Proceedings of the Sixth Workshop on Noisy User-Generated Text (W-NUT 2020), Online, 19 November 2020; Available online: https://aclanthology.org/2020.wnut-1.30 (accessed on 1 January 2024).
21. Wegmann, A.; Schraagen, M.; Nguyen, D. Same Author or Just Same Topic? Towards Content-Independent Style Representations. *arXiv* **2022**, arXiv:2204.04907.
22. Patel, A.; Rao, D.; Callison-Burch, C. Learning Interpretable Style Embeddings via Prompting LLMs. *arXiv* **2023**, arXiv:2305.12696.
23. Fan, R.E.; Chang, K.W.; Hsieh, C.J.; Wang, X.R.; Lin, C.J. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.* **2008**, *9*, 1871–1874.
24. Helleputte, T. LiblineaR: Linear Predictive Models Based On The Liblinear C/C++ Library. R Package Version 2.10-12. Available online: https://www.csie.ntu.edu.tw/~cjlin/liblinear/ (accessed on 1 January 2024).
25. Simon, R. Resampling strategies for model assessment and selection. In *Fundamentals of Data Mining in Genomics and Proteomics*; Dubitzky, W., Granzow, M., Berrar, D., Eds.; Springer: Boston, MA, USA, 2007; pp. 173–186.
26. Eder, M. Does size matter? Authorship attribution, short samples, big problem. In *Digital Humanities 2010: Conference Abstracts*; King's College London: London, UK, 2015; pp. 132–135.
27. Eder, M. Does size matter? Authorship attribution, small samples, big problem. *Digit. Scholarsh. Humanit.* **2015**, *30*, 167–182. [CrossRef]

information

Article

A Benchmark Dataset to Distinguish Human-Written and Machine-Generated Scientific Papers †

Mohamed Hesham Ibrahim Abdalla ‡, Simon Malberg ‡, Daryna Dementieva *, Edoardo Mosca and Georg Groh *

School of Computation, Information and Technology, Technical University of Munich, 80333 Munich, Germany; ge96rac@mytum.de (M.H.I.A.); simon.malberg@tum.de (S.M.); edoardo.mosca@tum.de (E.M.)
* Correspondence: daryna.dementieva@tum.de (D.D.); grohg@in.tum.de (G.G.)
† This paper is a substantially extended and revised version of research published in Mosca E.; Abdalla M.H.I.; Basso P.; Musumeci M.; and Groh G. Distinguishing Fact from Fiction: A Benchmark Dataset for Identifying Machine-Generated Scientific Papers in the LLM Era. In Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023), pages 190–207, Toronto, Canada. Association for Computational Linguistics.
‡ These authors contributed equally to this work.

Abstract: As generative NLP can now produce content nearly indistinguishable from human writing, it is becoming difficult to identify genuine research contributions in academic writing and scientific publications. Moreover, information in machine-generated text can be factually wrong or even entirely fabricated. In this work, we introduce a novel benchmark dataset containing human-written and machine-generated scientific papers from SCIgen, GPT-2, GPT-3, ChatGPT, and Galactica, as well as papers co-created by humans and ChatGPT. We also experiment with several types of classifiers—linguistic-based and transformer-based—for detecting the authorship of scientific text. A strong focus is put on generalization capabilities and explainability to highlight the strengths and weaknesses of these detectors. Our work makes an important step towards creating more robust methods for distinguishing between human-written and machine-generated scientific papers, ultimately ensuring the integrity of scientific literature.

Keywords: text generation; large language models; machine-generated text detection

Citation: Abdalla, M.H.I.; Malberg, S.; Dementieva, D.; Mosca, E.; Groh, G. A Benchmark Dataset to Distinguish Human-Written and Machine-Generated Scientific Papers. *Information* **2023**, *14*, 522. https://doi.org/10.3390/info14100522

Academic Editor: Peter Revesz

Received: 31 July 2023
Revised: 13 September 2023
Accepted: 14 September 2023
Published: 26 September 2023

1. Introduction

Generative *Natural Language Processing* (NLP) systems—often based on *Large Language Models* (LLMs) [1–3]—have experienced significant advancements in recent years, with state-of-the-art algorithms generating content that is almost indistinguishable from human-written text [1,4–7]. This progress has led to numerous applications in various fields, such as chatbots [8], automated content generation [9], and even summarization tools [10]. However, these advancements also raise concerns regarding the integrity and authenticity of academic writing and scientific publications [11,12].

It is indeed increasingly difficult to differentiate genuine research contributions from artificially generated content. Moreover, we are at an increased risk of including factually incorrect or entirely fabricated information [13,14]. Reliably identifying machine-generated scientific publications becomes, thus, crucial to maintaining the credibility of scientific literature and fostering trust among researchers.

This work introduces a novel benchmark to address this issue. Our contribution—also briefly sketched in Figure 1—can be summarized as follows:

(1) We present a dataset comprising human-written and machine-generated scientific documents from various sources: SCIgen [15], GPT-2 [4], GPT-3 [1], ChatGPT [8], and Galactica [16]. We also include a set of human–machine co-created documents resembling scientific documents with both human-written and machine-paraphrased

texts. Each document includes an *abstract*, *introduction*, and *conclusion* in a machine-readable format. While real titles were used to generate articles for the dataset, there is no title intersection between real and machine-generated papers in our dataset.

(2) We experiment with several classifiers—bag-of-words-based classifiers, RoBERTa [17], Galactica [16], GPT-3 [1], DetectGPT [18], ChatGPT [8], and a proposed novel classifier learning features using an LLM and Random Forest [19]—assessing their performance in differentiating between human-written and machine-generated content. We also assess each classifier's generalization capabilities on out-of-domain data and human–machine co-created papers to obtain a more accurate estimate of the likely real-world performance of the different classifiers.

(3) We explore explainability insights from different classifiers ranging from word-level explanations to more abstract concepts to identify typical differences between human-written and machine-generated scientific papers.

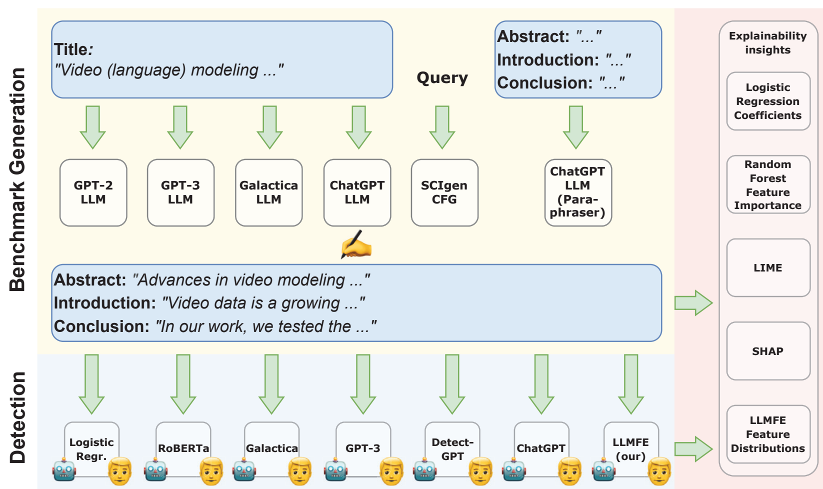


Figure 1. This work's overview. Six methods are used to machine-generate papers, which are then mixed with human-written ones to create our benchmark dataset. Seven models are then tested as baselines to identify the authorship of a given output.

We release our benchmark dataset, baseline models, and testing code to the public to promote further research and aid the development of more robust detection methods. (https://huggingface.co/datasets/tum-nlp/IDMGSP) (accessed on 31 July 2023). This work extends a previously published study [20].

2. Related Work

2.1. Machine-Generated Text Detection Benchmarks

Since the significant improvement of text generation models, the potential danger and harm of machine-generated text has been acknowledged by NLP researchers. For this reason, existing generations of generative models have been tested to create texts in various domains to compile human-written vs. machine-generated benchmarks.

One of the first datasets and models to detect neural generated texts in a news domain was Grover [7]. The Grover model for neural news generation was based on GPT-2 [4] and was used to create a benchmark for neural news detection. In addition, for the news domain, a dataset for automatic detection of machine-generated news headlines was created [21]. The machine-generated headlines were also created with GPT-2. Beyond fake news, the detection of generated scientific articles received attention as well, leading to the first task benchmarks introduced in [22].

With the increasing capabilities of LLMs, new benchmarks appeared recently, covering several neural text generators and domains. In [23], the M4 (multi-generator, multi-domain, and multi-lingual) dataset was presented. It covers various kinds of topics—Wikipedia articles, question-answering posts, news, and social posts—in six languages. In the MGTBench benchmark [24], LLMs were evaluated on several different question-answering datasets. Finally, the DeepfakeTextDetect dataset [25] covers news article writing, story generation, scientific writing, argument generation, and question-answering.

2.2. Scientific Publication Corpora: Human and Machine-Generated

The ACL Anthology (https://aclanthology.org) (accessed on 24 April 2023). Ref. [26] and arXiv [27] are widely used resources for accessing scientific texts and their associated metadata. However, these databases do not provide structured text for scientific documents, necessitating the use of PDF parsers and other tools to extract text and resolve references. Several efforts have been made to develop structured text databases for scientific documents [28–30].

Despite progress in generating text, machine-generated datasets for scientific literature remain limited. A recent study by Kashnitsky et al. [31] compiled a dataset including shortened, summarized, and paraphrased paper abstracts and excerpts, as well as text generated by GPT-3 [1] and GPT-Neo [32]. The dataset lists retracted papers as machine-generated, which may not always be accurate, and only includes excerpts or abstracts of the papers.

Liyanage et al. [22] proposed an alternative approach, in which they generated papers using GPT-2 [4] and arXiv-NLP (https://huggingface.co/lysandre/arxiv-nlp) (accessed on 24 April 2023). However, their dataset was limited to only 200 samples, which were restricted to the fields of Artificial Intelligence and Computation and Language.

2.3. Generative NLP for Scientific Articles

Generative NLP for scientific publications has evolved significantly in recent years. Early methods, such as SCIgen [15], used *Context-Free-Grammars* (CFG) to fabricate computer science publications. These often contain nonsensical outputs due to CFG's limited capacity for generating coherent text.

With the advent of attention, transformers [33] and LLMs [1] have paved the way for more sophisticated models capable of generating higher-quality scientific content. Some known (both opensourced and closed) LLMs—such as GPT-3 [1], ChatGPT [8], Bloom [2], LLaMa-2 [6], and PaLM-2 [34]—are built for general purposes. Others, instead, are domain-specific and specialized for generating scientific literature. Popular examples in this category are SciBERT [35] and Galactica [16].

Both general and domain-specific models have shown outstanding results in various scientific tasks, demonstrating their potential to generate coherent and contextually relevant scientific text. Consequentially, the same technology has been applied to other domains, including writing news articles [7], producing learning material [36], and creative writing [37]. Moreover, in education, the usage of advanced LLMs showed already promising results in providing "live" help in the teaching process [38]. For such use cases, it is important to develop trustworthy machine-generation technologies, able to provide both factually correct information as well as display fluency in communication with the users.

2.4. Detection of Machine-Generated Text

The ability to automatically generate convincing content has motivated researchers to work on its automatic detection, especially given its potential implications for various domains.

Several approaches to detecting machine-generated text have emerged, employing various techniques. In [39], a survey of the methods for machine-generated text detection was presented. One solution is utilizing hand-crafted features [40]. In addition, linguistic-based and bag-of-words features can be quite powerful and well-explainable baselines [41].

The topology of attention masks was proven to be one of the efficient methods to detect neural-generated texts in [42]. Finally, neural features in combination with supervised models can be trained to distinguish between human and machine-generated content [41,43,44].

Alternative approaches explore using the probability distribution of the generative model itself [18] or watermarking machine-generated text to facilitate detection [45].

2.5. Detection of Machine-Generated Scientific Publications

As we have seen in Section 2.4, several general-purpose solutions exist aiming to detect NLP-generated text. The detection of automatically generated scientific publications, however, is an emerging subarea of research with a large potential for improvement.

Previous approaches have primarily focused on identifying text generated by SCIgen [15] using hand-crafted features [46,47], nearest neighbor classifiers [48], and grammar-based detectors [49]. More recent studies have shown promising results in detecting LLM-generated papers using SciBERT [50], DistilBERT [51], and other transformer-based models [22,52]. Nonetheless, these approaches have mostly been tested only on abstracts or a substantially limited set of paper domains.

With the appearance of ChatGPT [8], several studies were dedicated to evaluating how good this model can be in generating scientific papers. In [53], it was shown that human annotators are incapable of identifying ChatGPT-generated papers. Since ChatGPT can not only be used to generate papers from scratch but also to paraphrase them, a method to identify the polish-ratio of ChatGPT in a piece of text was proposed in [54].

In the end, we can see the necessity for an explainable and robust detector able to detect machine-generated and edited articles from the most recent LLMs. With this work, we are aiming to make a step towards the creation of such automated detectors.

3. Benchmark Dataset

In this section, we delve into the construction of our benchmark dataset, which comprises human-written, machine-generated, and human–machine co-created scientific papers. Often, for simplicity, we refer to these groups as ***real***, ***fake***, and ***co-created***, respectively. In Section 3.1, we elaborate on the process we followed to extract data from the PDF documents of real papers. In Section 3.2, we describe our prompting pipelines and how we utilized various generators to produce fake scientific papers. In Section 3.3, we explain our approach to generating human–machine co-created papers.

Table 1 offers an overview of our dataset, including sources and numbers of samples and tokens.

Table 1. Data sources included in our dataset and their respective sizes.

Source	Quantity	Tokens
arXiv parsing 1 (real)	12 k	13.4 M
arXiv parsing 2 (real)	4 k	3.2 M
SCIgen (fake)	3 k	1.8 M
GPT-2 (fake)	3 k	2.9 M
Galactica (fake)	3 k	2.0 M
ChatGPT (fake)	3 k	1.2 M
GPT-3 (fake)	1 k	0.5 M
ChatGPT (paraphrased real)	4 k	3.5 M
Total real (extraction)	16 k	16.6 M
Total fake (generators)	13 k	8.4 M
Total co-created (paraphrased)	4 k	3.5 M
Total	33 k	28.5 M

3.1. Real Papers Collection

To collect human-written—or *real*—scientific papers for our dataset, we source them from the arXiv dataset [27] hosted on Kaggle (https://www.kaggle.com/datasets/Cornell-

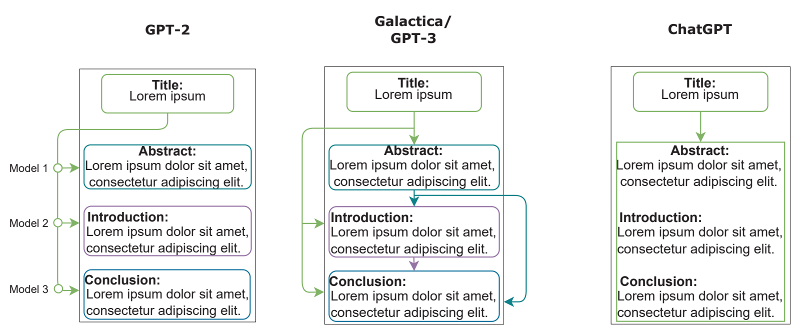
University/arxiv (accessed on 24 April 2023)). We exclude scientific papers published after ChatGPT (after November 2022) to avoid machine-generated papers leaking into our *real* dataset. While it is still possible that some of the remaining papers were machine-generated, we deem this to be highly unlikely and only affect a negligibly small number of papers, if at all, given the lower accessibility and quality of generators before ChatGPT.

The arXiv dataset provides comprehensive metadata, including title, abstract, publication date, and category. However, the introduction and conclusion sections are not part of the metadata, which implies the need for PDF parsing to extract these sections. From the metadata, each paper's ID and version are utilized to construct the document path and retrieve the corresponding PDF from the publicly accessible Google Cloud Storage bucket. Each PDF is then fed to the PyMuPDF [55] library to be parsed and to extract the relevant content. Unfortunately, parsing PDFs is known to be very challenging. This is particularly true for a double-column format, which many scientific papers have. Despite having tested several heuristic rules to identify and extrapolate the correct sections, the process can still fail at times. We discard data points where the parsing was unsuccessful.

The resulting set includes 12,000 real papers. Furthermore, we collect an additional 4000 samples undergoing a different parsing procedure. The intention is to ensure there are no recognizable parsing artifacts that inadvertently ease the detection process (see Section 4).

3.2. Fake Papers Generation

For the *fake* component of our dataset, we employ several models to generate abstracts, introductions, and conclusions based on scientific paper titles. The overview of the models used for generation is illustrated in Figure 2. The titles of the real papers sourced from the arXiv database (see Section 3.1) serve as prompts for the models to generate the target sections—i.e., ***abstract***, ***introduction***, and ***conclusion***.



(**a**) GPT-2 generation (**b**) Galactica/GPT-3 generation (**c**) ChatGPT generation

Figure 2. Generation pipeline used for each model. For GPT-2 (**a**), the abstract, introduction, and conclusion sections are generated by three separately fine-tuned model instances, each based solely on the paper title. In the case of Galactica and GPT-3 (**b**), each section is generated conditioning on the previous sections. Finally, ChatGPT's generation sequence (**c**) requires only the title to generate all the necessary sections at once.

To create fake scientific papers, we fine-tune GPT-2 and GPT-3 instances [1,4] and also leverage SCIgen [15], Galactica [16], and ChatGPT [8]. For each model—as shown in Figure 2—we employ a unique prompting/querying strategy to produce the desired paper sections.

This combination of models, ranging from CFG to state-of-the-art LLMs, aims to generate a diverse set of artificially generated scientific papers. Concrete examples of generated papers can be found in Appendix A.

3.2.1. SCIgen

Alongside the papers produced by the various LLMs, our fake dataset incorporates documents generated by SCIgen [15]. Although the quality of CFG-generated text is rather low and hence straightforward to identify, it remains relevant to ensure that current detectors can distinguish machine-generated papers even if poorly written and containing nonsensical content. Stribling and Aguayo [56] show that such papers have been accepted in scientific venues in the past.

Prompting SCIgen is done simply by running it as an offline script (https://github.com/soerface/scigen-docker) (accessed on 24 April 2023) which generates all the needed sections, including the title. The entire paper in LaTeXformat is generated as a result.

3.2.2. GPT-2

We fine-tune three distinct GPT-2 base (https://huggingface.co/gpt2) (accessed on 24 April 2023) models (124 M parameters) [4] to individually generate each section based on the given title. The models are trained in a seq2seq fashion [57], with the training procedure spanning six epochs and incorporating 3500 real papers. When encountering lengthy inputs, we truncate those exceeding 1024 tokens, potentially resulting in less coherent introductions and conclusions. Abstracts remain more coherent as they typically fall below this threshold. We release these separately fine-tuned GPT-2 instances to generate abstract (https://huggingface.co/tum-nlp/IDMGSP-GPT-2-ABSTRACT) (accessed on 31 July 2023), introduction (https://huggingface.co/tum-nlp/IDMGSP-GPT-2-INTRODUCTION) (accessed on 31 July 2023), and conclusion (https://huggingface.co/tum-nlp/IDMGSP-GPT-2-CONCLUSION) (accessed on 31 July 2023) for public usage and investigation.

Hyperparameters: For training, we use a batch size of 16 across all six epochs. We set the `max_new_token` to 512, `top_k` to 50, and `top_p` to 0.5 for all three models.

Post-processing: We remove generated "$\backslash n$" characters and any extra sections not explicitly mentioned in the prompt. Additionally, we remove incomplete sentences preceding the start of a new sentence. These are indeed common artifacts of GPT-2 and are easily identifiable by lowercase letters.

Although our GPT-2 model is specifically fine-tuned for the task, generating long pieces of text occasionally results in less meaningful content. Moreover, we observe that decoupling the generation of sections can lead to inconsistencies among the generated sections within the papers.

3.2.3. Galactica

Galactica is trained on a large corpus of scientific documents [16]. Therefore, it is already well-suited for the task of generating scientific papers. To facilitate the generation of coherent long-form text, we divide the generation process into smaller segments, with each section relying on preceding sections for context. For instance, while generating a conclusion, we provide the model with the title, abstract, and introduction as concatenated text.

Hyperparameters: We use Galactica base (https:// huggingface.co/facebook/galactica-1.3b) (accessed on 24 April 2023) (1.3 B parameters) [16] to generate each paper section based on the previous sections. The complete set of hyperparameters can be found in Table A1 in the Appendix A. Additionally, we enforce max length left padding. Due to the limited model capacity, restriction of the output number of tokens is necessary to avoid the hallucination risk introduced by long text generation.

Post-processing: To ensure completeness and coherence in the generated text, we devise a generation loop that meticulously assesses the quality of the output. For example, if the generated text lacks an `<EOS>` (end-of-sentence) token, the model is prompted to regenerate the text. Furthermore, we eliminate any special tokens introduced by Galactica during the process.

While Galactica base has 1.3 B parameters, it is still smaller than ChatGPT, which can result in less coherent outputs when generating longer text segments. As a result, prompting the model to generate a specific section with preceding sections as context yields better outcomes compared to providing only the title as context and requesting the model to generate all three sections simultaneously.

3.2.4. ChatGPT

To generate a cohesive document, we prompt ChatGPT (https://help.openai.com/en/articles/6825453-chatgpt-release-notes, release from 15 December 2022) [8] with "*Write a document with the title* [TITLE], *including an abstract, an introduction, and a conclusion*", substituting [TITLE] with the desired title utterance. ChatGPT's large size (20B parameters) and strong ability to consider context eliminate the necessity of feeding previous output sections into the prompt for generating newer ones.

Hyperparameters: For the entire generation process, we use the default temperature of 0.7.

Despite not being explicitly trained for scientific text generation, ChatGPT can produce extensive, human-like text in this domain. This capability likely stems from the model's large size, the extensive datasets it was trained on, and the incorporation of reinforcement learning with human feedback.

3.2.5. GPT-3

We fine-tune an instance of GPT-3 (`text-curie-001`, 6.7 B parameters) [1] with 178 real samples. Output papers generated through an iterative cascade process (as with Galactica) present a much higher quality than those forged in a single step (as with ChatGPT). Hence, we opt for the former strategy for GPT-3.

Pre/Post-Processing: To force the generation of cleaner outputs, we add an <END> token at the end of each input used for fine-tuning. GPT-3 mimics this behavior and predicts this token as well, so we remove every token added after generation <END>.

While still not on par with ChatGPT-generated outputs, we report a high quality for GPT-3-crafted papers.

3.3. Co-Created Papers Generation

The co-created component of our dataset mimics papers written by humans and models concurrently, a combination that is likely to appear in practice. That means texts originally written by either a human or an LLM and subsequently extended, paraphrased, or otherwise adjusted by the other. To create such papers at scale, we take a set of 4000 real papers from our TEST dataset (see Table 2) and paraphrase them with ChatGPT [8]. To stay within ChatGPT's context length limits, we paraphrase each paper section—i.e., ***abstract***, ***introduction***, and ***conclusion***—in a separate prompt. We then construct co-created papers with varying shares of human and machine input by combining original and paraphrased sections as shown in Figure 3.

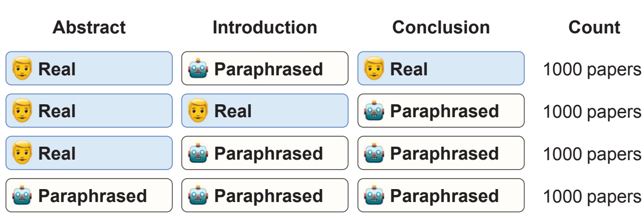


Figure 3. Our co-created test dataset TEST-CC contains 4000 papers with varying shares of *real* and ChatGPT-paraphrased sections.

Table 2. Overview of the datasets used to train and evaluate the classifiers. Each column represents the number of papers used per source. Concerning *real* papers, unless indicated, we use samples extracted with parsing 1 (see Section 3.1).

Dataset	arXiv (Real)	ChatGPT (Fake)	GPT-2 (Fake)	SCIgen (Fake)	Galactica (Fake)	GPT-3 (Fake)	ChatGPT (Co-Created)
Standard train (TRAIN)	8 k	2 k	2 k	2 k	2 k	-	-
Standard train subset (TRAIN-SUB)	4 k	1 k	1 k	1 k	1 k	-	-
TRAIN without ChatGPT (TRAIN-CG)	8 k	-	2 k	2 k	2 k	-	-
TRAIN plus GPT-3 (TRAIN + GPT3)	8 k	2 k	2 k	2 k	2 k	1.2 k	-
Standard test (TEST)	4 k	1 k	1 k	1 k	1 k	-	-
Out-of-domain GPT-3 only (OOD-GPT3)	-	-	-	-	-	1 k	-
Out-of-domain real (OOD-REAL)	4 k (parsing 2)	-	-	-	-	-	-
ChatGPT only (TECG)	-	1 k	-	-	-	-	-
Co-created test (TEST-CC)	-	-	-	-	-	-	4 k

Hyperparameters: For paraphrasing, we use OpenAI's `gpt-3.5-turbo-0613` model and set the temperature to 1.0 to achieve the largest deviation from the original human-written text.

4. Detection Experiments

In this section, we conduct experiments about identifying the source of a given paper—i.e., determining whether it is *fake* or *real*. We further investigate the ability of our baseline classifiers to detect co-created papers with varying degrees of *fake*—i.e., paraphrased—content. We start by defining data splits and subsets for training and testing, which are useful to evaluate generalization capabilities. Next, we outline the classifiers used as baselines to measure performance on the benchmark task. Finally, we examine the detection performance of the classifiers, investigate the obtained explanations, and apply additional post hoc explainability methods to the classifiers to gain deeper insights into the detection process.

4.1. Data Splits and Generalization Tests

We divide our dataset (displayed in Table 1) into *standard train* and *standard test* sets for training and testing our classifiers, respectively. Furthermore, we aim to evaluate models on out-of-domain test data. To achieve this, we create various data subsets by applying different splits to our benchmark. All the splits utilized for our experiments are detailed in Table 2. For instance, the reader can observe the composition of a data split with no access to ChatGPT samples (TRAIN-CG) and test sets composed only of differently-parsed real papers (OOD-REAL), only ChatGPT papers (OOD-CG), or only GPT-3 ones (OOD-GPT3).

4.2. Classifiers

We build and evaluate seven classifiers to perform the downstream task of classifying scientific papers as *fake* or *real* based on their content (abstract, introduction, and conclusion sections)—we remind the reader that all paper titles are *real* and will therefore not serve as input to the classifiers. To obtain an understanding of the difficulty of this classification task, we train two simple bag-of-words-based classifiers, Logistic Regression (LR) [58] and Random Forest (RF) [19]. Further, we fine-tune GPT-3 [1], Galactica [16], and RoBERTa [17] for this detection task. Lastly, we use a ChatGPT-based classifier without fine-tuning and a novel classifier that we call Large Language Model Feature Extractor (LLMFE) that learns explainability features using an LLM and then performs classification with Random Forest.

To accommodate memory and API limitations, we impose a restriction on the input tokens for GPT-3, Galactica, and RoBERTa by truncating texts after a certain number of tokens (details described in the following sections per model). However, since the average length of the combined input sections is about 900 tokens, which is less than the truncation limit, this constraint does not lead to significant information loss.

4.2.1. Bag-of-Words Classifier

As the simplest classifiers, we evaluate Random Forest [19] and Logistic Regression [58] on TF-IDF [59] features. This is to obtain an indication of the difficulty of the classification task—i.e., whether there is any classification signal in word frequencies alone or the detection of *fake* scientific papers requires more complex features. With Random Forest and Logistic Regression, we can explain the results by examining feature importance and learned coefficients.

Hyperparameters: We use the Random Forest and Logistic Regression implementations in scikit-learn [60] with default hyperparameters. We create features based on n-grams. A comparison of accuracies when using 1-grams, 2-grams, or a combination of both can be found in Table A2 in the appendix. In the following, we will report results based on 1-grams as these yielded the highest accuracy scores.

4.2.2. GPT-3

We fine-tune a GPT-3 [1] Ada model (`text-ada-001`, 350 M parameters) for the classification task. GPT-3 is fine-tuned in a causal manner, where the model is prompted with the concatenated paper sections along with their corresponding label. This is set up as a binary classification where the output is a single token indicating whether the paper is *real* (0) or *fake* (1). During inference, the model generates a single token based on the sections of a given paper.

As fine-tuning GPT-3 models requires a paid API, we train it only on a smaller subset of our dataset (TRAIN-SUB) shown in Table 2. We limit the number of input tokens to 2048 while retaining the default hyperparameters provided by the API.

4.2.3. Galactica

We adapt Galactica-mini (https://huggingface.co/facebook/galactica-125m) (accessed on 24 April 2023) [16] from a causal language model that predicts probabilities for each word in the vocabulary to a binary classifier with an output layer that predicts probabilities for two labels: *fake* and *real*.

The model is provided with all sections concatenated together with the corresponding label. Galactica, being a causal language model, generates a probability distribution spanning the entire vocabulary in its output. Nevertheless, this approach incurs significant memory usage, particularly when employed as a classifier. Therefore, we opted to retrain the output layer to yield a probability distribution for binary outcomes.

Hyperparameters: To cope with memory constraints, we limit the number of input tokens to 2048. Additionally, we adjust the batch size to 2 with gradient accumulation steps of 4 and enabled mixed precision. Furthermore, we set the number of epochs to 4, weight decay to 0.01, and warm-up steps to 1000. Our initial learning rate is 5×10^{-6}.

4.2.4. RoBERTa

We fine-tune RoBERTa base (125 M parameters) (https://huggingface.co/roberta-base) (accessed on 24 April 2023) [17] for the classification task. RoBERTa is limited to 512 input tokens, meaning that all text exceeding this limit is ignored. Our dataset exceeds this constraint for many entries. We choose to address the problem by fine-tuning three separate RoBERTa models to classify the three sections individually rather than retraining the input layer by enlarging the input size. https://huggingface.co/tum-nlp/IDMGSP-RoBERTa-TRAIN-ABSTRACT (accessed on 31 July 2023) (https://huggingface.co/tum-nlp/IDMGSP-RoBERTa-TRAIN-INTRODUCTION) (accessed on 31 July 2023) (https://huggingface.co/tum-nlp/IDMGSP-RoBERTa-TRAIN-CONCLUSION) (accessed on 31 July 2023) We take the majority vote from three model instances as the final output for each sample. We prompt each model with the capitalized name of the section plus the content of the latter, e.g., "*Abstract: In this paper . . .*".

Hyperparameters: To fine-tune the RoBERTa base, we set the number of epochs to 2, weight decay to 0.001, and batch size to 16. As with Galactica, the initial learning rate is 5×10^{-6}, and the warmup steps 1000.

4.2.5. DetectGPT

We evaluate DetectGPT [18] as another classifier as it has been shown to detect LLM-generated texts with high accuracy.

Hyperparameters: We use DetectGPT's default configuration and code (https://github.com/BurhanUlTayyab/DetectGPT) (accessed on 15 May 2023).

4.2.6. ChatGPT

To obtain natural-language explanations for classification directly, we prompt ChatGPT [8] via the OpenAI API. With this, we determine whether a scientific paper is *fake* or *real* and retrieve an explanation for its decision. The prompts include the concatenated sections, each beginning with the section name (e.g., "*Abstract:\nIn this paper ...*"), and task instructions. We compare the detection performance of four different prompting styles:

(1) **Input-Output Prompting (IO):** First, return the prediction (i.e., *fake* or *real*). Second, follow up with an explanation of the reasons for the prediction.
(2) **Chain-of-Thought Prompting (CoT) [61]:** First, return a sequence of thoughts on whether the paper is more likely *fake* or *real*. Second, return the final prediction.
(3) **Indicator Prompting (IP):** First, return a set of observations indicating that the paper was written by a human. Second, return a set of observations indicating that the paper was generated by a machine. Third, return the final prediction.
(4) **Few-Shot Prompting (FS) [1]:** Perform Input-Output Prompting but include a set of 6 annotated examples—one example from each generator and one *real* example—in the prompt (i.e., scientific papers with their abstract, introduction, conclusion, and *fake* or *real* label).

On our specific task, we observe the best classification results for the IO prompting style. Hence, we will only report accuracy scores for this prompting style in the following. For a detailed accuracy comparison of the different prompting styles, see Table A3 in the appendix. When using CoT prompting, there is a large number of instances where ChatGPT refuses to return a definite class label (*real* or *fake*) but instead returns *unknown*. We treat these results as incorrect answers and thus observe low accuracy scores for CoT prompting. We did not observe this behavior for the other prompting styles.

Hyperparameters: For classification, we use OpenAI's `gpt-3.5-turbo-0613` model with the default temperature of 0.7. Only for Few-Shot Prompting, we prompt the `gpt-3.5-turbo-16k-0613` model as a larger context length is needed. We do not perform task-specific fine-tuning. Due to API limitations, we classify only 100 randomly sampled papers from each test set using each of the four prompting styles. During implementation, we also experimented with larger samples and observed consistent classification accuracy scores independent of the sample size.

4.2.7. Large Language Model Feature Extractor (LLMFE)

Finally, we introduce and evaluate a novel explainable classifier LLMFE that learns human-understandable features using an LLM and an approach inspired by contrastive learning [62]. These features can range from very low-level (e.g., occurrences of a specific word) to very high-level (e.g., logical conclusiveness of argumentation). Figure 4 shows how LLMFE works conceptually. Training this classifier follows a four-step process:

(1) **Feature Engineering:** The LLM is presented with a pair of one *real* and one *fake* scientific paper and instructed to describe a list of features that would best distinguish these papers from each other. As we score each feature on a range of 0 to 10, we further instruct the LLM to label the meaning of the extreme ends of this scale for each feature

to avoid ambiguity. This prompt is repeated for `n_pairs` times to extract multiple different sets of features based on different example pairs.

(2) **Feature Consolidation:** As the previous step may have generated a large number of features, many of which are duplicates or semantically similar, we consolidate the extracted features into a smaller feature set. This is done by vectorizing each feature description using embeddings and performing hierarchical/agglomerative clustering [63] on the embeddings. We then manually investigate the cluster dendrogram and define a distance threshold `d_thres`. We finally merge all features less than `d_thres` apart from each other and represent each cluster through the feature closest to the cluster centroid. If `d_thres` is chosen carefully, this results in a significantly smaller, semantically diverse, and duplicate-free feature set. More detailed illustrations of this step can be found in Appendix B.4.

(3) **Feature Scoring:** The LLM is presented with an abstract, introduction, and conclusion of a scientific paper and descriptions of all features in the feature set. It is then instructed to assign an integer value from 0 to 10 to each feature that most accurately describes the scientific paper. This prompt is repeated for each example in the training dataset of size `n_sample`.

(4) **Classifier Training:** The previous steps resulted in a structured dataset of `n_sample` examples with one integer value for each feature in the learned feature set. Further, class labels (i.e., *real* or *fake*) are known. This dataset is used to train a Random Forest [19] classifier that learns to detect papers based on the features described by the LLM.

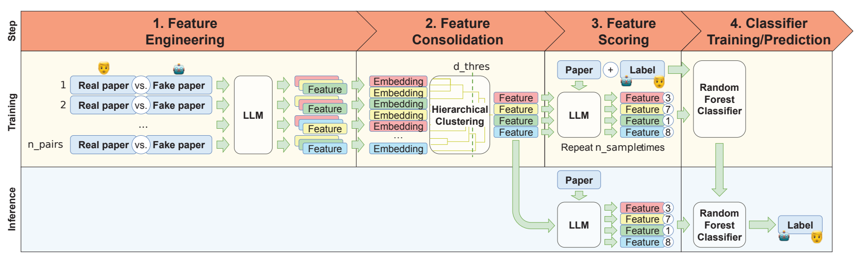


Figure 4. LLMFE follows a four-step process: (1) Generate features suitable for distinguishing *real* and *fake* papers using the LLM based on multiple pairs of one *real* and one *fake* paper each. (2) Remove duplicate features through hierarchical clustering on embeddings of the feature descriptions. (3) Score scientific papers along the remaining features using the LLM. (4) Finally, train a Random Forest Classifier to predict the *real* or *fake* label based on the feature scores.

Throughout the first three steps, the LLM is made aware of its overall goal of distinguishing *real* and *fake* scientific papers through the prompt instructions. We add this context information to best exploit the LLM's general world understanding obtained through extensive pre-training and to compensate for the relatively small sample sizes used for training. Inference on the test dataset then requires only two steps:

(1) **Feature Scoring:** Similar to the Feature Scoring step during training, a set of new papers is scored along the learned features.

(2) **Classifier Prediction:** The class label of the new papers is predicted using the trained Random Forest classifier.

Hyperparameters: Our LLMFE implementation uses OpenAI's `gpt-3.5-turbo-0613` with the default temperature of 0.7 for the Feature Engineering step and `gpt-3.5-turbo-16k-0613` with a temperature of 0.0—for deterministic behavior—for the Feature Scoring step. We set `n_pairs=100` and obtained 884 features from the Feature Engineering step. For the Feature Consolidation step, we create embeddings of the feature descriptions with OpenAI's `text-embedding-ada-002` and `chunk_size=1000`. We apply agglomerative clustering from Scipy's [64] linkage implementation with a cosine distance metric and calculate the average distance between clusters. We chose d_thres=0.05 as this resulted in a convenient balance between de-duplication and semantic feature diversity, yielding a final set of 83 features. We finally trained a Random Forest classifier with scikit-learn's [60] default hyperparameters on 600 papers from the TRAIN dataset (300 *real* papers and 60 *fake* papers from each generator).

4.3. Performance

Table 3 presents a summary of the accuracy scores achieved by our models on various splits. Given the significance of evaluating generalization to unseen generators, we highlight out-of-domain settings in blue. We exclude experiments entailing training GPT-3 on TRAIN + GPT3 and TRAIN-CG due to limited OpenAI API credits. Results of our fine-tuned models and LLMFE are also compared with DetectGPT as an existing zero-shot detection baseline [18], ChatGPT, and our Logistic Regression (LR) and Random Forest (RF) classifiers trained on 1-gram TF-IDF features.

Table 3. Experiment results reported with accuracy metric. Out-of-domain experiments, i.e., evaluation on unseen generators, are highlighted in blue. Highest values per test set are highlighted in bold. (*) ChatGPT-IO and LLMFE accuracies have been evaluated on randomly sampled subsets of 100 scientific papers per test set due to API limits.

Model	Train Dataset	TEST	OOD-GPT3	OOD-REAL	TECG	TEST-CC
LR-1gram (tf-idf) (our)	TRAIN	95.3%	4.0%	94.6%	96.1%	7.8%
LR-1gram (tf-idf) (our)	TRAIN + GPT3	94.6%	86.5%	86.2%	97.8%	13.7%
LR-1gram (tf-idf) (our)	TRAIN-CG	86.6%	0.8%	97.8%	32.6%	1.2%
RF-1gram (tf-idf) (our)	TRAIN	94.8%	24.7%	87.3%	**100.0%**	8.1%
RF-1gram (tf-idf) (our)	TRAIN + GPT3	91.7%	95.0%	69.3%	**100.0%**	15.1%
RF-1gram (tf-idf) (our)	TRAIN-CG	97.6%	7.0%	95.0%	57.0%	1.7%
Galactica (our)	TRAIN	98.4%	25.9%	95.5%	84.0%	6.8%
Galactica (our)	TRAIN + GPT3	98.5%	71.2%	95.1%	84.0%	12.0%
Galactica (our)	TRAIN-CG	96.4%	12.4%	97.6%	61.3%	2.4%
RoBERTa (our)	TRAIN	72.3%	55.5%	50.0%	**100.0%**	63.5%
RoBERTa (our)	TRAIN + GPT3	65.7%	**100.0%**	29.1%	**100.0%**	**75.0%**
RoBERTa (our)	TRAIN-CG	86.0%	2.0%	92.5%	76.5%	9.2%
GPT-3 (our)	TRAIN-SUB	**100.0%**	25.9%	99.0%	**100.0%**	N/A
DetectGPT	-	61.5%	0.0%	**99.9%**	68.7%	N/A
ChatGPT-IO (our) (*)	-	69.0%	49.0%	89.0%	0.0%	3.0%
LLMFE (our) (*)	TRAIN + GPT3	80.0%	62.0%	70.0%	90.0%	33.0%

Our simplest models, LR and RF, already achieve accuracy scores greater than 90% on the TEST dataset, suggesting that the classification task of distinguishing *real* and *fake* scientific papers is rather easy to learn if trained on comparable scientific papers. However, evaluated against out-of-domain scientific papers, accuracy scores drop significantly. All models perform poorly on out-of-domain papers generated by GPT-3 curie (OOD-GPT3). This result supports the findings of previous studies by Bakhtin et al. [43], which indicate that models trained on specific generators tend to overfit and perform poorly on data outside their training distribution. However, after training our Galactica and RoBERTa models with GPT-3 examples (TRAIN + GPT3), the models achieve higher accuracies (71% and 100%, respectively). A similar behavior can be observed for the LR and RF classifiers.

All models, except RoBERTa, perform poorly when detecting human–machine co-created papers (TEST-CC). Seeing papers generated by ChatGPT and GPT-3 during training each noticeably improves the detection accuracy for all models, presumably because these examples are most similar to the ChatGPT-paraphrased papers that are part of the TEST-CC dataset. RoBERTa still achieves an accuracy of 75%, which is remarkable given that many examples only contain a relatively low share of machine-generated text. This seems to be due to a high-recall bias of the trained RoBERTa model, which achieves comparatively high accuracy scores on datasets that only contain *fake* papers (i.e., OOD-GPT3, TECG) but lower scores on the remaining datasets that also contain *real* papers. GPT-3 and DetectGPT have not been evaluated against TEST-CC due to limited computing resources and API credits.

Models that were not fine-tuned to the classification task, DetectGPT and ChatGPT, perform noticeably worse than the fine-tuned models. Our ChatGPT-based LLMFE outperforms ChatGPT on all test datasets except OOD-REAL, indicating that LLM's detection abilities can be enhanced with a systematic prompting approach and guidance. In particular, we observe great improvements in the more sophisticated texts in our TECG and TEST-CC datasets. This may be because of the more high-level features identified by LLMFE—e.g., those that capture a paper's overall coherence.

It is worth noting that our RoBERTa model exhibits excellent results when evaluated on a dataset of ChatGPT-generated papers (TECG). The model achieves an accuracy of 77% without prior training on a similar dataset (TRAIN-CG), and 100% accuracy when a similar dataset is included in the training (TRAIN). These results outperform Galactica in both scenarios.

The overall good results on OOD-REAL—i.e., real paper processed with a different parser—indicate that our models are not exploiting any spurious artifact introduced during the parsing procedure. DetectGPT notably overfits papers generated with GPT-2 and deems most samples coming from a different source as real. Indeed, it performs well on OOD-REAL (100%) and poorly on OOD-GPT3 (0%).

4.4. Explainability Insights

The different types of classifier models provide a rich set of explainability insights that help us understand what characterizes *real* and *fake* scientific papers, respectively. LR and RF classifiers trained on TF-IDF 1-grams provide insights into individual words. For Galactica, RoBERTa, and GPT-3, we extract insights on more complex features of word combinations. Lastly, LLMFE learns very high-level, abstract features describing complex relationships between words, such as grammar and cohesion. Additionally, we analyze linguistic-based features such as readability scores and the length of papers.

4.4.1. Word-Level Insights from LR and RF

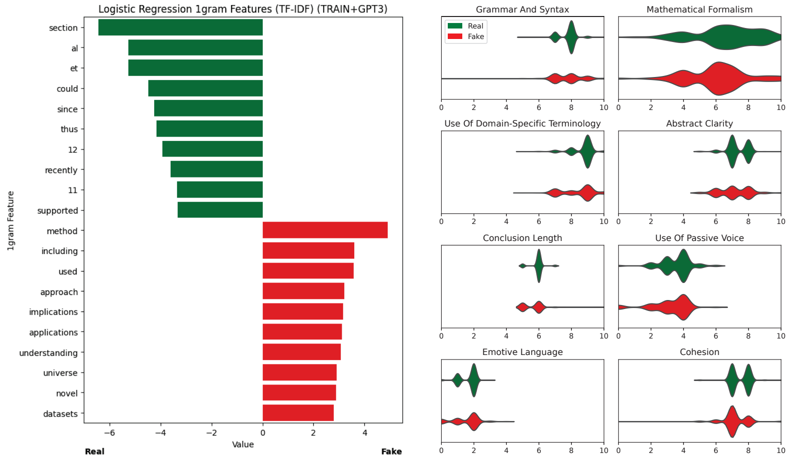
The coefficients learned by LR (see Figure 5a) and feature importance learned by RF indicate that *real* papers draw from a diverse set of words and—more often than *fake* papers—make references to specific sections ("`section`"), other papers ("`et`" and "`al`"), or recent trends ("`recently`"). In contrast, *fake* papers tend to rely on one-size-fits-all vocabulary such as "`method`", "`approach`", or "`implications`" more than *real* papers.

4.4.2. LIME and SHAP Insights for Galactica, RoBERTa, and GPT-3

We use LIME [65] and SHAP [66] to inspect predictions made by Galactica, RoBERTa, and GPT-3. While these explanations fail to convey a concise overview, they are still useful to notice patterns and similarities across samples sharing labels and sources [67,68].

Often, RoBERTa and Galactica models tend to classify papers as *real* when the papers include infrequent words and sentences starting with adverbs. In addition, we notice that SHAP explanations corresponding to *real* papers have all words with low Shapley values. We believe this is intuitive as a paper appears *real* if it does not contain any artifact that strongly signals an AI source.

On the other hand, papers whose sections begin with *"In this paper, . . . "*, *"In this work, . . . "*, or *"In this study, . . . "* are often marked as *fake*. The same goes for those containing repeated words, spelling mistakes, or word fragments such as *"den"*, *"oly"*, *"um"*. Detectors are also able to spot incoherent content and context, as well as sections that are unnaturally short and do not convey any specific point. Several explanation instances of Galactica and RoBERTa can be found in Appendix C for further inspection. We choose not to provide an explanation for our GPT-3 classifier since it requires many requests to OpenAI's paid API.



(**a**) LR 1-gram coefficients (**b**) LLMFE feature distributions

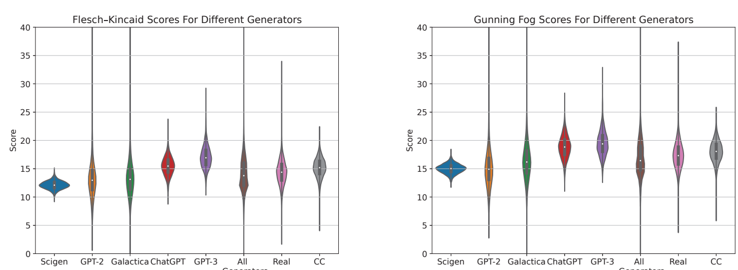
Figure 5. Explainability insights from our Logistic Regression (LR) and Large Language Model Feature Extractor (LLMFE) classifiers. (**a**) shows the 1-grams with the 10 lowest (indicating *real*) and highest (indicating *fake*) coefficients learned by LR. (**b**) shows the distributions of scores for the eight most important features (according to Random Forest feature importance) learned by LLMFE.

4.4.3. Abstract Features from LLMFE

LLMFE identifies more abstract features such as *grammar and syntax*, *use of domain-specific terminology*, or *cohesion* as shown in Figure 5b. We observe that score distributions of *real* papers tend to be narrower than those of *fake* papers. This is not surprising given that *fake* papers were generated by multiple generators, some more and some less advanced. For many features, the distributions of *real* and *fake* papers have the same mode, suggesting that collectively our dataset of machine-generated papers resembles *real* papers quite well.

4.4.4. Readability Metrics for Different Generators

Figure 6 shows the distribution of Flesch–Kincaid Grade Level [69] and Gunning Fog [70] readability metrics [71] for papers from the different generators and *real* papers. Flesch–Kincaid measures the technical difficulty of the papers, while Gunning Fog measures the readability of the papers. The comparison confirms our observation that our machine-generated papers are representative of *real* papers with a slight increase in writing sophistication from SCIgen and GPT-2 to ChatGPT and GPT-3 generators, with Galactica being the median.

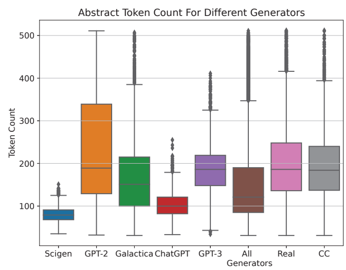


(**a**) Flesch Kincaid (**b**) Gunning Fog

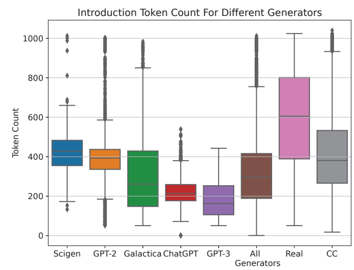
Figure 6. Distribution of readability metrics for papers from the different generators. (**a**) shows Flesch–Kincaid scores while (**b**) shows Gunning Fog scores for all generators.

4.4.5. Generated Texts Length

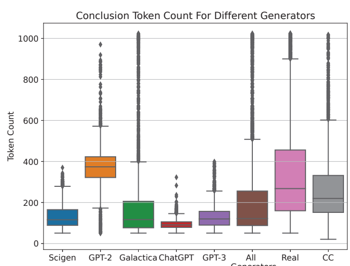
We observe differences in the length of the sections in our *fake* scientific papers depending on the generator. Figure 7 shows the length distributions of sections generated by the different generators. On average, machine-generated sections from all generators are shorter than sections from *real* papers—the only exception being abstracts and conclusions generated by GPT-2, which are slightly longer than *real* abstracts and conclusions, on average. For most generators, we also see less length variety compared to real papers.



(**a**) Abstract Length



(**b**) Introduction Length



(**c**) Conclusion Length

Figure 7. The generators exhibit different tendencies for the length of the generated *fake* scientific papers. (**a**) shows the length distribution of generated abstracts, (**b**) shows the same for introductions, and (**c**) shows conclusion lengths.

For the co-created scientific papers (CC), despite prompting ChatGPT to return paraphrased sections with a similar length or even the exact word count as the original sections, we observe a tendency of ChatGPT to summarize sections during paraphrasing. While paraphrased abstracts have roughly the same length as their originals, paraphrased introductions, and conclusions sections are often significantly shorter, as shown in Figure 8. We conclude that ChatGPT does not reliably follow length constraints when confronted with a *paraphrasing* task on longer texts.

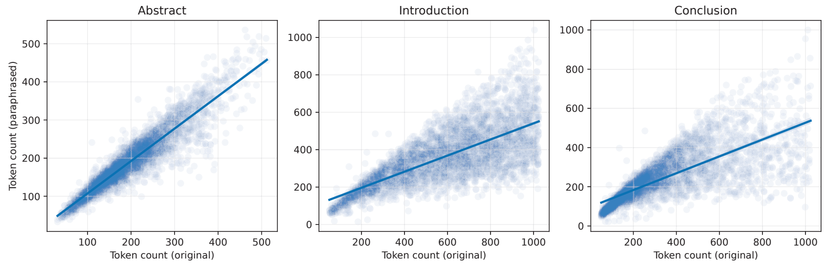


Figure 8. Paraphrasing sections with ChatGPT has a tendency to result in sections shorter than the original. The reduction in section length is most visible for the longer introduction and conclusion sections. For an analysis of lengths of generated *fake* scientific papers, see Figure 7 in the appendix.

5. Limitations and Future Work

Despite memory, GPU, and API limitations presenting significant obstacles for our project, we could still create high-quality *fake* scientific papers. Nonetheless, we believe there is room for improvement in addressing such limitations. For instance, beyond simply improving the quality of the generated papers, further insights could be gained from exploring generation processes entailing a collaboration between different models and input prompts.

Due to the complexity of parsing PDFs, we are currently limited to specific sections (abstract, introduction, conclusion) instead of complete papers. Moreover, processing entire publications would require substantial computational efforts. We believe that selecting sections dynamically at random instead of a fixed choice is worth exploring and will be the focus of future work.

Beyond DetectGPT [18], other zero-shot text detectors such as GPTZero (https://gptzero.me) (accessed on 31 July 2023) present promising solutions worth testing on our benchmark dataset. However, at the time of writing, such solutions are not available for experiments at scale.

In future work, we aim to address these limitations by exploring dynamic section selection, combining models and prompts in the generation process, improving papers' quality, and investigating the potential of zero-shot text detectors such as GPTZero as they become more accessible and scalable. We think that future research should further investigate how stable classifiers, such as the ones presented in this paper, are against newly appearing LLMs and how to improve the classifiers' generalization capabilities to out-of-domain samples.

6. Discussion, Ethical Considerations, and Broader Impact

It is important to emphasize that our work does not condemn the usage of LLMs. The legitimacy of their usage should be addressed by regulatory frameworks and guidelines. Still, we strongly believe it is crucial to develop countermeasures and strategies to detect machine-generated papers to ensure accountability and reliability in published research.

Our benchmark dataset serves as a valuable resource for evaluating detection algorithms, contributing to the integrity of the scientific community. However, potential challenges include adversarial attacks and dataset biases [72,73]. It is essential to develop robust countermeasures and strive for a diverse, representative dataset.

7. Conclusions

This work introduced a benchmark dataset for identifying machine-generated scientific papers in the LLM era. Our work creates a resource that allows researchers to evaluate the effectiveness of detection methods and thus support the trust and integrity of the scientific process.

We generated a diverse set of papers using both SCIgen and state-of-the-art LLMs—ChatGPT, Galactica, GPT-2, and GPT-3. This ensures a variety of sources and includes models capable of generating convincing content. We fine-tuned and tested several baseline detection models—Logistic Regression, Random Forest, GPT-3, Galactica, and RoBERTa—and compared their performance to DetectGPT, ChatGPT, and a novel Large Language Model Feature Extractor (LLMFE) that we propose. The results demonstrated varying degrees of success, with some models showing remarkable performance on specific subsets while sometimes struggling with out-of-domain data.

By providing a comprehensive platform for evaluating detection techniques, we contribute to the development of robust and reliable methods for identifying machine-generated content. Moving forward, we plan to address the current limitations and further enhance the utility of our benchmark for the research community.

We release a repository containing our benchmark dataset as well as the code used for experimental results (https:// huggingface.co/datasets/tum-nlp/IDMGSP) (accessed on 31 July 2023).

Author Contributions: Design of experiments, E.M. and D.D.; Dataset creation, E.M., M.H.I.A. and S.M.; Experiments, M.H.I.A. and S.M.; writing—original draft preparation, E.M.; writing—journal version preparation, D.D. and S.M.; writing—review and editing, G.G. All authors have read and agreed to the published version of the manuscript.

Funding: This paper has been supported by the German *Federal Ministry of Education and Research* (BMBF, grant 01IS17049).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: All data created within this research is available online (https://huggingface.co/datasets/tum-nlp/IDMGSP, accessed on 27 July 2023).

Acknowledgments: We would like to acknowledge the help of Social Research Computing Group for providing an opportunity to conduct this research and Paolo Basso and Margherita Musumeci for participating in the first version of this publication. Additionally, we would like to thank Leslie McIntosh for her guidance throughout our research journey.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

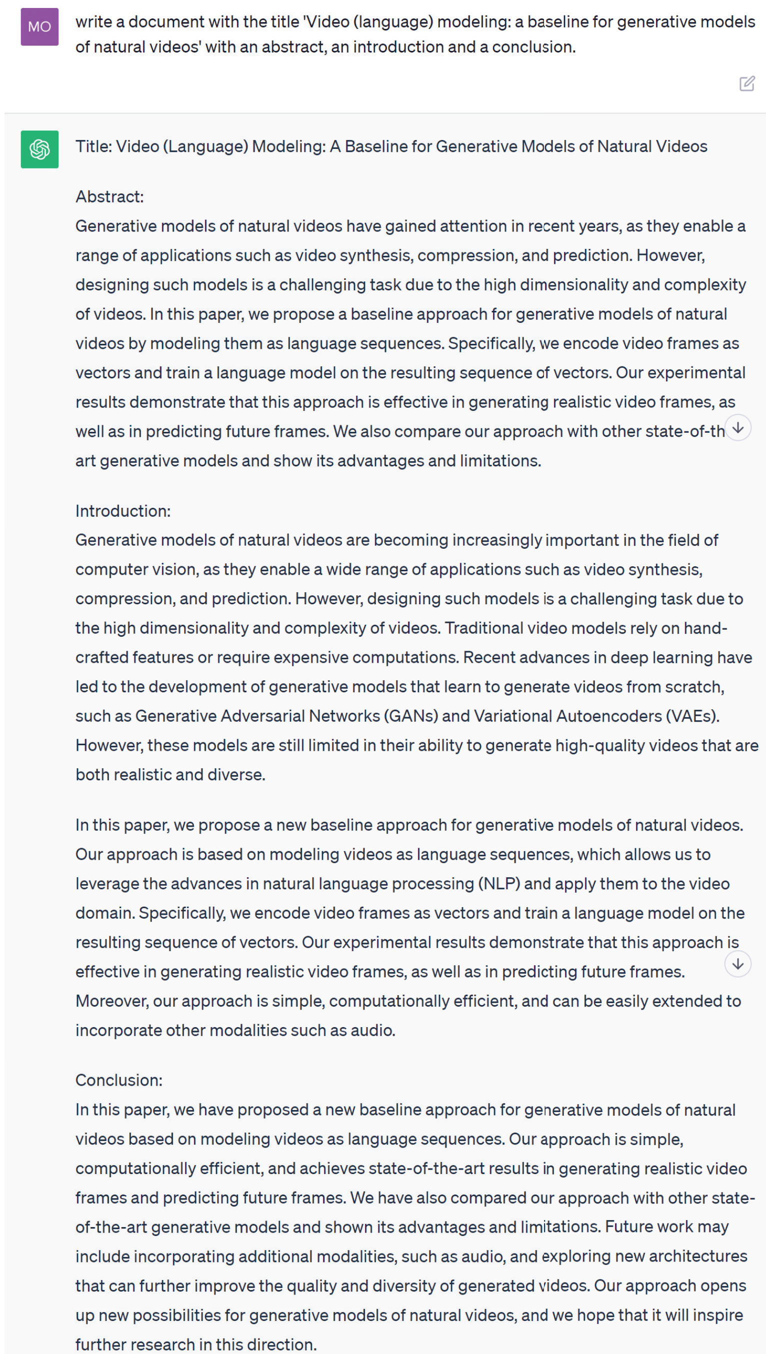
NLP	Natural Language Processing
LLM	Large Language Model
BoW	Bag-of-Words
OOD	Out-of-Domain
CFG	Context-Free-Grammars

Appendix A. Generation Examples

In this section, we present examples of text that were generated using the models we employed.

Appendix A.1. ChatGPT

Figures A1 and A2 provide examples of prompts that can be used to generate paper sections using ChatGPT.



MO write a document with the title 'Video (language) modeling: a baseline for generative models of natural videos' with an abstract, an introduction and a conclusion.

Title: Video (Language) Modeling: A Baseline for Generative Models of Natural Videos

Abstract:
Generative models of natural videos have gained attention in recent years, as they enable a range of applications such as video synthesis, compression, and prediction. However, designing such models is a challenging task due to the high dimensionality and complexity of videos. In this paper, we propose a baseline approach for generative models of natural videos by modeling them as language sequences. Specifically, we encode video frames as vectors and train a language model on the resulting sequence of vectors. Our experimental results demonstrate that this approach is effective in generating realistic video frames, as well as in predicting future frames. We also compare our approach with other state-of-th art generative models and show its advantages and limitations.

Introduction:
Generative models of natural videos are becoming increasingly important in the field of computer vision, as they enable a wide range of applications such as video synthesis, compression, and prediction. However, designing such models is a challenging task due to the high dimensionality and complexity of videos. Traditional video models rely on hand-crafted features or require expensive computations. Recent advances in deep learning have led to the development of generative models that learn to generate videos from scratch, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs). However, these models are still limited in their ability to generate high-quality videos that are both realistic and diverse.

In this paper, we propose a new baseline approach for generative models of natural videos. Our approach is based on modeling videos as language sequences, which allows us to leverage the advances in natural language processing (NLP) and apply them to the video domain. Specifically, we encode video frames as vectors and train a language model on the resulting sequence of vectors. Our experimental results demonstrate that this approach is effective in generating realistic video frames, as well as in predicting future frames. Moreover, our approach is simple, computationally efficient, and can be easily extended to incorporate other modalities such as audio.

Conclusion:
In this paper, we have proposed a new baseline approach for generative models of natural videos based on modeling videos as language sequences. Our approach is simple, computationally efficient, and achieves state-of-the-art results in generating realistic video frames and predicting future frames. We have also compared our approach with other state-of-the-art generative models and shown its advantages and limitations. Future work may include incorporating additional modalities, such as audio, and exploring new architectures that can further improve the quality and diversity of generated videos. Our approach opens up new possibilities for generative models of natural videos, and we hope that it will inspire further research in this direction.

Figure A1. Example 1 of prompting ChatGPT to produce sections of a scientific paper given the paper title.

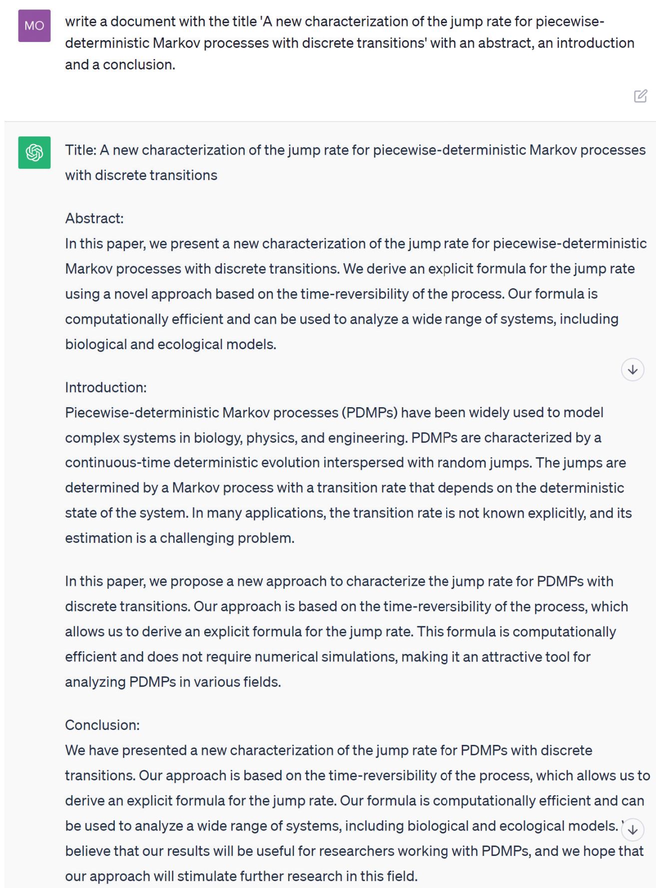


Figure A2. Example 2 of prompting ChatGPT to produce sections of a scientific paper given the paper title.

Appendix A.2. Galactica

For generating text with the Galactica model, an overview of the hyperparameters used is provided in Table A1. Each subsection shows how Galactica is prompted to generate the abstract, the introduction and the conclusion sections.

Table A1. Hyperparameters used to generate each paper section in the Galactica model. Each row corresponds to a decoding of a section based on the previous input sections. Here we used parameters of the MODEL.GENERATE function provided by Huggingface [74].

Input Section(s)	Output Section	Tokenizer Max_Input_Size	Max_New_Tokens	Do_Sample	Temperature	top_k	top_p	No_Repeat_Ngram_Size	Early_Stopping
Title	Abstract	64	512	True	-	-	-	-	-
Title + Abstract	Introduction	576 (64 + 512)	1024	True	0.7	25	0.9	10	True
Title + Abstract + Introduction	Conclusion	1600 (64 + 512 + 1024)	1024	True	0.7	25	0.9	10	True

Appendix A.2.1. Abstract Generation

INPUT:

Title: On the Global Structure of Hopf Hypersurfaces in Complex Space Form

OUTPUT:

Abstract: For a generic hypersurface in complex space form, all Hopf hypersurfaces are proved to be homogeneous or Hopf homogeneous. As a consequence, it is shown that the generalized Sasakian metric is a Ricci soliton on a Hopf hypersurface.

Appendix A.2.2. Introduction Generation

INPUT:

Title: On the Global Structure of Hopf Hypersurfaces in Complex Space Form

Abstract: For a generic hypersurface in complex space form, all Hopf hypersurfaces are proved to be homogeneous or Hopf homogeneous. As a consequence, it is shown that the generalized Sasakian metric is a Ricci soliton on a Hopf hypersurface.

Introduction:

OUTPUT:

A Hopf hypersurface is a hypersurface which admits a Hopf field, that is, a vector field such that $[\xi, \xi]$ is a normal vector field. The Hopf field is unique up to multiplication by a positive constant. The existence of Hopf fields is equivalent to the existence of a 1-form η such that $\eta \wedge d\eta^{n-1}$ is a volume form. Let M^n be a hypersurface in a complex space form $M^{2n+1}(c)$ with constant sectional curvature c. We denote by g the metric induced by the Riemannian metric of $M^{2n+2}(c)$. The Hopf field is given by $\xi = \frac{1}{\sqrt{n}} J\nabla f$, where J is the complex structure on M^{2n+3} and f is a function on $M^n \subset M^{2n+1}$. In [1], Wang and Chen showed that a Hopf hypersurface in $M^{2n+l}(c)$ is homogeneous if and only if the Hopf field is parallel. In this paper, we study the structure of Hopf hypersurfaces in complex space form. We show that a Hopf hypersurface is homogeneous if and only if it is Hopf homogeneous. We also show that a Hopf hypersurface is Hopf homogeneous if and only if the generalized Sasakian metric is a Ricci solitons on it.

Appendix A.2.3. Conclusion Generation

INPUT:

Title: On the Global Structure of Hopf Hypersurfaces in Complex Space Form

Abstract: For a generic hypersurface in complex space form, all Hopf hypersurfaces are proved to be homogeneous or Hopf homogeneous. As a consequence, it is shown that the generalized Sasakian metric is a Ricci soliton on a Hopf hypersurface.

Introduction:

A Hopf hypersurface is a hypersurface which admits a Hopf field, that is, a vector field such that $[\xi, \xi]$ is a normal vector field. The Hopf field is unique up to multiplication by a positive constant. The existence of Hopf fields is equivalent to the existence of a 1-form η such that $\eta \wedge d\eta^{n-1}$ is a volume form. Let M^n be a hypersurface in a complex space form $M^{2n+1}(c)$ with constant sectional curvature c. We denote by g the metric induced by the Riemannian metric of $M^{2n+2}(c)$. The Hopf field is given by $\xi = \frac{1}{\sqrt{n}} J\nabla f$, where J is the complex structure on M^{2n+3} and f is a function on $M^n \subset M^{2n+1}$. In [1], Wang and Chen showed that a Hopf hypersurface in $M^{2n+l}(c)$ is homogeneous if and only if the Hopf field is parallel. In this paper, we study the structure of Hopf hypersurfaces in complex space form. We show that a Hopf hypersurface is homogeneous if and only if it is Hopf homogeneous. We also show that a Hopf hypersurface is Hopf homogeneous if and only if the generalized Sasakian metric is a Ricci solitons on it.

Conclusion:

OUTPUT:

We obtain a complete classification of Hopf hypersurfaces in complex space forms.

Appendix A.3. GPT-2

In the following, we show an example of how the fine-tuned GPT-2 was used to generate the abstract, the introduction, and the conclusion sections.

Appendix A.3.1. Abstract Generation

INPUT:

Competitive Multi-Agent Load Balancing with Adaptive Policies in Wireless Networks <|sep|>

OUTPUT:

In this paper, we propose adaptive multi-agent load balancing (AML) based on adaptive policies to achieve better performance. The adaptive policies can be designed to maximize the number of agent users and to reduce the number of latency. We present a comprehensive evaluation of AML on three wireless networks. We demonstrate that the adaptive policies outperform state-of-the-art AML strategies by an average of 6.5% and by an average of 2.5% when compared to state-of-the-art strategies. The performance of adaptive policies is evaluated on the three wireless networks with multiple users and multiple latency.

Appendix A.3.2. Introduction Generation

INPUT:

Competitive Multi-Agent Load Balancing with Adaptive Policies in Wireless Networks <|sep|>

OUTPUT:

Wireless networks are increasingly becoming the backbone of the modern world. The wireless networks have been the main source of energy for many years, and they are the main source of power for many businesses. However, wireless networks are also the main source of information for many people. For example, people in many countries rely on a variety of information services, such as social networks, social media, and e-mail. However, the information services are not available in all countries. Therefore, the information services are not available for all countries.

Appendix A.3.3. Conclusion Generation

INPUT:

Competitive Multi-Agent Load Balancing with Adaptive Policies in Wireless Networks <|sep|>

OUTPUT:

In this paper, we propose adaptive policies that optimize the network performance of different types of wireless networks. We show that adaptive policies can achieve competitive performance for different types of wireless networks. The proposed adaptive policies have been tested in several real-world wireless networks. We find that adaptive policies outperform state-of-the-art policies for the most common wireless networks. We also show that adaptive policies can be used to optimize the network performance of wireless networks by adapting their performance to different types of wireless networks. The proposed adaptive policies are able to achieve competitive performance for different types of wireless networks.

Appendix A.4. GPT-3

The following frames show how each section is generated using a GPT-3 generator.

Appendix A.4.1. Abstract Generation

INPUT:

Primitive Representation Learning for Scene Text Recognition

OUTPUT:

Primitive scene representation learning is an important subfield of deep learning, which has been widely applied in many image recognition tasks such as image segmentation and object detection. In recent years, several methods have been proposed to learn the primitive scene representation. Among them, most methods are based on U-Net architecture and use either a global or local context to capture the global or local structure of the input scenes. However, these approaches do not fully exploit the structure information in scenes. In this paper, we propose a novel primitive scene representation learning method for scene text recognition using a residual block-based encoder-decoder strategy with multiple level-up passes. To this end, we design three residual block structures with different subsampling ratios: 1:1 (global), 2:1 (local) and 1:2 (global+local). We also designed an attention module that can be used to further guide the search space in each level-up pass by enforcing certain conditions on hierarchical representations obtained during previous levels.

Appendix A.4.2. Introduction Generation

INPUT:

Primitive Representation Learning for Scene Text Recognition

Primitive scene representation learning is an important subfield of deep learning, which has been widely applied in many image recognition tasks such as image segmentation and object detection. In recent years, several methods have been proposed to learn the primitive scene representation. Among them, most methods are based on U-Net architecture and use either a global or local context to capture the global or local structure of the input scenes. However, these approaches do not fully exploit the structure information in scenes. In this paper, we propose a novel primitive scene representation learning method for scene text recognition using a residual block-based encoder-decoder strategy with multiple level-up passes. To this end, we design three residual block structures with different subsampling ratios: 1:1 (global), 2:1 (local) and 1:2 (global+local). We also designed an attention module that can be used to further guide the search space in each level-up pass by enforcing certain conditions on hierarchical representations obtained during previous levels.

Introduction:

OUTPUT:

[1,2]. It has been widely applied in many image segmentation and object detection [3,4] tasks. In these applications, deep learning is often used to learn the complex parameters such as models and training schemes. Alternatively, it can be also applied on the more fundamental problems of learning a better model for predicting new input data that have not been seen before. Learning representations of unseen data is one of the most fundamental problems within machine learning and has attracted tremendous attention in recent years [5]. Primitive scene representation learning (PSRL) is an important subfield of deep learning which aims at modeling the basic visual concepts such as objects and scenes in order to solve various real-world applications likeimage segmentation [6], pedestrian detection [7], video classification [8] etc.

Appendix A.4.3. Conclusion Generation

INPUT:

Primitive Representation Learning for Scene Text Recognition

Primitive scene representation learning is an important subfield of deep learning, which has been widely applied in many image recognition tasks such as image segmentation and object detection. In recent years, several methods have been proposed to learn the primitive scene representation. Among them, most methods are based on U-Net architecture and use either a global or local context to capture the global or local structure of the input scenes. However, these approaches do not fully exploit the structure information in scenes. In this paper, we propose a novel primitive scene representation learning method for scene text recognition using a residual block-based encoder-decoder strategy with multiple level-up passes. To this end, we design three residual block structures with different subsampling ratios: 1:1 (global), 2:1 (local) and 1:2 (global+local). We also designed an attention module that can be used to further guide the search space in each level-up pass by enforcing certain conditions on hierarchical representations obtained during previous levels.

[1,2]. It has been widely applied in many image segmentation and object detection [3,4] tasks. In these applications, deep learning is often used to learn the complex parameters such as models and training schemes. Alternatively, it can be also applied on the more fundamental problems of learning a better model for predicting new input data that have not been seen before. Learning representations of unseen data is one of the most fundamental problems within machine learning and has attracted tremendous attention in recent years [5]. Primitive scene representation learning (PSRL) is an important subfield of deep learning which aims at modeling the basic visual concepts such as objects and scenes in order to solve various real-world applications likeimage segmentation [6], pedestrian detection [7], video classification [8] etc.

OUTPUT:

We propose a novel residual block-based encoder-decoder (R-BiDe) method for learning the primitive scene representation. We introduce a residual block structure with different subsampling ratios, and design an attention module that can be used to guide the search space in each level-up pass. The proposed R-BiDe method achieves better performance than other stateof-theart methods on several difficult scene text recognition benchmarks with no overfitting issues. In future works, we will focus on exploring the model design principles of R-BiDe as well as its application scenarios such as object detection and autonomous driving using deep learning frameworks such as Caffe and PyTorch.

Appendix A.5. SCIgen

In the case of SCIgen papers, a script is used to generate the paper documents. Once the documents are generated, they are subsequently parsed to extract the relevant sections. The following frame shows an example of a generation after parsing.

Title: Towards the Understanding of Randomized Algorithms

Abstract: Many security experts would agree that, had it not been for online algorithms, the investigation of XML might never have occurred. Given the current status of heterogeneous epistemologies, statisticians predictably desire the evaluation of public-private key pairs, which embodies the important principles of programming languages. In order to solve this riddle, we present an analysis of expert systems (Judas), disconfirming that symmetric encryption and hash tables are never incompatible. It at first glance seems counterintuitive but fell in line with our expectations .

Introduction: Recent advances in interposable modalities and trainable modalities do not necessarily obviate the need for 802.11 mesh networks. After years of robust research into Byzantine fault tolerance, we disprove the evaluation of information retrieval systems. The notion that security experts collaborate with atomic symmetries is generally adamantly opposed. The evaluation of link-level acknowledgements would tremendously amplify vacuum tubes. The basic tenet of this solution is the development of IPv7. For example, many frameworks allow the study of the transistor. On a similar note, we emphasize that our system improves systems. Thus, we use lossless communication to disprove that online algorithms and journaling file systems can interact to fulfill this ambition. Relational methodologies are particularly unfortunate when it comes to "smart" information. This is an important point to understand. However, for example, many frameworks observe the memory bus. Thusly, we see no reason not to use trainable communication to develop concurrent theory. This outcome is usually a private ambition but is supported by related work in the field. We describe new perfect modalities, which we call Judas. Though such a claim might seem counterintuitive, it has ample historical precedence. To put this in perspective, consider the fact that little-known futurists never use IPv6 to surmount this problem. Contrarily, stochastic technology might not be the panacea that cyberneticists expected. Two properties make this method distinct: our application prevents homogeneous configurations, and also Judas is copied from the analysis of DHTs [1]. To put this in perspective, consider the fact that little-known information theorists rarely use 802.11b to address this challenge. Combined with the UNIVAC computer, such a claim synthesizes new stochastic modalities. The rest of this paper is organized as follows. Primarily, we motivate the need for the transistor. Similarly, we place our work in context with the prior work in this area. On a similar note, to solve this question, we construct an analysis of telephony (Judas), which we use to show that the seminal relational algorithm for the exploration of active networks by Thompson [1] runs in $\Omega(\log \log n)$ time. In the end, we conclude.

Conclusion: Our method will address many of the issues faced by today's theorists. Similarly, Judas can successfully prevent many link-level acknowledgements at once. Our methodology for constructing the improvement of the Turing machine is particularly excellent. We plan to explore more problems related to these issues in future work.

Appendix B. Classifier Details

Appendix B.1. Bag-of-Words Classifiers

Table A2 shows the detailed results for the different bag-of-words classifiers introduced in Section 4.2.1.

Table A2. Experiment results for the different bag-of-words classifiers reported with accuracy metric. Out-of-domain experiments are highlighted in blue. The highest values per test set are highlighted in bold.

Model	Train Dataset	TEST	OOD-GPT3	OOD-REAL	TECG	TEST-CC
LR-1gram (tf-idf)	TRAIN	95.3%	4.0%	94.6%	96.1%	7.8%
LR-1gram (tf-idf)	TRAIN + GPT3	94.6%	86.5%	86.2%	97.8%	13.7%
LR-1gram (tf-idf)	TRAIN-CG	86.6%	0.8%	97.8%	32.6%	1.2%
LR-2gram (tf-idf)	TRAIN	89.1%	0.5%	96.5%	91.3%	6.4%
LR-2gram (tf-idf)	TRAIN + GPT3	90.0%	89.7%	86.1%	97.3%	15.7%
LR-2gram (tf-idf)	TRAIN-CG	73.3%	0.0%	**99.6%**	1.4%	0.6%
LR-(1,2)gram (tf-idf)	TRAIN	94.8%	0.2%	97.8%	94.6%	2.7%
LR-(1,2)gram (tf-idf)	TRAIN + GPT3	95.1%	83.3%	92.6%	97.8%	5.9%
LR-(1,2)gram (tf-idf)	TRAIN-CG	83.3%	0.2%	99.3%	1.7%	0.3%
RF-1gram (tf-idf)	TRAIN	94.8%	24.7%	87.3%	**100.0%**	8.1%
RF-1gram (tf-idf)	TRAIN + GPT3	91.7%	95.0%	69.3%	**100.0%**	15.1%
RF-1gram (tf-idf)	TRAIN-CG	**97.6%**	7.0%	95.0%	57.0%	1.7%
RF-2gram (tf-idf)	TRAIN	90.8%	12.4%	76.8%	99.3%	29.9%
RF-2gram (tf-idf)	TRAIN + GPT3	87.7%	**96.8%**	54.6%	99.9%	**44.0%**
RF-2gram (tf-idf)	TRAIN-CG	85.8%	3.4%	88.8%	44.1%	8.5%
RF-(1,2)gram (tf-idf)	TRAIN	95.4%	22.4%	87.8%	93.8%	9.1%
RF-(1,2)gram (tf-idf)	TRAIN + GPT3	93.8%	96.0%	66.6%	**100.0%**	19.7%
RF-(1,2)gram (tf-idf)	TRAIN-CG	87.8%	1.9%	96.8%	43.8%	1.1%

Appendix B.2. GPT-3

The following frame shows a GPT-3 classifier training prompt. The input label (`1` for *fake* and `0` for *real*) is separated from the input by the separator token (###).

Abstract:

For a generic hypersurface in complex space form, all Hopf hypersurfaces are proved to be homogeneous or Hopf homogeneous. As a consequence, it is shown that the generalized Sasakian metric is a Ricci soliton on a Hopf hypersurface.

Introduction:

A Hopf hypersurface is a hypersurface which admits a Hopf field, that is, a vector field such that $[\xi, \xi]$ is a normal vector field. The Hopf field is unique up to multiplication by a positive constant. The existence of Hopf fields is equivalent to the existence of a 1-form η such that $\eta \wedge d\eta^{n-1}$ is a volume form. Let M^n be a hypersurface in a complex space form $M^{2n+1}(c)$ with constant sectional curvature c. We denote by g the metric induced by the Riemannian metric of $M^{2n+2}(c)$. The Hopf field is given by $\xi = \frac{1}{\sqrt{n}} J\nabla f$, where J is the complex structure on M^{2n+3} and f is a function on $M^n \subset M^{2n+1}$. In [1], Wang and Chen showed that a Hopf hypersurface in $M^{2n+l}(c)$ is homogeneous if and only if the Hopf field is parallel. In this paper, we study the structure of Hopf hypersurfaces in complex space form. We show that a Hopf hypersurface is homogeneous if and only if it is Hopf homogeneous. We also show that a Hopf hypersurface is Hopf homogeneous if and only if the generalized Sasakian metric is a Ricci solitons on it.

Conclusion:

For a generic hypersurface in complex space form, all Hopf hypersurfaces are proved to be homogeneous or Hopf homogeneous. As a consequence, it is shown that the generalized Sasakian metric is a Ricci soliton on a Hopf hypersurface.

###

1

Appendix B.3. ChatGPT

Table A3 shows the detailed results for the different ChatGPT prompting styles introduced in Section 4.2.6.

Table A3. Experiment results for different ChatGPT prompting styles reported with accuracy metric. Out-of-domain experiments are highlighted in blue. Highest values per test set are highlighted in bold. (*) ChatGPT accuracies have been evaluated on randomly sampled subsets of 100 scientific papers per test set and prompting style due to API limits.

Model	Train Dataset	TEST	OOD-GPT3	OOD-REAL	TECG	TEST-CC
ChatGPT-IO (*)	-	**69%**	**49%**	89%	0%	3%
ChatGPT-CoT (*)	-	63%	2%	70%	3%	1%
ChatGPT-IP (*)	-	57%	18%	92%	**7%**	**5%**
ChatGPT-FS (*)	TRAIN + GPT3	59%	2%	**100%**	0%	0%

Appendix B.4. Large Language Model Feature Extractor (LLMFE)

Figure A3 show an extract from the hierarchical clustering dendrogram learned during the feature consolidation step of LLMFE.

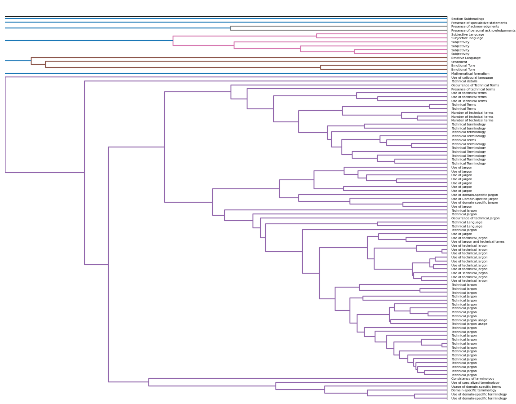
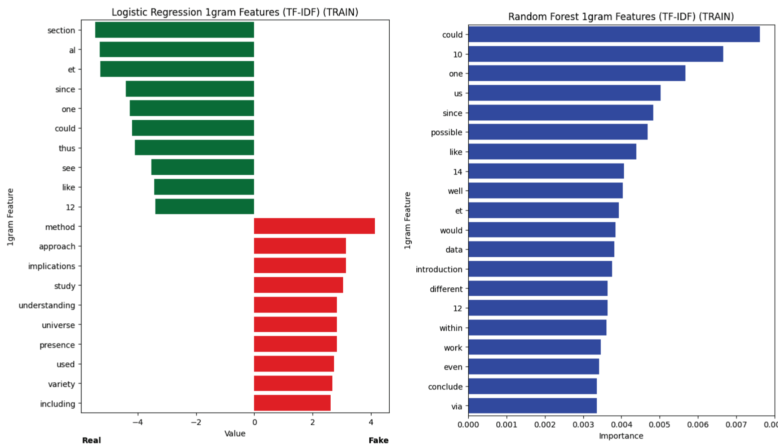


Figure A3. Extract from the hierarchical clustering dendrogram learned during the feature consolidation step of LLMFE. The full dendrogram lists all 884 features. The distance threshold was chosen so that 83 clusters were created from the 884 features.

Appendix C. Explainability Results

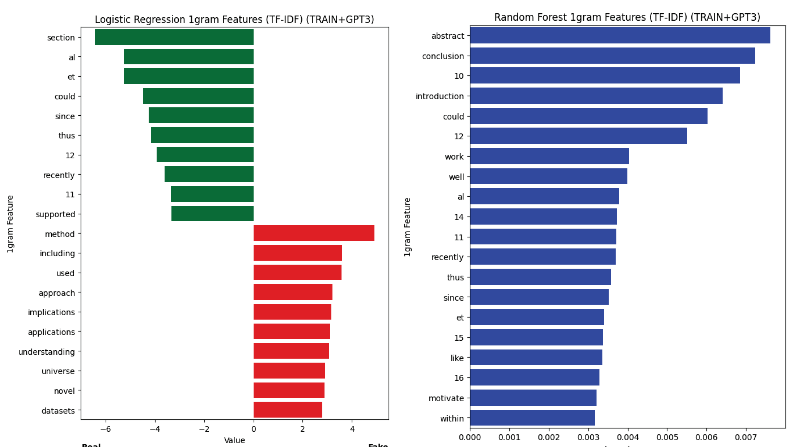
Appendix C.1. Bag-of-Words Classifiers

Figures A4–A6 show the coefficients and feature importance learned by our Logistic Regression (LR) and Random Forest (RF) classifiers on the TRAIN, TRAIN + GPT3, and TRAIN-CG datasets, respectively.



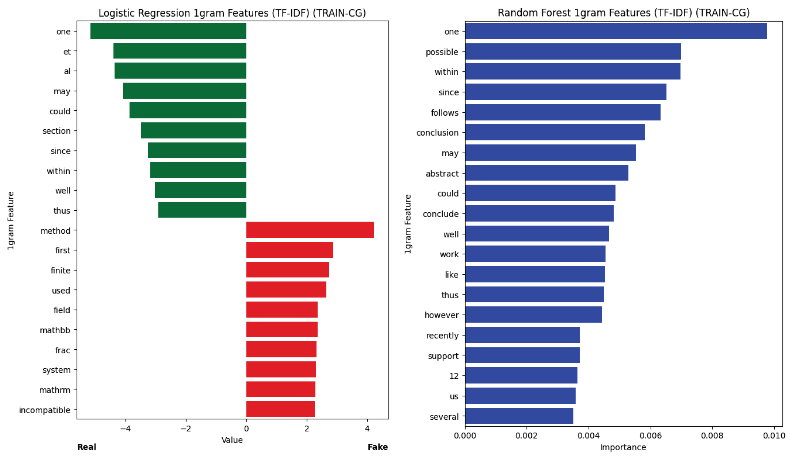
(**a**) LR 1-gram coefficients (**b**) RF 1-gram feature importance

Figure A4. Explainability insights from our Logistic Regression (LR) and Random Forest (RF) classifiers on the TRAIN dataset. (**a**) shows the 1-grams with the 10 lowest (indicating *real*) and highest (indicating *fake*) coefficients learned by LR. (**b**) shows the feature importance extracted from RF after training.



(**a**) LR 1-gram coefficients (**b**) RF 1-gram feature importance

Figure A5. Explainability insights from our Logistic Regression (LR) and Random Forest (RF) classifiers on the TRAIN + GPT3 dataset. (**a**) shows the 1-grams with the 10 lowest (indicating *real*) and highest (indicating *fake*) coefficients learned by LR. (**b**) shows the feature importance extracted from RF after training.



(**a**) LR 1-gram coefficients (**b**) RF 1-gram feature importance

Figure A6. Explainability insights from our Logistic Regression (LR) and Random Forest (RF) classifiers on the TRAIN-CG dataset. (**a**) shows the 1-grams with the 10 lowest (indicating *real*) and highest (indicating *fake*) coefficients learned by LR. (**b**) shows the feature importance extracted from RF after training.

Appendix C.2. RoBERTa

Selected samples of SHAP and LIME explanations for our RoBERTa classifier can be found in Figures A7–A17.

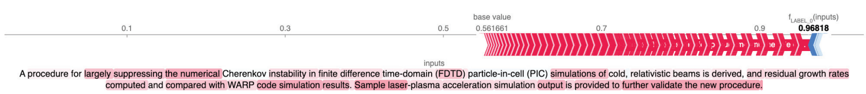


Figure A7. RoBERTa: Example of SHAP explanation on a real abstract correctly classified.

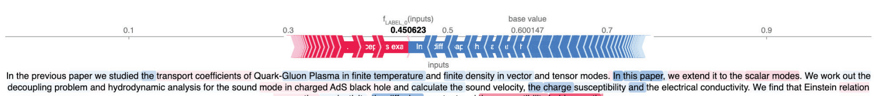


Figure A8. RoBERTa: Example of SHAP explanation on a real misclassified abstract.

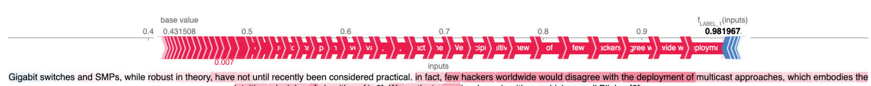


Figure A9. RoBERTa: Example of SHAP explanation on a SCIgen generated abstract correctly classified.

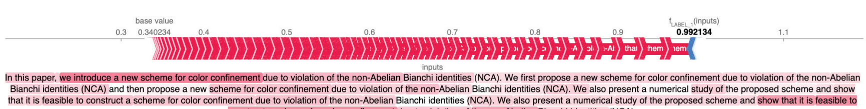


Figure A10. RoBERTa: Example of SHAP explanation on a GPT-2 generated abstract correctly classified.

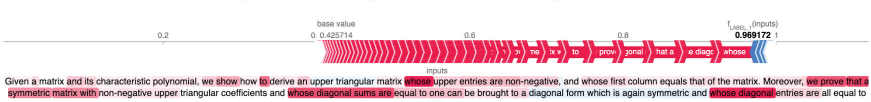


Figure A11. RoBERTa: Example of SHAP explanation on a Galactica generated abstract correctly classified.

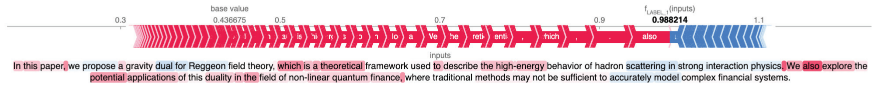


Figure A12. RoBERTa: Example of SHAP explanation on a ChatGPT generated abstract correctly classified.

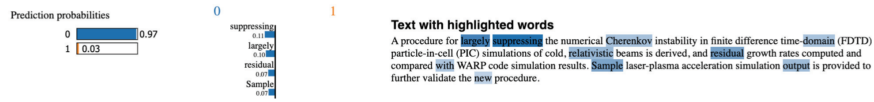


Figure A13. RoBERTa: Example of LIME explanation on a real abstract correctly classified.

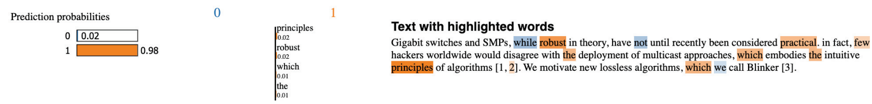


Figure A14. RoBERTa: Example of LIME explanation on a SCIgen generated abstract correctly classified.

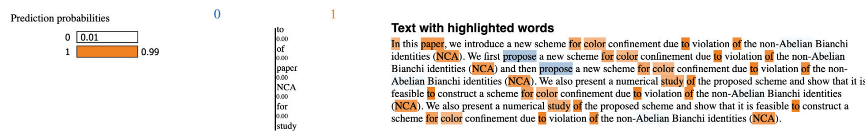


Figure A15. RoBERTa: Example of LIME explanation on a GPT-2 generated abstract correctly classified.

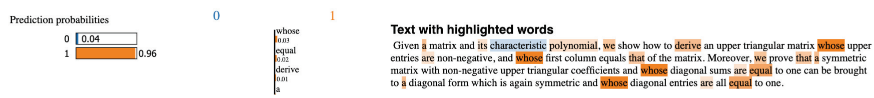


Figure A16. RoBERTa: Example of LIME explanation on a Galactica generated abstract correctly classified.

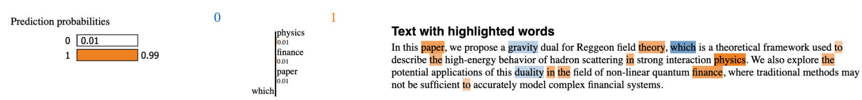


Figure A17. RoBERTa: Example of LIME explanation on a ChatGPT generated abstract correctly classified.

Appendix C.3. Galactica

Selected samples of SHAP explanations for our Galactica classifier can be found in Figures A18–A21.

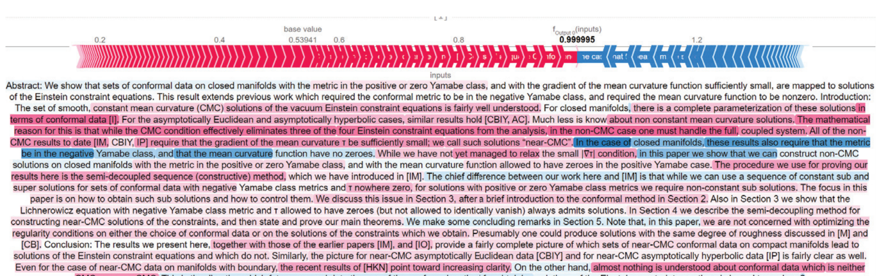


Figure A18. Galactica: Example of SHAP explanation on a real paper correctly classified.

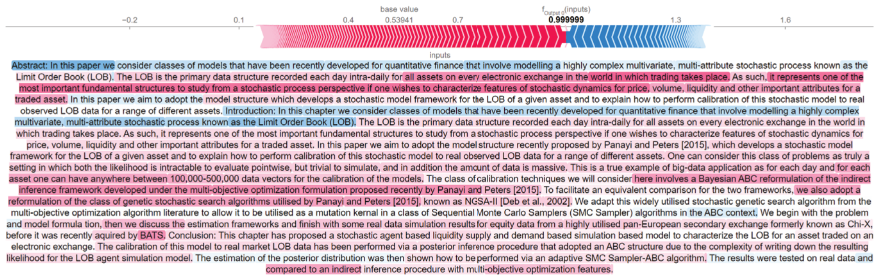


Figure A19. Galactica: Example of SHAP explanation on a misclassified real paper.

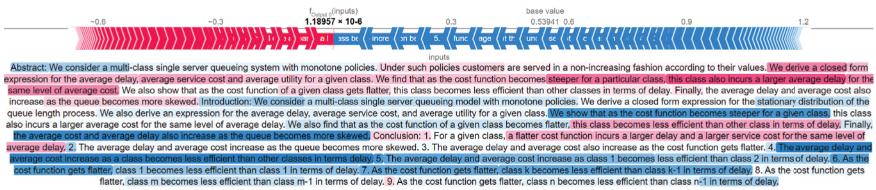


Figure A20. Galactica: Example of SHAP explanation on a Galactica generated paper correctly classified.

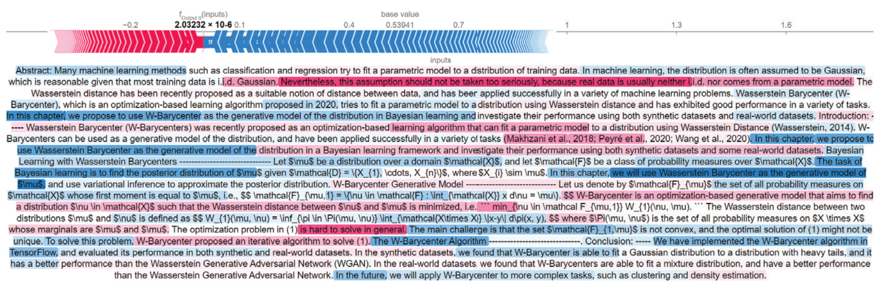


Figure A21. Galactica: Example of SHAP explanation on a misclassified Galactica generated paper.

References

1. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
2. Scao, T.L.; Fan, A.; Akiki, C.; Pavlick, E.; Ilić, S.; Hesslow, D.; Castagné, R.; Luccioni, A.S.; Yvon, F.; Gallé, M.; et al. Bloom: A 176b-parameter open-access multilingual language model. *arXiv* **2022**, arXiv:2211.05100.
3. OpenAI. GPT-4 Technical Report. *arXiv* **2023**, arXiv:2303.08774.
4. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models are Unsupervised Multitask Learners. *OpenAI Blog*. **2019**. Available online: https://insightcivic.s3.us-east-1.amazonaws.com/language-models.pdf (accessed on 31 July 2023).
5. Keskar, N.S.; McCann, B.; Varshney, L.R.; Xiong, C.; Socher, R. CTRL: A Conditional Transformer Language Model for Controllable Generation. *arXiv* **2019**, arXiv:1909.05858. https://doi.org/10.48550/arXiv.1909.05858.
6. Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv* **2023**, arXiv:2307.09288. https://doi.org/10.48550/arXiv.2307.09288.
7. Zellers, R.; Holtzman, A.; Rashkin, H.; Bisk, Y.; Farhadi, A.; Roesner, F.; Choi, Y. Defending Against Neural Fake News. In Proceedings of the Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, 8–14 December 2019.
8. OpenAI. ChatGPT. 2022. Available online: https://openai.com/blog/chat-ai/ (accessed on 26 February 2023).
9. Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H.P.d.O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. Evaluating large language models trained on code. *arXiv* **2021**, arXiv:2107.03374.
10. Liu, Y. Fine-tune BERT for extractive summarization. *arXiv* **2019**, arXiv:1903.10318.
11. Dergaa, I.; Chamari, K.; Zmijewski, P.; Saad, H.B. From human writing to artificial intelligence generated text: Examining the prospects and potential threats of ChatGPT in academic writing. *Biol. Sport* **2023**, *40*, 615–622. [CrossRef]
12. Stokel-Walker, C. AI bot ChatGPT writes smart essays-should academics worry? *Nature* **2022**. [CrossRef]

13. Maynez, J.; Narayan, S.; Bohnet, B.; McDonald, R. On Faithfulness and Factuality in Abstractive Summarization. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics. Online, 5–10 August 2021; pp. 1906–1919. [CrossRef]
14. Tian, R.; Narayan, S.; Sellam, T.; Parikh, A.P. Sticking to the facts: Confident decoding for faithful data-to-text generation. *arXiv* **2019**, arXiv:1910.08684.
15. Stribling, J.; Krohn, M.; Aguayo, D. SCIgen—An Automatic CS Paper Generator. 2005. Available online: https://pdos.csail.mit.edu/archive/scigen/ (accessed on 1 March 2023).
16. Taylor, R.; Kardas, M.; Cucurull, G.; Scialom, T.; Hartshorn, A.; Saravia, E.; Poulton, A.; Kerkez, V.; Stojnic, R. Galactica: A large language model for science. *arXiv* **2022**, arXiv:2211.09085.
17. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
18. Mitchell, E.; Lee, Y.; Khazatsky, A.; Manning, C.D.; Finn, C. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature. *arXiv* **2023**, arXiv:2301.11305.
19. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
20. Mosca, E.; Abdalla, M.H.I.; Basso, P.; Musumeci, M.; Groh, G. Distinguishing Fact from Fiction: A Benchmark Dataset for Identifying Machine-Generated Scientific Papers in the LLM Era. In Proceedings of the 3rd Workshop on Trustworthy Natural Language Processing (TrustNLP 2023), Toronto, ON, Canada, 9–14 July 2023; pp. 190–207.
21. Maronikolakis, A.; Schutze, H.; Stevenson, M. Identifying automatically generated headlines using transformers. *arXiv* **2020**, arXiv:2009.13375.
22. Liyanage, V.; Buscaldi, D.; Nazarenko, A. A Benchmark Corpus for the Detection of Automatically Generated Text in Academic Publications. In Proceedings of the Thirteenth Language Resources and Evaluation Conference, LREC 2022, Marseille, France, 20–25 June 2022; pp. 4692–4700.
23. Wang, Y.; Mansurov, J.; Ivanov, P.; Su, J.; Shelmanov, A.; Tsvigun, A.; Whitehouse, C.; Afzal, O.M.; Mahmoud, T.; Aji, A.F.; et al. M4: Multi-generator, Multi-domain, and Multi-lingual Black-Box Machine-Generated Text Detection. *arXiv* **2023**, arXiv:2305.14902. https://doi.org/10.48550/arXiv.2305.14902.
24. He, X.; Shen, X.; Chen, Z.; Backes, M.; Zhang, Y. MGTBench: Benchmarking Machine-Generated Text Detection. *arXiv* **2023**, arXiv:2303.14822. https://doi.org/10.48550/arXiv.2303.14822.
25. Li, Y.; Li, Q.; Cui, L.; Bi, W.; Wang, L.; Yang, L.; Shi, S.; Zhang, Y. Deepfake Text Detection in the Wild. *arXiv* **2023**, arXiv:2305.13242. https://doi.org/10.48550/arXiv.2305.13242.
26. Bird, S.; Dale, R.; Dorr, B.; Gibson, B.; Joseph, M.; Kan, M.Y.; Lee, D.; Powley, B.; Radev, D.; Tan, Y.F. The ACL Anthology Reference Corpus: A Reference Dataset for Bibliographic Research in Computational Linguistics. In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08); European Language Resources Association (ELRA), Marrakech, Morocco, 15–20 July 2008.
27. arXiv.org submitters. *arXiv Dataset*. **2023**. [CrossRef]
28. Cohan, A.; Goharian, N. Scientific Article Summarization Using Citation-Context and Article's Discourse Structure. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 12–17 September 2015; pp. 390–400. [CrossRef]
29. Saier, T.; Färber, M. Bibliometric-Enhanced arXiv: A Data Set for Paper-Based and Citation-Based Tasks. In Proceedings of the 8th International Workshop on Bibliometric-Enhanced Information Retrieval (BIR 2019) Co-Located with the 41st European Conference on Information Retrieval (ECIR 2019), Cologne, Germany, 14 April 2019; pp. 14–26.
30. Lo, K.; Wang, L.L.; Neumann, M.; Kinney, R.; Weld, D. S2ORC: The Semantic Scholar Open Research Corpus. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 4969–4983. [CrossRef]
31. Kashnitsky, Y.; Herrmannova, D.; de Waard, A.; Tsatsaronis, G.; Fennell, C.C.; Labbe, C. Overview of the DAGPap22 Shared Task on Detecting Automatically Generated Scientific Papers. In Proceedings of the Third Workshop on Scholarly Document Processing, Association for Computational Linguistics. Gyeongju, Republic of Korea, 17 October 2022; pp. 210–213.
32. Gao, L.; Biderman, S.; Black, S.; Golding, L.; Hoppe, T.; Foster, C.; Phang, J.; He, H.; Thite, A.; Nabeshima, N.; et al. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv* **2021**, arXiv:2101.00027.
33. Waswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the NIPS, 4–9 December 2017.
34. Anil, R.; Dai, A.M.; Firat, O.; Johnson, M.; Lepikhin, D.; Passos, A.; Shakeri, S.; Taropa, E.; Bailey, P.; Chen, Z.; et al. PaLM 2 Technical Report. *arXiv* **2023**, arXiv:2305.10403. https://doi.org/10.48550/arXiv.2305.10403.
35. Maheshwari, H.; Singh, B.; Varma, V. SciBERT Sentence Representation for Citation Context Classification. In Proceedings of the Second Workshop on Scholarly Document Processing, Online, 10 June 2021; pp. 130–133.
36. MacNeil, S.; Tran, A.; Leinonen, J.; Denny, P.; Kim, J.; Hellas, A.; Bernstein, S.; Sarsa, S. Automatically Generating CS Learning Materials with Large Language Models. *arXiv* **2022**, arXiv:2212.05113.
37. Swanson, B.; Mathewson, K.; Pietrzak, B.; Chen, S.; Dinalescu, M. Story centaur: Large language model few shot learning as a creative writing tool. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, Online, 21–23 April 2021; pp. 244–256.

38. Liu, S.; He, T.; Li, J.; Li, Y.; Kumar, A. An Effective Learning Evaluation Method Based on Text Data with Real-time Attribution—A Case Study for Mathematical Class with Students of Junior Middle School in China. *ACM Trans. Asian Low Resour. Lang. Inf. Process.* **2023**, *22*, 63:1–63:22. [CrossRef]
39. Jawahar, G.; Abdul-Mageed, M.; Lakshmanan, L.V.S. Automatic Detection of Machine Generated Text: A Critical Survey. In Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), 8–13 December 2020; pp. 2296–2309. [CrossRef]
40. Gehrmann, S.; Strobelt, H.; Rush, A.M. Gltr: Statistical detection and visualization of generated text. *arXiv* **2019**, arXiv:1906.04043.
41. Fagni, T.; Falchi, F.; Gambini, M.; Martella, A.; Tesconi, M. TweepFake: About detecting deepfake tweets. *PLoS ONE* **2021**, *16*, e0251415. [CrossRef]
42. Kushnareva, L.; Cherniavskii, D.; Mikhailov, V.; Artemova, E.; Barannikov, S.; Bernstein, A.; Piontkovskaya, I.; Piontkovski, D.; Burnaev, E. Artificial Text Detection via Examining the Topology of Attention Maps. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event/Punta Cana, Dominican Republic, 7–11 November, 2021; Moens, M., Huang, X., Specia, L., Yih, S.W., Eds.; Association for Computational Linguistics: Cedarville, OH, USA, 2021; pp. 635–649. [CrossRef]
43. Bakhtin, A.; Gross, S.; Ott, M.; Deng, Y.; Ranzato, M.; Szlam, A. Real or fake? Learning to discriminate machine from human generated text. *arXiv* **2019**, arXiv:1906.03351.
44. Ippolito, D.; Duckworth, D.; Callison-Burch, C.; Eck, D. Automatic detection of generated text is easiest when humans are fooled. *arXiv* **2019**, arXiv:1911.00650.
45. Kirchenbauer, J.; Geiping, J.; Wen, Y.; Shu, M.; Saifullah, K.; Kong, K.; Fernando, K.; Saha, A.; Goldblum, M.; Goldstein, T. On the Reliability of Watermarks for Large Language Models. *arXiv* **2023**, arXiv:2306.04634. https://doi.org/10.48550/arXiv.2306.04634.
46. Amancio, D.R. Comparing the topological properties of real and artificially generated scientific manuscripts. *Scientometrics* **2015**, *105*, 1763–1779. [CrossRef]
47. Williams, K.; Giles, C.L. On the use of similarity search to detect fake scientific papers. In Proceedings of the Similarity Search and Applications: 8th International Conference, SISAP 2015, Glasgow, UK, 12–14 October 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 332–338.
48. Nguyen, M.T.; Labbé, C. Engineering a tool to detect automatically generated papers. In Proceedings of the BIR 2016 Bibliometric-enhanced Information Retrieval, Padua, Italy, 20 March 2016.
49. Cabanac, G.; Labbé, C. Prevalence of nonsensical algorithmically generated papers in the scientific literature. *J. Assoc. Inf. Sci. Technol.* **2021**, *72*, 1461–1476. [CrossRef]
50. Beltagy, I.; Lo, K.; Cohan, A. SciBERT: A Pretrained Language Model for Scientific Text. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 3615–3620. [CrossRef]
51. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
52. Glazkova, A.; Glazkov, M. Detecting generated scientific papers using an ensemble of transformer models. In Proceedings of the Third Workshop on Scholarly Document Processing, Gyeongju, Republic of Korea, 17 October 2022; pp. 223–228.
53. Liu, Z.; Yao, Z.; Li, F.; Luo, B. Check Me If You Can: Detecting ChatGPT-Generated Academic Writing using CheckGPT. *arXiv* **2023**, arXiv:2306.05524. https://doi.org/10.48550/arXiv.2306.05524.
54. Yang, L.; Jiang, F.; Li, H. Is ChatGPT Involved in Texts? Measure the Polish Ratio to Detect ChatGPT-Generated Text. *arXiv* **2023**, arXiv:2307.11380. https://doi.org/10.48550/arXiv.2307.11380.
55. Rudduck, P. PyMuPDF: Python Bindings for the MuPDF Renderer. 2021. Available online: https://pypi.org/project/PyMuPDF/ (accessed on 7 March 2023).
56. Stribling, J.; Aguayo, D. Rooter: A Methodology for the Typical Unification of Access Points and Redundancy. 2021. Available online: https://dipositint.ub.edu/dspace/bitstream/123456789/2243/1/rooter.pdf (accessed on 31 July 2023).
57. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 3104–3112.
58. Cox, D.R. The regression analysis of binary sequences. *J. R. Stat. Soc. Ser. B* **1958**, *20*, 215–232. [CrossRef]
59. Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.* **1988**, *24*, 513–523. [CrossRef]
60. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
61. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv* **2023**, arXiv:2201.11903.
62. Hadsell, R.; Chopra, S.; LeCun, Y. Dimensionality reduction by learning an invariant mapping. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, 17–22 June 2006; Volume 2, pp. 1735–1742.
63. Müllner, D. Modern hierarchical, agglomerative clustering algorithms. *arXiv* **2011**, arXiv:1109.2378.
64. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef]

65. Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 13–17 August 2016; pp. 1135–1144.
66. Lundberg, S.; Lee, S.I. A Unified Approach to Interpreting Model Predictions. International Conference on Machine Learning. *arXiv* **2017**, arXiv:1705.07874.
67. Mosca, E.; Szigeti, F.; Tragianni, S.; Gallagher, D.; Groh, G. SHAP-Based Explanation Methods: A Review for NLP Interpretability. In Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 4593–4603.
68. Mosca, E.; Harmann, K.; Eder, T.; Groh, G. Explaining Neural NLP Models for the Joint Analysis of Open-and-Closed-Ended Survey Answers. In Proceedings of the 2nd Workshop on Trustworthy Natural Language Processing (TrustNLP 2022), Seattle, WA, USA, 14 July 2022; pp. 49–63. [CrossRef]
69. Thomas, G.; Hartley, R.D.; Kincaid, J.P. Test-retest and inter-analyst reliability of the automated readability index, Flesch reading ease score, and the fog count. *J. Read. Behav.* **1975**, *7*, 149–154. [CrossRef]
70. Gunning, R. *The Technique of Clear Writing*; McGraw-Hill: New York, NY, USA, 1952. Available online: https://books.google.de/books?id=ofI0AAAAMAAJ (accessed on 31 July 2023).
71. DiMascio, C. py-readability-metrics. 2019. Available online: https://github.com/cdimascio/py-readability-metrics (accessed on 31 July 2023).
72. Mosca, E.; Agarwal, S.; Rando Ramírez, J.; Groh, G. "That Is a Suspicious Reaction!": Interpreting Logits Variation to Detect NLP Adversarial Attacks. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022; pp. 7806–7816. [CrossRef]
73. Huber, L.; Kühn, M.A.; Mosca, E.; Groh, G. Detecting Word-Level Adversarial Text Attacks via SHapley Additive exPlanations. In Proceedings of the 7th Workshop on Representation Learning for NLP, Dublin, Ireland, 26 May 2022; pp. 156–166. [CrossRef]
74. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C. Hugging Face's Transformers: State-of-the-Art Natural Language Processing. 2019. Available online: https://github.com/huggingface/transformers (accessed on 31 July 2023) .

information

MDPI

Article

A Survey on Using Linguistic Markers for Diagnosing Neuropsychiatric Disorders with Artificial Intelligence

Ioana-Raluca Zaman [1] and Stefan Trausan-Matu [1,2,*]

[1] Department of Computer Science and Engineering, Politehnica Bucharest, National University for Science and Technology, 060042 Bucharest, Romania; ioana_raluca.zaman@stud.acs.upb.ro
[2] Research Institute for Artificial Intelligence "Mihai Draganescu" of the Romanian Academy, 050711 Bucharest, Romania
* Correspondence: stefan.trausan@upb.ro

Abstract: Neuropsychiatric disorders affect the lives of individuals from cognitive, emotional, and behavioral aspects, impact the quality of their lives, and even lead to death. Outside the medical area, these diseases have also started to be the subject of investigation in the field of Artificial Intelligence: especially Natural Language Processing (NLP) and Computer Vision. The usage of NLP techniques to understand medical symptoms eases the process of identifying and learning more about language-related aspects of neuropsychiatric conditions, leading to better diagnosis and treatment options. This survey shows the evolution of the detection of linguistic markers specific to a series of neuropsychiatric disorders and symptoms. For each disease or symptom, the article presents a medical description, specific linguistic markers, the results obtained using markers, and datasets. Furthermore, this paper offers a critical analysis of the work undertaken to date and suggests potential directions for future research in the field.

Keywords: neuropsychiatric disorders; depression; dementia; hallucinations; linguistic markers; natural language processing; artificial intelligence

Citation: Zaman, I.-R.; Trausan-Matu, S. A Survey on Using Linguistic Markers for Diagnosing Neuropsychiatric Disorders with Artificial Intelligence. *Information* **2024**, *15*, 123. https://doi.org/10.3390/info15030123

Academic Editor: Arkaitz Zubiaga

Received: 30 January 2024
Revised: 10 February 2024
Accepted: 19 February 2024
Published: 22 February 2024

1. Introduction

In recent years, the advances of Artificial Intelligence (AI) have been seen in different areas of medicine, such as: oncology [1], cardiology [2], endocrinology [3], neurology, and psychiatry [4,5]. Neuropsychiatric disorders are becoming a challenge faced by more and more people nowadays. The conditions include both mental health problems (e.g., depression, anxiety, and schizophrenia) and neurological diseases (e.g., Alzheimer's disease, Parkinson's disease, and epilepsy) [6]. One challenge regarding the detection and the understanding of the disorders is the complexity of the symptoms, which vary from patient to patient but also overlap between certain diseases. Problems related to neuropsychiatric conditions are encountered more and more often, especially due to certain contexts (e.g., epidemics) or for categories exposed to certain factors (e.g., low income) [7]. In a meta-analysis [8], it was discovered that the emergence of the first mental disorder takes place before the age of 14 in over a third of cases (34.6%) and before the age of 25 in almost two-thirds of cases (62.5%). Therefore, particularly for this group of disorders, early detection has a significant impact; applying treatment in time ensures that worsening of the symptoms is slowed down and that the patients have the needed support.

One method for the discovery of new and less obvious symptoms of neuropsychiatric disorders implies studying the language of people, focusing on clues unnoticeable by humans (e.g., the presence or high frequency of specific words, syntactic density, grammar complexity, etc.) [9,10]. In order to find these differences between healthy people and those suffering from certain neuropsychiatric disorders, their speech may be analyzed using AI Natural Language Processing (NLP) methods. This paper presents some of such work and also analyses their approaches.

In recent years, NLP systems have used several Machine Learning (ML) techniques, especially Deep Artificial Neural Networks (DANNs), which perform very well and can include analyzing patients' utterances in neuropsychiatric dialogues [11]. However, people need to trust the decisions made by the ML models, particularly in the medical field. Currently, Transformer-based models [12] have the best performance; however, their results are based only on experience, which can cause the classifications to be based on superficial or invalid criteria [13]. Analyzing conversations from patients and finding patterns in data using AI tools should also allow the interpretability of the results provided by DANNs (which can be seen as black-box models), helping people to have more trust in the AI's contributions to medicine. An online study (N = 415) that measured people's trust in the involvement of AI in medicine based on questionnaires referring to medical scenarios demonstrated that people still have more trust in a doctor than in an AI model [14]. Linguistic markers are characteristics or traits of the text or speech that can provide information about the speaker. These markers can be divided into several categories, for example: grammar markers or lexical semantic markers [15]. If such markers (which can be understood by humans) would be provided for assisting the diagnosis of a patient, the interaction between AI systems based on ML models and doctors would face fewer challenges, and patients would be more open to considering the indications coming from AI.

For a clear and systematic picture aiming to aid the reader with understanding this paper, a concept map illustrating a summary of the main topics and their relations discussed in our work is shown in Figure 1.

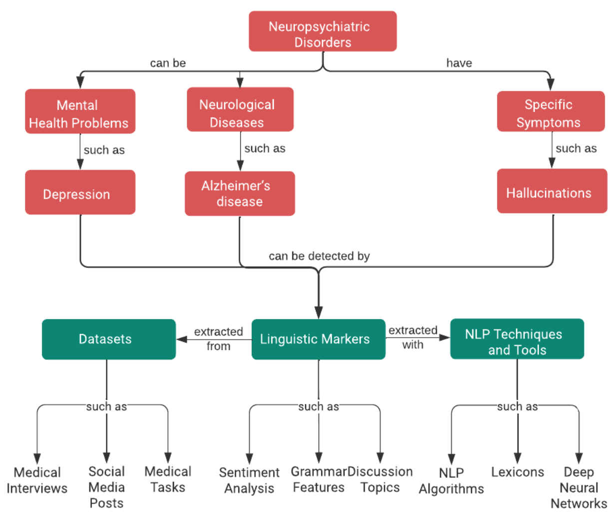


Figure 1. Concept map of the main topics and their relations discussed in the paper.

2. Materials and Methods

A formal literature search was conducted on Google Scholar from 29 August 2023 to 7 September 2023. The used search terms were the following: *("depression" OR "dementia" OR "Alzheimer's disease" OR "hallucinations") AND ("linguistic markers" OR "linguistic analysis" OR "linguistic style")*. There were several inclusion and exclusion criteria used in this study. Firstly, the year of publication was chosen to be at least 2015 in order to analyze only information from recent years when ML and especially DANN architectures dramatically increased the performance of the implemented systems. Another screening criterion involved the domain. This research exclusively incorporated papers related to Computer Science. Consequently, papers addressing neuropsychiatric linguistic analysis through an AI-related approach were taken into account. Research studies originating from diverse domains, such as Medicine, were not taken into account. The ultimate criterion pertained to the language of the publication, wherein only documents available in English were included. This criterion aimed to reduce the complications linked to the process of translation. Subsequently, for the selected papers, their eligibility was tested firstly by the abstracts of the papers, and then full papers were subjected to a detailed review. This process was important to guarantee that only those papers having linguistic analysis of the mentioned disorders or symptoms as their focus were examined. Regarding the datasets, in addition to those utilized in the selected publications, datasets found using the following terms were also selected: *("depression" OR "dementia" OR "Alzheimer's disease" OR "hallucinations") AND ("dataset" OR "corpus")*.

3. Medical Descriptions

This section provides a description of the clinical characteristics, symptoms, and impacts of depression and the neurocognitive disorder (NCD) dementia. As regards NCDs, Alzheimer's disease (AD), the most common form of dementia, will be studied in particular. Moreover, hallucinations will be analyzed, these being a specific symptom of several mental diseases. In addition to that, a comparison between hallucinations produced by humans and those produced artificially by Large Language Models (LLMs—DANNs trained with a huge number of texts) [16] will be illustrated. A deep understanding of the medical symptoms of neuropsychiatric diseases is relevant for effective application of NLPs in studies. Knowing the medical symptoms of the disorders can help with finding associations between certain symptoms and patterns in speech or text. For instance, if two diseases have a common symptom, it could be useful to search for the same linguistic features associated with that symptom for both diseases.

3.1. Depression

Depression, medically known as major depressive disorder (MDD) [17], is the general condition for the class of depressive disorders. It can be seen in several forms, from medication-induced depressive disorder to dysthymia (persistent depressive disorder), and the disease is marked by distinct episodes lasting at least two weeks [17]. The criteria on which the diagnosis of this disease is based are the following: depressed mood (i.e., from feeling hopeless to even feeling irritated, especially for adolescents and children), deeply reduced enjoyment in activities, notable changes in appetite and weight, daily sleep problems (i.e., insomnia or hypersomnia), overwhelming fatigue, feelings of worthlessness or guilt, and in some cases even delusion thoughts, indecisiveness or trouble concentrating, and even suicidal thoughts [17]. Depression's evolution or appearance can be influenced by various risk factors such as: temperament (i.e., neurotic people have a tendency towards anxiety, anger, and emotional instability [18]), environment (i.e., shocking events, especially in childhood), and genetics.

3.2. Dementia and Alzheimer's Disease

Dementia involves conditions wherein the main problem affects the cognitive functions that were acquired rather than present from birth [19]. These conditions affect a category of people over a certain age; at the age of 65, the general prevalence of dementia is approximately 1–2%, and by the age of 85, it is 30% [17]. Dementia is a general term that refers to a series of diseases having various early symptoms depending on which area of the brain is affected [20]. Due to the fact that in the majority of cases of AD, the first part of the brain affected is the hippocampus, the patient initially has problems remembering facts from the recent past. After that, if the amygdala is affected, the person refers to memories more from an emotional point of view than a factual one. As AD progresses, its damage extends to various brain areas and lobes, resulting in a thinner cortex and overall brain shrinkage. The left hemisphere's impairment leads to issues with semantic memory and language, causing difficulty with word retrieval. Damage to the visual system in the temporal lobes hampers face and object recognition, although auditory recognition might still be possible. Right parietal lobe damage affects spatial judgment (e.g., tasks like navigating stairs). Frontal lobe damage results in challenges with decision making, planning, and organizing complex tasks. Despite these losses, long-acquired abilities like procedural memories (e.g., dancing or playing the piano) tend to remain intact, even in advanced stages of AD [20]. Besides AD, there are other forms of dementia such as: vascular dementia, frontotemporal dementia, Lewy body dementia [21], and dementia caused by other diseases (e.g., Huntington's disease or Parkinson's disease) [22].

3.3. Hallucinations

Hallucinations are a symptom present in a variety of diseases, from mental health conditions such as psychotic depression to AD; however, most often they are found in conditions within the spectrum of schizophrenia [23]. This symptom manifests as vivid experiences resembling perceptions but arising without the external triggers [17]. Hallucinations can affect all the senses; however, a survey conducted on 10,448 participants showed that the most frequent are auditory hallucinations (29.5%) (e.g., voices, laughing, and crying), succeeded by visual hallucinations (21.5%) (e.g., shadows and people moving), tactile hallucinations (19.9%) (e.g., being touched and formication), and olfactory hallucinations (17.3%) (e.g., fire, food, and drinks) [24]. Besides these types, there are also gustatory hallucinations (e.g., metallic taste), presence hallucinations (i.e., the feeling that someone is present in the room or behind the subject), and proprioceptive hallucinations (i.e., the feeling that the body is moving) [23].

4. State of the Art

This section provides an overview of the current state of the art in utilizing AI for analyzing neuropsychiatric disorders and their symptoms. The section is structured as follows. The first subsection presents NLP techniques used to understand linguistic signs in conversations about the disorders. It highlights the use of sentiment analysis, topic modeling, and patterns concerning depression, dementia, and hallucinations. In the second subsection, we examine the distinctive linguistic markers associated with each of the diseases. Additionally, a comparison between the differences in linguistic markers between human- and LLM-generated hallucinations is illustrated. The last subsection examines datasets in order to offer insights into selecting suitable resources for NLP-based analysis of neuropsychiatric disorders.

4.1. NLP Techniques

In the field of NLP, a variety of techniques and tools have been employed to investigate linguistic markers associated with neuropsychiatric disorders and have provided valuable insights from the textual data from individuals with these conditions. In this section, the techniques and tools used in state-of-the-art works will be presented. All the studies and techniques mentioned in this section will be presented in more detail in Section 4.2.

Sentiment analysis is a fundamental method utilized to evaluate the emotional tone and sentiments from text or speech. One of the main approaches for sentiment analysis is the usage of lexicons particularly created for this task. Linguistic Inquiry and Word Count (LIWC) [25] is a lexicon-based tool used by researchers for extracting emotional and psychological dimensions. This tool was used to extract features for predicting depression and anxiety from therapy sessions [26] and for the detection of Reddit posts related to depression [27,28]. There are sentiment lexicons specialized for scores of positivity, negativity, and neutrality, such as: SentiWordNet [29] and VADER [30]. The former was utilized by Titla-Tlatelpa et al. [31] for extracting the polarity of posts in order to create a user profile. Moreover, lexicons designed for specific linguistic markers can be created: for instance, the Behaviour Activation lexicon [32].

Topics of discussion represent indicators of certain mental disorders, and they can be identified by selecting key words. One often utilized method for this task is to consider the Term Frequency–Inverse Document Frequency (TF-IDF) [33], which measures the importance of words within a corpus. A smoothed version of TF-IDF was used by

Wolohan et al. [28], who combined it with LIWC and n-grams (sequences of n words) in order to capture word sequences and patterns. Another topic modeling algorithm is Latent Dirichlet Allocation (LDA) [34]; an example of using this method is illustrated in the work of Tadesse et al. [27]. Furthermore, tools such as KHCoder [35] can be utilized for plotting co-occurrence networks or other statistics from texts. The results from part-of-speech (POS) tagging tasks [36] are also relevant markers for neuropsychiatric disorders. For certain disorders (e.g., depression), the tense of verbs is an important clue, and tools such as the Stanford University Time (SUTime) temporal tagging system [37] can be used for analyzing the tenses.

4.2. Linguistic Markers

4.2.1. Depression

Understanding how language can reveal insights about depression has become an area of growing interest that is marked by evolving findings and methodologies. There exists an established association between self-centeredness and depression [38], and this preoccupation with the self is also reflected in linguistic patterns [27]. In the meta-analysis conducted by Tølbøll [39], 26 papers published between the years 2004 and 2019 were examined to study the link between the existence and severity of depression and first-person singular pronouns (e.g., 'I' and 'me'), positive emotion words, and negative emotion words. The conclusions related to the usage of first-person singular pronouns and depression indicated a medium effect (Cohen's d of 0.44) and a positive correlation (Pearson's r of 0.19). One study analyzed Reddit posts from 12,106 users and reconfirmed the link between first-person singular pronouns and depression [28]. Furthermore, the authors found that individuals experiencing depression used more dialogue terms in their posts, specifically addressing the readers using second-person pronouns (e.g., "you") and writing the posts as if talking directly to them. In addition to the linguistic markers discovered, Wolohan et al. [28] created a depression classification system that performed best using LIWC features and n-grams and achieved an F1 score of 0.729. Burkhardt et al. [26] evaluated therapy sessions on Talkspace from over 6500 unique patients and stated correlations between both first-person singular and plural pronouns, which is a conclusion that has also been validated in other research [32] for singular but not plural forms.

Regarding POS tagging, we analyzed the Distress Analysis Interview Corpus/Wizard-of-Oz (DAIC-WOZ) dataset [40] from the University of Southern California and concluded that individuals suffering from depression utilized fewer prepositions, conjunctions, and singular proper nouns. Regarding verb tenses, depressives also have a tendency to use more verbs in gerund or past participle form [41]. Moreover, there are studies supporting future language as an indicator of depression [42]. Using SUTime, the authors discovered that depressed participants refer to future events more distally and think more about the past and future rather than the present. The researchers created an FTR (Future Time Reference) classifier that offers more information about the modality of verbs and achieved an F score over 0.99 on a Reddit post classification task.

Some emotions are more often found in people with certain mental disorders. Tølbøll [39] discovered a strong effect (Cohen's d of 0.72) between depression and negative emotions and a negative correlation (Pearson's r of -0.21) between the disease and the usage of positive words; they also confirmed the correlation between negative emotions and depression for the analyzed conversations. Burkhardt et al. [26] extracted 49 emotion-related features using both LIWC and the GoEmotion dataset [43]. The authors measured the explanatory impact of each feature by using the amount of variance explained by R2 (i.e., the variability of a dependent variable that is explained by an independent variable in a regression model), and the top LIWC features for depression had values in the interval [0.716, 0.729]. With the first tool, sadness, anger, anxiety, and both negative and positive emotions were identified as indicators of the mental disease. The most relevant emotions for depression are: grief, less pride, less excitement, relief, disgust [26], and fear [41]. These were confirmed as well in the work of Tadesse et al. [27], which additionally highlighted:

words of anger and hostility (e.g., hate), suicidal thoughts (e.g., stop-stop, want die), interpersonal processes (e.g., feel alone, lonely), and cues of meaninglessness (e.g., empty, pointless) and hopelessness (e.g., end, need help). Moreover, the usage of absolutist words (e.g., all, always, never) is a marker for depression and its symptoms of suicidal ideation and anxiety [44].

The topics addressed in discussions can be indicators for depression. One method to acquire the topics is by selecting the 100 most-used words with TF-IDF and dividing them into categories. Using this methodology, Wolohan et al. [28] concluded that people suffering from depression more often discuss: therapy (e.g., psychiatrist) and medications (e.g., Prozac) or Reddit, manga, and video games (e.g., Goku). By developing a new lexicon, [26] found that depressed individuals more frequently approach subjects from biology and health categories, and individuals having severe depression talk less about activities and relate them less with positive feelings (e.g., enjoyment, reward). Using LIWC, Tadesse et al. [27] detected correlations (with a Pearson's r coefficients in the interval [0.11, 0.19]) between depressed people and psychological processes such as social processes (e.g., mate, talk), affective processes (e.g., cry, hate), cognitive processes (e.g., know, think), as well as personal concerns such as work, money, and death using. By analyzing depression-related text with LDA, Tadesse et al. [27] selected the top 20 most frequent topics, combined the extracted features with LIWC, bigrams, and an MLP, and obtained an F1 score of 0.93 on a Reddit post classification task. The authors reconfirmed the job topic but also added keywords such as: tired, friends, and broke; they also added sleep and help [41]. In their study, they used the KHCoder tool to identify the topics of the interviews using co-occurrence networks and concluded that in terms of relationships, depressed people talked more about child–parent relationship, while the control group talked more about friends and family, and in terms of jobs, the first category referred more to finding a job, while the second category referred to a dream job. Another approach is to take into consideration the profile (i.e, gender and age) of the speaker when analyzing the text for depression and using age-based classifiers and gender-based classifiers [31]. With this methodology, the authors revealed differences between depressed and non-depressed users per category (e.g., the word calories used in negative contexts can be a marker for depression in young females, while drunk can be used as a marker for senior male users).

4.2.2. Dementia and Alzheimer's Disease

Although it is a field that is just at the beginning, lexical–semantic and acoustic metrics show promising results as digital voice biomarkers for AD [45]. Automating the analysis of vocal tasks like semantic verbal fluency and storytelling provides an undemanding method to support early patient characterization. Some of the speech features we extracted have unique patterns (e.g., the ones related to tone and rhythm). This method could be used as a clear way to tell if someone has depression or mild cognitive problems [46]. Patients with AD have shortfalls with using verbs and nouns [47]: especially verbs during arguments [48]. Using only information from POS tagging, some features (e.g., readability, propositional density, and content density) can be extracted and show promising results for AD classification tasks. For instance, Guerrero et al. [49] achieved an accuracy of 0.817 for Pitt corpus by using a Random Forest (RF) model and, as input, a fusion of features extracted from grammar characteristics, TF-IDF, and Word2Vec (W2V). Eyigoz et al. [50] predicted the beginning of AD by analyzing linguistic characteristics. One of the conclusions they reached was that participants who will be diagnosed with AD had telegraphic speech, writing mistakes, and more repetitions. Telegraphic speech is summarized and contains only the essential words (e.g., nouns and verbs), the connective POS (e.g., articles or adverbs) being omitted. Another characteristic of AD speech was referential specificity: a semantic feature by which unique nouns are differentiated from general nouns (e.g., proper names). More studies support the idea that one of the earliest signs in terms of linguistics is semantic impairment [51,52]. Karlekar et al. [53] identified clusters specific to this disease: namely, clarification questions (e.g., 'Did I say elephant?'), outbursts in speaking and brief answers (e.g., ’oh!’, ‘yes’), and statements starting with interjections (e.g., ‘Well . . . ’, ‘Oh . . . ’). An accuracy of 0.911 was obtained by researchers [53] in an experiment using POS-tagged utterances and a CNN-RNN model. In the case of dementia and AD, the results can be improved by combining linguistic markers with features extracted using Computer Vision (CV) or biomarkers. For instance, Koyama et al. [54] highlighted the role of peripheral inflammatory markers in dementia and AD and found links between increased levels of C-reactive protein or interleukin-6 and dementia. By using CV, neuroimaging techniques can be utilized to detect changes in the brain that are signs of AD or mild cognitive impairment (MCI), such as increased grey matter brain atrophy or hyperactivation within the hippocampus memory network [55].

4.2.3. Hallucinations

Hallucinations from People with Neuropsychiatric Disorders

Hallucinations are a complex phenomenon that can manifest in a unique way from person to person. This symptom, especially an auditory one, is difficult to detect, particularly the moment of its appearance, but using mobile data [56], dictation devices [57], or auditory verbal recognition tasks [58], it is still possible. In accord with a review [59], hallucinations are influenced by cultural aspects such as: religion, race, or environment (e.g., magical delusions exhibited a high frequency in rural areas). Gender is not a factor for auditory hallucinations, but female patients reported experiencing olfactory, tactile, and gustatory hallucinations more frequently [60].

In a study of Dutch language [61], the researchers compared the auditory verbal hallucinations from clinical (i.e., diagnosed with schizophrenia, bipolar disorder, or psychotic disorder) and non-clinical participants and observed that the hallucinations from the first category of participants were more negative (i.e., 34.7% vs. 18.4%); this aspect was also confirmed by [9]. They identified the most frequently encountered semantic classes in the auditory hallucinations in Twitter posts, with the top three being abusive language (e.g., hell), relatives (e.g., son), and religious terms (e.g., prayer), followed by semantic classes related to the sense of hearing (e.g., audio recording, audio and visual media, audio device, or communication tools). Another observation is that tweets containing auditory hallucinations exhibited a greater proportional distribution during the hours of 11 p.m. to 5 a.m. compared to other tweets. By using a set of 100 semantic features, the authors of [9] classified if a Twitter post was related to auditory hallucination and with a Naive Bayes (NB) model reached an AUC of 0.889 and an F2 score of 0.831; the baseline value was 0.711. In this study, the leave-one-out technique showed that the best results were obtained when lexical distribution features were excluded (i.e., an AUC of 0.889 and F2 score of 0.833).

Artificial Hallucinations from ML Models

This subsection presents specific contexts (e.g., tasks or topics of discussion) in which hallucinations were not emitted by humans but were generated from AI systems that generate texts based on LLMs. The models from the studies presented in this section represent a range of representative DANN models, such as: Generative Pre-trained Transformer models (e.g., GPT-2, GPT-4, and ChatGPT) or Transformer-based multimodal models (e.g., VLP, and LXMERT-GDSE). Image captioning is a task in which models may hallucinate; for example, Testoni and Bernardi [62] used the GuessWhat?! game (the goal of the game is for one player to guess the target object by asking the other player binary questions about an image) to force the models to produce hallucinations. The majority of hallucinations manifested in consecutive turns, leading to hypotheses such as previous triggering words and the cascade effect (i.e., the amplification of hallucinations) [62–65]; these phenomena are not present in human hallucinations. However, the models can detect that they are wrong: ChatGPT [66] detects 67.37% of cases and GPT-4 detects 87.03% [65]. Another difference is that in these experiments, the hallucinations appeared more frequently after negative responses; in human dialogues, this is not the case.

Dziri et al. [63] tried to discover the origin of hallucinations in conversational models based on Verbal Response Modes (VRM) [67] and affirmed that the most effective strategies for creating hallucinations were disclosure (i.e., sharing subjective opinions) and edification (i.e., providing objective information). The researchers [63] also studied the level of amplification of hallucinations and concluded that, for instance, GPT2 amplifies full hallucinations by 19.2% in the Wizard of Wikipedia (WOW) dataset. Alkaissi and McFarlane [68] tested ChatGPT for scientific writing, and the model generated nonexistent paper titles with unrelated PubMed IDs and artificial hallucinations [69] regarding medical subjects. Self-contradiction is a form of hallucination that can appear in human hallucinations and LLM-generated hallucinations; for the second type of hallucinations, there are algorithms regarding their evaluation, detection, and mitigation [70]. The authors created a test covering 30 subjects (e.g., Angela Merkel and Mikhail Bulgakov) for the models, and for the detection task, they achieved F1 scores with values up to 0.865.

An overview of each research study presented in Sections 4.1 and 4.2 in chronological order and grouped by medical condition is shown in Tables 1–4 following.

Table 1. Overview of the linguistic markers for depression extracted in the selected papers. Source: Own work.

Dataset Source	Data Type	Linguistic Markers or Features	Tools and Techniques	Year	Ref.
Reddit	Reddit posts	N-grams, topics, psychological and personal concern process features	N-grams, LDA, LIWC	2019	[27]
Reddit	Reddit posts	N-grams, topics, grammatical features, emotions	N-grams, smoothed TF-IDF, LIWC	2019	[28]
Reddit and Twitter	Social media posts	Polarity, gender, age, Bow/BoP representations	Bag of Words (BoW), Bag of Polarities (BoP), SentiWordNet	2021	[31]
Talkspace	Messaging therapy sessions	Grammatical features, topics and emotions	LIWC, GoEmotions	2022	[26]
Reddit	Reddit posts	Temporal features, modal semantics	SUTime	2022	[42]
Public forums	Forum posts	Absolutist index, LIWC features	LIWC, absolutist dictionary	2022	[44]
DAIC-WOZ	Clinical interviews	POS tagging, grammatical features, topics and emotions	NLTK, NRCLex, TextBlob, pyConverse, KHCoder	2023	[41]

Table 2. Overview of the linguistic markers for dementia extracted in the selected papers. Source: Own work.

Dataset Source	Data Type	Linguistic Markers or Features	Tools and Techniques	Year	Ref.
Public blogs	Posts from public blogs	Context-free grammar features, POS tagging, syntactic complexity, psycholinguistic features, vocabulary richness, repetitiveness	Stanford Tagger, Stanford Parser, L2 Syntactic Complexity Analyzer	2017	[10]
Pitt Corpus—Dementia Bank	Cookie Theft picture description task	Grammatical features, POS tagging	Activation clustering, first-derivative saliency heat maps	2018	[53]
Pitt Corpus—Dementia Bank	Cookie Theft picture description task	Word embeddings, grammatical features, POS tagging	Word2Vec, TF-IDF	2020	[49]
FHS study	Cookie Theft picture description task	Word embeddings, grammatical features, POS tagging	GloVe, NLTK	2020	[50]

4.3. Relevant Datasets

This subsection presents an overview of the relevant datasets used in state-of-the-art works in which the mentioned neuropsychiatric disorders were studied. These datasets are utilized for both the detection of the disorder and the extraction of linguistic markers specific to the disease. The data can be obtained by web scraping (e.g., social media posts), artificially (e.g., content generated with an LLM following a pattern), or from medical sources (e.g., dialogues between a patient and a doctor). Another aspect of the data is that it should be gathered over a period of time (e.g., having interviews with a patient over five years periodically), which allows early detection and the evolution of symptoms to be studied.

Table 3. Overview of the linguistic markers for hallucinations from people extracted in the selected papers. Source: Own work.

Dataset Source	Data Type	Linguistic Markers or Features	Tools and Techniques	Year	Ref.
Twitter	Twitter posts	Semantic classes, POS tagging, use of nonstandard language, polarity, key phrases, semantic and lexical features	TweetNLP tagger, MySpell	2016	[9]
Clinical study	Audio reports from sleep onset and REM and non-REM sleep	Grammatical features	Measure of Hallucinatory States (MHS)	2017	[57]
"Do I see ghosts?" Dutch study	Auditory verbal recognition task	Age, gender, education, and the presence of visual, tactile, and olfactory hallucinations	IBM SPSS Statistics	2017	[57]
Clinical study	Electronic health records (EHRs)	Age, gender, race, NLP symptoms	Clinical Record Interactive Search (CRIS)	2020	[60]
Clinical study	Recordings of participants' hallucinations	Grammatical features, emotions, POS tagging	CLAN software, Pattern Python package, Dutch lexicons	2022	[61]
Clinical study	Audio diary by mobile phone with periodic pop-ups asking about the hallucinations	Word embeddings	VGGish model, BERT, ROCKET	2023	[56]

Table 4. Overview of the linguistic markers for artificial hallucinations extracted in the selected papers. Source: Own work.

Dataset Source	Data Type	Linguistic Markers or Features	Tools and Techniques	Year	Ref.
500 randomly selected images	Image captioning task	CHAIR metrics—CHAIR-i and CHAIR-s, METEOR, CIDEr, SPICE	MSCOCO annotations, FC model, LRCN, Att2In, TopDown, TopDown-BB, Neural Baby Talk (NBT)	2018	[64]
GuessWhat?! game	Utterances from GuessWhat?! game	CHAIR metrics—CHAIR-i and CHAIR-s, analysis of hallucinations	MSCOCO annotations, BL, GDSE, LXMERT-GDSE, VLP	2021	[62]
Wizard of Wikipedia (WOW), CMUDOG, TOPICALCHAT	Dialogues between two speakers	Hallucination rate, entailment rate, Verbal Response Modes (VRMs)	GPT2, DoHA, CTRL	2022	[63]
3 new datasets consisting of yes/no questions	QA task answers	Snowballing of hallucinations, hallucination detection, LM (in)consistency	ChatGPT, GPT-4	2023	[65]
Dataset consisting of generated encyclopedic text descriptions for Wikipedia topics	Description task	Average no. of sentences, perplexity, self-contradictory features	ChatGPT, GPT-4, Llama2-70B-Chat, Vicuna-13B	2023	[70]

4.3.1. Depression

In our depression study [41], we used the DAIC-WOZ dataset, which is a corpus containing the conversations between an agent Ellie and 189 participants: 133 non-depressed and 56 depressed. The agent is human-controlled and operates based on a predefined set of questions for the conversations. In order to label the participants, the Patient Health Questionnaire-8 (PHQ-8) is utilized, and for each entry, the database contains: an audio recording and transcript of the conversation, the responses for the PHQ-8, the gender of the participant, and metadata (i.e., non-verbal and verbal features). To minimize the effects of dataset imbalance, we created an additional subset of similar conversations of depressed patients using ChatGPT. Depression-related challenges are another source for datasets; for instance, DepreSym is a corpus created from the eRisk 2023 Lab.

A methodology used to retrieve medical dialogues is through online platforms, such as those specialized for therapy sessions (e.g., Talkspace) [26], or forums [44]. Extracting social media posts is a popular method for constructing new corpora; for instance, Shen et al. [71] developed from Twitter posts a dataset with three subsets: depression, non-depression, and depression candidate. Several researchers [31,72] have also used Twitter as a source for their data. Another social platform from which data are collected is Reddit [27,28,73].

4.3.2. Dementia and Alzheimer's Disease

One method to create a dataset for dementia or AD is from tasks designed to emphasize the particular symptoms of the conditions, such as "Boston Cookie Theft" (a task in which the participants were asked to describe a given picture) or a recall test (a task in which the participants were asked to recall attributes of a previously shown story or picture). DementiaBank [74] is a database of corpora containing video, audio, and transcribed materials for AD, Mild Cognitive Impairment (MCI), Primary Progressive Aphasia (PPA), and dementia in multiple languages (e.g., English, German, and Mandarin).

The Framingham Heart Study (FHS) is a study started in 1948 and which continues to this day. Its aim is to discover factors that play a role in the development of cardiovascular disease (CVD). However, it also contains recordings of participants suffering from conditions such as AD, MCI, or dementia. Researchers have used the data from this study in order to detect linguistic markers that can be utilized for the early prediction of the previously mentioned diseases [50,75]. Dementia and AD can also be studied in an online environment, such as blog posts. For instance, Masrani et al. [10] created the Dementia Blog Corpus by scraping 2805 posts from 6 public blogs, and the authors of [76] studied dementia using data from Twitter.

4.3.3. Hallucinations

One of the signs of the presence of hallucinations in speech can be the unreliability of the facts presented in the conversation. To highlight this sign, Liu et al. [77] created HaDeS (HAllucination DEtection dataSet), a corpus built by perturbing raw texts from the WIKI-40B [78] dataset using BERT [79], and then checked the validity of the hallucinations with human annotators. The authors of [80] studied the correlations between hallucinations and psychological experiences using a dataset containing 10,933 narratives from patients diagnosed with mental illnesses (e.g., schizophrenia or obsessive compulsive disorder); the data had been previously collected by the authors [81].

Artificial hallucinations are usually generated from conversational agents by using certain games [62] or by addressing sensitive subjects such as religion, politics, or conspiracy ideas. The Medical Domain Hallucination Test (Med-HALT) [82] is a collection of seven datasets containing hallucination from LLMs in the medical field. The datasets are based on two tasks: more precisely, the Reasoning Hallucination Test (RHT) (i.e., a task in which the model has to choose an option from a set of options for questions) and the Memory Hallucination Test (MHT) (i.e., a task in which the model has to retrieve information from a given input). The data utilized as input for the models are questions from medical exams (e.g., the residency examination from Spain and the United States Medical Licensing Examination (USMILE)) and PubMed.

5. Discussion and Challenges

A key area for the improvement of the discussed approaches involves the expansion and refinement of existing datasets and the development of new corpora; for instance, more emphasis should be on collecting data periodically over a longer period of time to study the evolution of diseases and to find the most relevant linguistic symptoms. Additionally, the building of diverse datasets covering various demographic groups and different stages of these disorders could improve the results. Integrating multimodal approaches that combine linguistic markers with medical imaging or other biological signals could offer a more comprehensive understanding of these disorders. Correlating linguistic patterns with physiological and visual data may amplify the accuracy of early diagnosis and prediction.

Considering that ethics is indispensable in a project using data from people, especially such sensitive data as those from patients suffering from neuropsychiatric disorders, various aspects such as algorithmic fairness, biases, data privacy, informed consent to use, safety, and transparency [83] have to be taken into account for a project to be ethically valid. Fulfilling all these conditions can create difficulties in a project, such as non-cooperation and lack of patient consent for the collection of new data or legal challenges that require the involvement of legal professionals. Another problem is represented by the limited access to such data; for example, a significant part of medical datasets are accessible only to researchers affiliated with certain universities or having certain citizenship.

Another aspect is the interpretability of the results. Especially in the medical field, each diagnostic offered by a model should be argued and explained; the Explainable Artificial Intelligence (XAI) [84] domain is at the beginning of development. DANN models perform better than classic ML models (e.g., SVM, RF, and NB), yet they have the disadvantage of a black-box nature; therefore, a trade-off between interpretability and performance is still necessary [85]. An application based on a DANN model, particularly in the medical field, should have the following characteristics: fairness (i.e., ensure that the model does not discriminate), accountability (i.e., decide who is responsible for the decision), transparency (i.e., interpretability and understandability of the model's decisions), privacy, and robustness [86]. Meeting these criteria can pose challenges in situations where data are scarce or originate exclusively from a specific category, such as being restricted to more-developed countries. Lastly, future research should concentrate on refining these linguistic markers and models to support real-time diagnostics, early intervention, and treatment monitoring for neuropsychiatric disorders. Validation studies in clinical settings are necessary to evaluate the reliability and generalizability of these linguistic models.

The generalizability of the presented research findings can represent a potential challenge to the use of AI in the medical field, especially in such subjective areas as mental or psychiatric illnesses. A wrong generalization can be generated from the beginning by using data limited only to certain categories of people. For example, a study [87] performed on 94 adults demonstrated the link between depression and demographics and clinical and personality traits. Larøi et al. [88] studied the influence that culture (i.e., multiple factors such as religion and political beliefs) has on hallucinations. Taking these into account, the existence of a heterogeneous dataset that includes as many different elements as possible would contribute to the discovery of linguistic symptoms that are as general as possible. Another perspective from which this problem can be viewed is that of the model. As mentioned, the models with the best performance are based on DANN; these types of models are prone to unreliable results based on incorrect criteria if the training data are biased.

A fundamental theoretical problem of DANNs, which are now considered the best approaches for NLP and were used in the research discussed herein, is that transformers and neural networks, in general, are based on an empiricist paradigm. It should be mentioned that to obtain the best results, there is a need to integrate empiricist with nativist perspectives, the latter being used in symbolic, knowledge-based AI. These two paradigms correspond, in fact, to the two main, opposing philosophical schools of thought that have Aristotle and Plato as parents, with the latter being also advocated by Chomsky [13].

6. Conclusions

This survey demonstrates the potential of NLP for identifying linguistic patterns related to neuropsychiatric disorders. Advanced methods have identified specific linguistic traits and offer promising results for the early recognition and treatment of these disorders. The identified markers (e.g., specific emotions and verb tenses) linked to conditions such as depression, dementia, or hallucinations represent cues that are sometimes undiscoverable by conventional diagnosis methods. This interdisciplinary field that combines linguistic analysis, medical science, AI, and multimodal approaches offers a promising direction for future research and practical applications and will potentially revolutionize early detection, treatment, and care for neuropsychiatric disorders. However, despite these advancements, future efforts are needed to enhance AI model accuracy and interpretability. At last, but of course not at least, it should be mentioned that the very important ethical aspects need be permanently considered, and it should also be taken into account that AI ethics is now a major subject of discussion, research, and regulation [89–91].

Author Contributions: Conceptualization, I.-R.Z. and S.T.-M.; methodology, I.-R.Z. and S.T.-M.; validation, S.T.-M.; investigation, I.-R.Z. and S.T.-M.; resources, I.-R.Z. and S.T.-M.; data curation, I.-R.Z. and S.T.-M.; writing—original draft preparation, I.-R.Z.; writing—review and editing, S.T.-M.; supervision, S.T.-M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: We would like to thank the authors of all datasets described in this paper for making the data available to the community for research purposes.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Luchini, C.; Pea, A.; Scarpa, A. Artificial intelligence in oncology: Current applications and future perspectives. *Br. J. Cancer* **2021**, *126*, 4–9. [CrossRef]
2. Gupta, M.; Kunal, S.; Mp, G.; Gupta, A.; Yadav, R.K. Artificial intelligence in cardiology: The past, present and future. *Indian Heart J.* **2022**, *74*, 265–269. [CrossRef] [PubMed]
3. Giorgini, F.A.; Di Dalmazi, G.; Diciotti, S. Artificial intelligence in endocrinology: A comprehensive review. Journal of Endocrinological Investigation. *J. Endocrinol. Investig.* **2023** . [CrossRef]
4. Zhong, Y.; Chen, Y.; Zhang, Y.; Lyu, Y.; Yin, J.; Yujun, G. The Artificial intelligence large language models and neuropsychiatry practice and research ethic. *Asian J. Psychiatry* **2023**, *84*, 103577. [CrossRef] [PubMed]
5. Rainey, S.; Erden, Y.J. Correcting the brain? the convergence of neuroscience, neurotechnology, psychiatry, and artificial intelligence. *Sci. Eng. Ethics* **2020**, *26*, 2439–2454. [CrossRef] [PubMed]
6. World Health Organization (WHO). Available online: https://platform.who.int (accessed on 10 November 2023).
7. Leung, C.M.C.; Ho, M.K.; Bharwani, A.A.; Cogo-Moreira, H.; Wang, Y.; Chow, M.S.C.; Fan, X.; Galea, S.; Leung, G.M.; Ni, M.Y. Mental disorders following COVID-19 and other epidemics: A systematic review and meta-analysis. *Transl. Psychiatry* **2022**, *12*, 205. [CrossRef]
8. Solmi, M.; Radua, J.; Olivola, M.; Croce, E.; Soardo, L.; de Pablo, G.S.; Shin, J.I.; Kirkbride, J.B.; Jones, P.; Kim, J.H.; et al. Age at onset of mental disorders worldwide: Large-scale meta-analysis of 192 epidemiological studies. *Mol. Psychiatry* **2021**, *27*, 281–295. [CrossRef]
9. Belousov, M.; Dinev, M.; Morris, R.;Berry, N.; Bucci, S.; Nenadic, G. Mining Auditory Hallucinations from Unsolicited Twitter Posts. 2016. Available online: https://ep.liu.se/en/conference-article.aspx?series=&issue=128&Article_No=5 (accessed on 29 January 2024).
10. Masrani, V.; Murray, G.; Field, T.; Carenini, G. Detecting dementia through retrospective analysis of routine blog posts by bloggers with dementia. *ACL Anthol.* **2017**. Available online: https://aclanthology.org/W17-2329/ (accessed on 29 January 2024).
11. Yoon, J.; Kang, C.; Kim, S.; Han, J. D-VLog: Multimodal vlog dataset for Depression Detection. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 12226–12234. [CrossRef]
12. Vaswani, A.; Shazeer, N.; Parmar, N. Attention Is All you Need. Part of Advances in Neural Information Processing Systems 30. 2017. Available online: https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html (accessed on 29 January 2024).
13. Ranaldi, L.; Pucci, G. Knowing Knowledge: Epistemological Study of knowledge in transformers. *Appl. Sci.* **2023**, *13*, 677. [CrossRef] [CrossRef]
14. Yokoi, R.; Eguchi, Y.; Fujita, T.; Nakayachi, K. Artificial Intelligence Is Trusted Less than a Doctor in Medical Treatment Decisions: Influence of Perceived Care and Value Similarity. *Int. J. Hum.–Comput. Interact.* **2020**, *37*, 981–990. [CrossRef]
15. Kozhemyakova, E.A.; Petukhova, M.E.; Simulina, S.; Ivanova, A.M.; Zakharova, A. Linguistic markers of native speakers. In Proceedings of the International Conference "Topical Problems of Philology and Didactics: Interdisciplinary Approach in Humanities and Social Sciences" (TPHD 2018), 2019 . [CrossRef]

16. Rawte, V.; Chakraborty, S.; Pathak, A.; Sarkar, A.; Tonmoy, S.M.T.I.; Chadha, A.; Sheth, A.P.; Das, A. The troubling emergence of hallucination in large language models—An extensive definition, quantification, and prescriptive remediations. *arXiv* **2023**, arXiv:abs/2310.04988.
17. American Psychiatric Association. *Diagnostic and Statistical Manual of Mental Disorders (DSM-5-TR) (5TH ED)*; British Library; American Psychiatric Association: Washington, DC, USA, 2013.
18. Widiger, T.A.; Oltmanns, J.R. Neuroticism is a fundamental domain of personality with enormous public health implications. *World Psychiatry Off. J. World Psychiatr. Assoc. (WPA)* **2017**, *16*, 144–145. [CrossRef]
19. Dementia Australia. Available online: https://www.dementia.org.au (accessed on 11 November 2023).
20. Alzheimer's Society. Available online: https://www.alzheimers.org.uk (accessed on 11 November 2023).
21. Dementia UK. Available online: https://www.dementiauk.org (accessed on 11 November 2023).
22. Alzheimer's Association. Available online: https://www.alz.org (accessed on 11 November 2023).
23. Cleveland Clinic. Available online: https://my.clevelandclinic.org (accessed on 11 November 2023).
24. Linszen, M.M.J.; de Boer, J.N.; Schutte, M.J.L.; Begemann, M.J.H.; de Vries, J.; Koops, S.; Blom, R.E.; Bohlken, M.M.; Heringa, S.M.; Blom, J.D.; et al. Occurrence and phenomenology of hallucinations in the general population: A large online survey. *Schizophrenia* **2022**, *8*, 41. [CrossRef]
25. Tausczik, Y.R.; Pennebaker, J.W. The psychological meaning of words: LIWC and computerized text analysis methods. *J. Lang. Soc. Psychol.* **2019**, *29*, 24–54. [CrossRef]
26. Burkhardt, H.; Pullmann, M.; Hull, T.; Aren, P.; Cohen, T. Comparing emotion feature extraction approaches for predicting depression and anxiety. In Proceedings of the Eighth Workshop on Computational Linguistics and Clinical Psychology, Seattle, USA, July 2022 . [CrossRef]
27. Tadesse, M.M.; Lin, H.; Xu, B.; Yang, L. Detection of depression-related posts in Reddit social media forum. *IEEE Access* **2019**, *7*, 44883–44893. [CrossRef]
28. Wolohan, J.; Hiraga, M.; Mukherjee, A.; Sayyed, Z.A.; Millard, M. Detecting linguistic traces of depression in topic-restricted text: Attending to self-stigmatized depression with NLP. In Proceedings of the First International Workshop on Language Cognition and Computational Models, Santa Fe, NM, USA, August 2018 ; pp. 11–21. Available online: https://aclanthology.org/W18-4102/ (accessed on 29 January 2024).
29. Baccianella, S.; Esuli, A.; Sebastiani, F. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. *ACL Anthol.* **2010**, *10*, 2200–2204. https://aclanthology.org/L10-1531/.
30. Hutto, C.; Gilbert, E. VADER: A parsimonious rule-based model for sentiment analysis of social media text. *Proc. Int. AAAI Conf. Web Soc. Media* **2014**, *8*, 216–225. [CrossRef]
31. Titla-Tlatelpa, J.d.J.; Ortega-Mendoza, R.M.; Montes-y-Gómez, M.; Villaseñor-Pineda, L. A profile-based sentiment-aware approach for depression detection in social media. *EPJ Data Sci.* **2021**, *10*, 54. [CrossRef]
32. Burkhardt, H.A.; Alexopoulos, G.S.; Pullmann, M.D.; Hull, T.D.; Areán, P.A.; Cohen, T. *Behavioral Activation and Depression Symptomatology: Longitudinal Assessment of Linguistic Indicators in Text-Based Therapy Sessions (Preprint)*; JMIR Publications Inc.: Toronto, ON, Canada, 2021. [CrossRef]
33. Jones, K.S. A statistical interpretation of term specificity and its application in retrieval. *J. Doc.* **1972**, *28*, 11–21. [CrossRef]
34. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
35. Higuch, K. A Two-Step Approach to Quantitative Content Analysis: KH Coder Tutorial Using Anne of Green Gables (Part II). *Ritsumeikan Soc. Sci. Rev.* **2017**, *52*, 77–91.
36. Toutanova, K.; Manning, C.D. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics, Hong Kong, China, 7–8 October 2000. [CrossRef]
37. Chang, A.X.; Manning, C.D. SUTIME: A library for recognizing and normalizing time expressions. In Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, 21–27 May 2012; pp. 3735–3740.
38. Wegemer, C.M. Selflessness, depression, and neuroticism: An interactionist perspective on the effects of self-transcendence, perspective-taking, and materialism. *Front. Psychol.* **2020**, *11*, 523950. [CrossRef] [PubMed]
39. Tølbøll, K.B. Linguistic features in depression: A meta-analysis. *J. Lang.-Work.–Sprogvidenskabeligt Stud.* **2019**, *4*, 39–59.
40. Gratch, J.; Artstein, R.; Lucas, G.; Stratou, G.; Scherer, S.; Nazarian, A.; Wood, R.; Boberg, J.; DeVault, D.; Marsella, S.; et al. The Distress Analysis Interview Corpus of human and computer interviews. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), Reykjavik, Iceland, May 2014; pp. 3123–3128. Available online: https://aclanthology.org/L14-1421/ (accessed on 29 January 2024).
41. Zaman, I.-R.; Trausan-Matu, S.; Rebedea, T. Analysis of medical conversations for the detection of depression. In Proceedings of the 19th International Conference on Human-Computer Interaction—RoCHI 2023, 2023; pp. 15–22. Available online: http://rochi.utcluj.ro/articole/11/RoCHI2023-Zaman.pdf (accessed on 29 January 2024).
42. Robertson, C.; Carney, J.; Trudell, S. Language about the future on social media as a novel marker of anxiety and depression: A big-data and experimental analysis. *Curr. Res. Behav. Sci.* **2023**, *4*, 100104. [CrossRef] [PubMed]
43. Demszky, D.; Movshovitz-Attias, D.; Ko, J.; Cowen, A.; Nemade, G.; Ravi, S. GoEmotions: A dataset of fine-grained emotions. *arXiv* **2020**, arXiv:2005.00547.

44. Al-Mosaiwi, M.; Johnstone, T. In an absolute state: Elevated use of absolutist words is a marker specific to anxiety, depression, and suicidal ideation. *Clin. Psychol. Sci.* **2018**, *6*, 529–542. [CrossRef] [PubMed]
45. Lanzi, A.M.; Saylor, A.K.; Fromm, D.; Liu, H.;MacWhinney, B.; Cohen, M.L. DementiaBank: Theoretical rationale, protocol, and illustrative analyses. *Am. J. Speech-Lang. Pathol.* **2023**, *32*, 426–438. [CrossRef] [PubMed]
46. Labbé, C.; König, A.; Lindsay, H.; Linz, N.; Tröger, J.; Robert, P. Dementia vs. Depression: New methods for differential diagnosis using automatic speech analysis. *Alzheimer's Dement. J. Alzheimer's Assoc.* **2022**, *18*, e064486. [CrossRef]
47. Almor, A.; Aronoff, J.M.; MacDonald, M.C.; Gonnerman, L.M.; Kempler, D.; Hintiryan, H.; Hayes, U.L.; Arunachalam, S.; Andersen, E.S. A common mechanism in verb and noun naming deficits in Alzheimer's patients. *Brain Lang.* **2009**, *111*, 8–19. [CrossRef]
48. Kim, M.; Thompson, C.K. Verb deficits in Alzheimer's disease and agrammatism: Implications for lexical organization. *Brain Lang.* **2004**, *88*, 1–20. [CrossRef]
49. Guerrero-Cristancho, J.; Vasquez, J.C.; Orozco, J.R. Word-Embeddings and grammar features to detect language disorders in alzheimer's disease patients. *Inst. Tecnol. Metrop.* **2020**, *23*, 63–75. Available online: https://www.redalyc.org/journal/3442/344262603030/html/ (accessed on 29 January 2024). [CrossRef]
50. Eyigoz, E.; Mathur, S.; Santamaria, M.; Cecchi, G.; Naylor, M. Linguistic markers predict onset of Alzheimer's disease. *EClinicalMedicine* **2020**, *28*, 100583. [CrossRef]
51. Martin, A.; Fedio, P. Word production and comprehension in Alzheimer's diseáse: The breakdown of semantic knowledge. *Brain Lang.* **1983**, *19*, 124–141. [CrossRef] [PubMed]
52. Appell, J.; Kertesz, A.; Fisman, M. A study of language functioning in Alzheimer patients. *Brain Lang.* **1982**, *17*, 73–91. [CrossRef] [PubMed]
53. Karlekar, S.; Niu, T.; Bansal, M. Detecting linguistic characteristics of Alzheimer's dementia by interpreting neural models. *ACL Anthol.* **2018**. Available online: https://aclanthology.org/N18-2110/ (accessed on 29 January 2024).
54. Koyama, A.; O'Brien, J.; Weuve, J.; Blacker, D.; Metti, A.L.; Yaffe, K. The role of peripheral inflammatory markers in dementia and Alzheimer's disease: A meta-analysis. *J. Gerontol. Ser. Biol. Sci. Med. Sci.* **2012**, *68*, 433–440. [CrossRef]
55. Ewers, M.; Sperling, R.A.; Klunk, W.E.; Weiner, M.W.; Hampel, H. Neuroimaging markers for the prediction and early diagnosis of Alzheimer's disease dementia. *Trends Neurosci.* **2011**, *34*, 430–442. [CrossRef]
56. Mirjafari, S.; Nepal, S.; Wang, W.; Campbell, A.T. Using mobile data and deep models to assess auditory verbal hallucinations. *arXiv* **2023**, arXiv:2304.11049.
57. Speth, C.; Speth, J. A new measure of hallucinatory states and a discussion of REM sleep dreaming as a virtual laboratory for the rehearsal of embodied cognition. *Cogn. Sci.* **2017**, *42*, 311–333. [CrossRef]
58. de Boer, J.N.; Linszen, M.M.J.; de Vries, J.; Schutte, M.J.L.; Begemann, M.J.H.; Heringa, S.M.; Bohlken, M.M.; Hugdahl, K.; Aleman, A.;Wijnen, F.N.K.; et al. Auditory hallucinations, top-down processing and language perception: A general population study. *Psychol. Med.* **2019**, *49*, 2772–2780. [CrossRef]
59. Viswanath, B.; Chaturvedi, S.K. Cultural aspects of major mental disorders: A critical review from an Indian perspective. *Indian J. Psychol. Med.* **2012**, *34*, 306–312. [CrossRef]
60. Irving, J.; Colling, C.; Shetty, H.;Pritchard, M.; Stewart, R.; Fusar-Poli, P.; McGuire, P.; Patel, R. Gender differences in clinical presentation and illicit substance use during first episode psychosis: A natural language processing, electronic case register study. *BMJ Open* **2021**, *11*, e042949. [CrossRef] [PubMed]
61. de Boer, J.N.; Corona Hernández, H.; Gerritse, F.; Brederoo, S.G.; Wijnen, F.N.K.; Sommer, I.E. Negative content in auditory verbal hallucinations: A natural language processing approach. *Cogn. Neuropsychiatry* **2021**, *27*, 139–149. [CrossRef]
62. Testoni, A.; Bernardi, R. "I've seen things you people wouldn't believe": Hallucinating entities in guesswhat?! In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop, Online, August 2021. [CrossRef]
63. Dziri, N.; Milton, S.; Yu, M.; Zaiane, O.; Reddy, S. On the Origin of Hallucinations in Conversational Models: Is it the Datasets or the Models? In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Seattle, DC, USA, 10–15 July 2022. [CrossRef]
64. Rohrbach, A.; Hendricks, L.A.; Burns, K.; Darrell, T.; Saenko, K. Object hallucination in image captioning. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018. [CrossRef]
65. Zhang, M.; Press, O.; Merrill, W.; Liu, A.; Smith, N.A. How language model hallucinations can snowball. *arXiv* **2023**, arXiv:2305.13534. https://arxiv.org/abs/2305.13534.
66. OpenAI. ChatGPT 2023. Available online: https://chat.openai.com/chat (accessed on 29 January 2024).
67. Stiles, W.B. Verbal response modes taxonomy. In *The Cambridge Handbook of Group Interaction Analysis*; Cambridge University Press: Cambridge, UK, 2019; pp. 630–638. [CrossRef]
68. Alkaissi, H.; McFarlane, S.I. Artificial Hallucinations in ChatGPT: Implications in Scientific Writing. 2023. Available online: https://pubmed.ncbi.nlm.nih.gov/36811129/ (accessed on 29 January 2024).
69. Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.J.; Madotto, A.; Fung, P. Survey of hallucination in natural language generation. *ACM Comput. Surv.* **2023**, *55*, 1–38. [CrossRef]

70. Mündler, N.; He, J.; Jenko, S.; Vechev, M. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. *arXiv* **2023**, arXiv:2305.15852.
71. Shen, G.; Jia, J.; Nie, L.; Feng, F.; Zhang, C.; Hu, T.; Chua, T.-S.; Zhu, W. Depression detection via harvesting social media: A multimodal dictionary learning solution. In Proceedings of the IJCAI, Melbourne, Canada, 2017; pp. 3838–3844. Available online: https://www.ijcai.org/proceedings/2017/536 (accessed on 29 January 2024).
72. Megan, E.; Wittenborn, A.; Bogen, K.; McCauley, H. #MyDepressionLooksLike: Examining public discourse about depression on twitter. *JMIR Ment. Health* **2017**, *4*, e8141. [CrossRef]
73. Cohan, A.; Desmet, B.; Yates, A.; Soldaini, L.; MacAvaney, S.; Goharian, N. SMHD: A large-scale resource for exploring online language usage for multiple mental health conditions. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, August 2018; pp. 1485–1497. Available online: https://aclanthology.org/C18-1126/ (accessed on 29 January 2024).
74. Becker, J.T.; Boiler, F.; Lopez, O.; Saxton, J.; McGonigle, K.L. The Natural History of Alzheimer's Disease: Description of study cohort and accuracy of diagnosis. *Arch. Neurol.* **1994**, *51*, 585–594. [CrossRef] [PubMed]
75. Xue, C.; Karjadi, C.; Paschalidis, I.C.; Au, R.; Kolachalama, V.B. Detection of dementia on voice recordings using deep learning: A Framingham Heart Study. *Alzheimer's Res. Ther.* **2021**, *13*, 1–15. [CrossRef]
76. Azizi, M.; Jamali, A.A.; Spiteri, R. Identifying Tweets Relevant to Dementia and COVID-19: A Machine Learning Approach (Preprint). *PREPRINT-SSRN* **2023**. Available online: https://pesquisa.bvsalud.org/global-literature-on-novel-coronavirus-2019-ncov/resource/pt/ppzbmed-10.2139.ssrn.4458777 (accessed on 29 January 2024).
77. Liu, T.; Zhang, Y.; Brockett, C.; Mao, Y.; Sui, Z.; Chen, W.; Dolan, W.B. A token-level reference-free hallucination detection benchmark for free-form text generation. *ACL Anthol.* **2022**. Available online: hhttps://aclanthology.org/2022.acl-long.464/ (accessed on 29 January 2024).
78. Guo, M.; Dai, Z.; Vrandečić, D.; Al-Rfou', R. Wiki-40B: Multilingual language model dataset. In Proceedings of the Twelfth Language Resources and Evaluation Conference, Marseille, France, May 2020; pp. 2440–2452. Available online: https://aclanthology.org/2020.lrec-1.297/ (accessed on 29 January 2024).
79. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
80. Ghosh, C.C.; McVicar, D.; Davidson, G.; Shannon, C.; Armour, C. Exploring the associations between auditory hallucinations and psychopathological experiences in 10,933 patient narratives: Moving beyond diagnostic categories and surveys. *BMC Psychiatry* **2023**, *23*, 1–10. [CrossRef]
81. Ghosh, C.C.; McVicar, D.; Davidson, G.; Shannon, C. Measuring diagnostic heterogeneity using text-mining of the lived experiences of patients. *BMC Psychiatry* **2021**, *21*, 1–12. [CrossRef] [PubMed]
82. Pal, A.; Umapathi, L.K.; Sankarasubbu, M. Med-HALT: Medical domain hallucination test for large language models. *arXiv* **2023**, arXiv:2307.15343.
83. Gerke, S.; Minssen, T.; Cohen, G. Ethical and Legal Challenges of Artificial Intelligence-Driven Healthcare. 2020. Available online: https://www.sciencedirect.com/science/article/pii/B9780128184387000125 (accessed on 29 January 2024).
84. Gohel, P.; Singh, P.; Mohanty, M. Explainable AI: Current status and future directions. *arXiv* **2021**, arXiv:2107.07045.
85. Arrieta, A.B.; Díaz-Rodríguez, N.; Del Ser, J.; Bennetot, A.; Tabik, S.; Barbado, A.; Garcia, S.; Gil-López, S.; Molina, D.; Benjamins, R.; et al. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI. *arXiv* **2019**, arXiv:1910.10045.
86. Wei, W.; Landay, J. CS 335: Fair, Accountable, and Transparent (FAccT) Deep Learning, Stanford University, April 2020. Available online: https://hci.stanford.edu/courses/cs335/2020/sp/lec1.pdf (accessed on 29 January 2024).
87. Enns, M.W.; Larsen, D.K.; Cox, B.J. Discrepancies between self and observer ratings of depression. *J. Affect. Disord.* **2000**, *60*, 33–41. [CrossRef]
88. Larøi, F.; Luhrmann, T.M.; Bell, V.; Christian, W.A.; Deshpande, S.N.; Fernyhough, C.; Jenkins, J.H.; Woods, A. Culture and Hallucinations: Overview and Future Directions. *Schizophr. Bull.* **2014**, *40*, S213–S220. [CrossRef]
89. Council of the EU Artificial Intelligence Act: Council and Parliament Strike a Deal on the First Rules for AI in the World. Available online: https://www.consilium.europa.eu (accessed on 28 January 2024).
90. AI HLEG. Ethics Guidelines for Trustworthy AI. Available online: https://ec.europa.eu (accessed on 28 January 2024).
91. European Parliament. EU Guidelines on Ethics in Artificial Intelligence: Context and Implementation. Available online: https://www.europarl.europa.eu (accessed on 28 January 2024).

information

Article

Linguistic Profiling of Text Genres: An Exploration of Fictional vs. Non-Fictional Texts

Akshay Mendhakar

Faculty of Applied Linguistics, Uniwersytet Warszawsk, 00-927 Warszawa, Poland; a.mendhakar@uw.edu.pl

Abstract: Texts are composed for multiple audiences and for numerous purposes. Each form of text follows a set of guidelines and structure to serve the purpose of writing. A common way of grouping texts is into text types. Describing these text types in terms of their linguistic characteristics is called 'linguistic profiling of texts'. In this paper, we highlight the linguistic features that characterize a text type. The findings of the present study highlight the importance of parts of speech distribution and tenses as the most important microscopic linguistic characteristics of the text. Additionally, we demonstrate the importance of other linguistic characteristics of texts and their relative importance (top 25th, 50th and 75th percentile) in linguistic profiling. The results are discussed with the use case of genre and subgenre classifications with classification accuracies of 89 and 73 percentile, respectively.

Keywords: genres; subgenres; linguistic profiling; text; NLP

Citation: Mendhakar, A. Linguistic Profiling of Text Genres: An Exploration of Fictional vs. Non-Fictional Texts. *Information* **2022**, *13*, 357. https://doi.org/10.3390/info13080357

Academic Editor: Peter Revesz

Received: 21 June 2022
Accepted: 22 July 2022
Published: 26 July 2022

1. Introduction

With the advancement in computers and their processing abilities, powerful algorithms that can process complex data in seconds have led to the development of modern-day natural language processing (NLP) algorithms. Present-day NLP techniques focus on both the context and form of the text rather than focusing on just one of them.

The development of sophisticated NLP pipelines and the availability of multiple large-scale corpora have given rise to a new range of data-driven NLP tools. These modern tools can be used to answer classical linguistic research topics and many more topics with relative ease. By accomplishing this, we can highlight the set of linguistic variables which are suited for the given task and try training machine learning algorithms to build models for a given task. These models represent a text type based on its linguistic features and can be used for solving complex linguistic problems when coupled with complex statistical methods. One such classical linguistics problem is identifying text patterns and highlighting the linguistic characteristics/linguistic profiling [1]. This traditional question has led to multiple advanced concepts such as genre identification [2], identification of one's native language [3], author identification [4], author attribution [5], author verification [6] etc.

Similar complex linguistic use cases have given rise to areas such as computational register analysis [7], which looks at the register and genre variation from a functional spectrum of context-driven linguistic differences; computational sociolinguistics [8], which focuses on the social dimension of language and the underlying diversity associated with it; computational stylometry is aimed at extracting knowledge from texts to verify and attribute authorship [9]; and many more. While classical stylometric techniques place a special emphasis on identifying the most salient or the rarest feature in a text, modern techniques can uncover patterns even in smaller segments of text [1,10]. Identifying a specific linguistic profile of different text types can be used for classification tasks and measurement of readability [11].

2. Literature Review

The concept of linguistic profiling for identifying specific features is not new and has been attempted by multiple researchers. However, the usage of linguistic profiling to understand the genre variation computationally is the focus of this review. Ref [12] was the first to propose the multi-dimensional (MD) method for genre variation. The MD approach has several salient characteristics [13]:

1. MD is a corpus-driven computational method, defined based on the analysis of a naturally occurring large number of texts.
2. MD helps in identifying linguistic features and patterns in individual texts and genres computationally.
3. MD is built on the hypothesis that different types of texts differ linguistically and functionally and that analysing only one or two of them is insufficient for reaching inferences.
4. MD is, as the name suggests, an explicitly multi-dimensional approach that assumes that in any discourse, it is anticipated that numerous parameters of variation will be active.
5. MD is quantitative in nature, i.e., early statistical techniques as reported by [14,15] have been reported to be useful in measuring frequency counts of linguistic features. Recent multivariate statistical techniques are useful in understanding the relationship between linguistic elements of the text.
6. MD combines macro- and micro-level analysis. That is, macroscopic evaluation of general linguistic patterns combined with microscopic measurement of specific features of the specific texts.

In earlier days, the knowledge extraction methods for register and stylistic analysis focused on the extraction of simple language-specific features such as pronouns, prepositions, auxiliary and modal verbs, conjunctions, determiners, etc. and a few language-independent features such as frequency of linguistic features. Significant progress in information extraction from text has lately been made feasible because of the creation of strong and reasonably accurate text analysis pipelines for a variety of languages [9]. This is also true in all the aforementioned instances where NLP-based technologies that automate the feature extraction process play a critical role. Various programmes exist now that utilize distinctive kinds of features to evaluate register, stylistic, or linguistic complexity.

Among these, the Stylo package [16] provides a comprehensive and user-friendly set of functions for stylometric studies. Stylo focuses on shallow text characteristics, such as n-grams at the token and character levels, that may be automatically extracted without the usage of language-dependent annotation tools. It should be noted, however, that it can also handle the output of linguistic annotation software. Text complexity may also be assessed using a variety of other tools. Coh-Metrix is a well-known example which uses characteristics retrieved from multi-level linguistic annotation to calculate over 200 indices of cohesion, language and readability from an input text [17]. Similarly, L2 Syntactical Complexity Analyzer (L2SCA) [18] and TAASSC [19] both estimate multiple linguistic variables that highlight grammatical complexity at the phrasal and sentence levels. These types of features are relevant in studies on first and second language acquisition. SweGram, a system specifically designed to profile texts in the Swedish language [20], is a striking exception to the preceding technologies, which are all designed for the English language. From this brief review, we can note that language-independent tools, such as Stylo, typically use shallow features that do not require language-specific preprocessing, whereas techniques based on a wide variety of multilevel linguistic features are frequently monolingual.

Profiling–UD [21] is a computational text analysis tool based on linguistic profiling concepts. It allows for the extraction of over 130 linguistic features from the given text. Because it is built on the Universal Dependencies framework, Profiling–UD is a multilingual tool that supports 59 languages. The features extracted from the tool can be grouped under raw text-related properties, lexical variety related features, morpho-syntactic features, verbal predicate structure-based measures, Global and Local Parsed Tree Structures, syntactic and subordination related measures. Table 1 highlights the information on feature categories extracted from the Profiling–UD tool. For more details about the tool, visit http://linguistic-profiling.italianlp.it/ (accessed on 2 March 2022).

Table 1. Features extracted from Profiling–UD.

Category of Feature	Definition of the Feature	Name as Seen in the Tool
Raw text features	This measures raw text features such as document length, sentence and word lengths, number and characters per token.	(n_sentences), (n_tokens), (tokens_per_sent), (char_per_tok)
Lexical variety	Measured in terms of its Type/Token Ratio (TTR) for both the first 100 and 200 tokens of a text in lemma and form. The TTR value ranges from one (high TTR) to zero (low lexical variety).	(ttr_lemma_chunks_100), (ttr_lemma_chunks_200), (ttr_form_chunks_100), (ttr_form_chunks_200)
Morpho–syntactic information	These measures highlight the percentage distribution of 17 core part-of-speech categories defined in the Universal POS tags, the lexical density of content words and inflectional morphology.	(upos_dist_*): distribution of the 17 core part-of-speech categories and (lexical_density), (verbs_tense_dist_*), (verbs_mood_dist_*), (verbs_form_dist_*), (verbs_gender_dist_*), (verbs_num_pers_dist_*)
Verbal predicate structure	This estimates the distribution of verbal heads and roots.	(verbal_head_per_sent), (verbal_root_perc), (avg_verb_edges), (verb_edges_dist_*)
Global and local parsed tree structures	These measure the average depth of the syntactic tree, average clause length, length of dependency links, the average depth of embedded complement chains governed by a nominal head, word order phenomena.	(avg_max_depth), (avg_token_per_clause), (avg_links_len), (avg_max_links_len), (max_links_len), (avg_prepositional_chain_len), (n_ prepositional_chains), (prep_dist_*), (obj_pre), (obj_post), (subj_pre), (subj_post)
Syntactic relations	This estimates the distribution of dependency relations of 37 universal syntactic relations used in UD.	37 (dep_dist_*)
Subordination phenomena/Use of Subordination	This evaluates the distribution of subordinate and main clauses, the relative order of subordinates concerning the verbal head and the average depth of embedded subordinate clauses.	(principal_proposition_dist), (subordinate_proposition_dist), (subordinate_post), (subordinate_pre), (avg_subordinate_chain_len), (subordinate_dist_*)

There have been increasingly large collections of data compiled across the internet. With advancements in technologies, these datasets are annotated and automatically analysed for multiple purposes [22]. However, linguistic profiling of texts is usually carried out for multiple different projects with a variety of end goals in mind. Language verification, author identification and verification, and text classification are a few to highlight here. Our focus is to identify specific linguistic features of a given text that influence the text classification into genres and specific subgenres. A brief review of the studies which have focused on linguistic profiling of fictional and non-fictional texts points to the study by [11], where they tried to estimate the readability of Italian fictional prose based on the linguistic profiling of the texts. Even though their study shows promising results, from a fictional prose point of view the dataset considered in the study is devoid of the fictional texts or does not cover most of the subgenres of the fictional type. Therefore, it is very important to conduct studies that consider multiple fictional subgenres that are popularly noted in the literature and compare their linguistic composition with the non-fictional text type. In the study by [11], the four major categories considered were literature further divided into children and adult literature, journalism (newspaper), educational writing (educational materials for primary school and high school) and scientific prose. When we look at the datasets which are utilized across literature for the task of classification or readability or author identification, we note that the Brown Corpus [23], the Lancaster-Oslo/Bergen (LOB) Corpus [24] and the British National Corpus (BNC) (The BNC is the result of a collaboration, supported by the Science and Engineering Research Council (SERC Grant No. GR/F99847), and the UK Department of Trade and Industry, between Oxford University Press (lead partner), Longman Group Ltd. (London, UK), Chambers Harrap (London, UK), Oxford University Computer Services, the British Library and Lancaster University) are the most prevalently used ones. Even though the nature of the BNC is the availability of a large mixed corpus which renders a possibility to analyse multiple genres of texts, it is not suitable for understanding comparing genres of fiction and non-fiction in detail. The Brown Corpus consists of over 500 samples coded under 15 genres as an early attempt at corpus linguistics. These 15 genres represented are not the universally accepted classification of genres. In fact, when the scope of the study is to measure readability or classification, the available datasets are acceptable. However, if we are interested in understanding the linguistic composition of various genres and subgenres of fictional and non-fictional texts, it is crucial that we define what we consider genres and subgenres of texts. Genre is a fluid concept which is always in constant flux due to the vast majority of researchers proposing different classification systems and different research goals. As the scope of our study is to highlight the linguistic similarities and differences in various subgenres of fiction and non-fictional texts, it is very important to consider a new dataset suitable for the goal of the experiment.

The goal of the present study was to investigate variation within and between genres by comparing a corpus of literary texts to corpora representing other textual genres using contrastive linguistic analysis.

3. Method

The study was carried out at the LELO laboratory located at the Institute of Specialized Studies (IKSI), Faculty of Applied Linguistics at the University of Warsaw. The study was carried out after obtaining ethical clearance from the local ethical committee at the University of Warsaw. The methodology section is divided into three sections, the first part deals with the corpora and the related preprocessing of the dataset. The second part deals with the linguistic profiling results of individual genres. The third section highlights the results of the classifier performance based on the linguistic profiled features for genre identification.

3.1. Corpora and Preprocessing

For the creation of corpus, we considered the text classification of [25] (fictional, non-fictional and poetry). We choose to ignore the category of poetry, as it is beyond the scope of our study. Further classification into subgenres was performed after considering the Reading Habits Questionnaire (RHQ) by [26]. Table 2 highlights the subgenre classification considered for the creation of corpus. The data for the corpus was gathered from various sources. The data for the fictional texts were gathered from the Gutenberg project. Project Gutenberg is a digital archive of over 65,000 books categorized under various subheadings and can be accessed in multiple formats such as HTML, PDF, TXT, EPUB, MOBI etc. All the materials downloaded from the Gutenberg project are covered under the Creative Commons license which makes them ready to use for this research study. Project Gutenberg is an online repository of texts such as short stories, novels, poems, biographies and many more. Despite being smaller than other public collections such as HathiTrust [27] and Google Books, Project Gutenberg has several advantages over those collections. It can be downloaded as a single package or can be scrolled for individual texts, which makes it versatile enough for multiple experiments. Also, most of the online repositories of digital documents use OCR technology to convert and preserve the documents. Texts under Project Gutenberg have been proofread by a human and in some cases even hand-typed, making them more suitable for experimental usage. The texts needed for the non-fiction were gathered from student writing samples of http://www.corestandards.org/ (accessed on 2 March 2022) [28] and various articles from the procedural texts we chose different projects/articles from the https://www.instructables.com/ (accessed on 2 March 2022) [29] website. Instructables is a dedicated web portal to obtain step-by-step instructions in building and carrying out a variety of projects.

Table 2. Summary of the dataset of the study.

Fiction (2153)	Non-Fiction (1514)
Children's Fiction (190)	Discussion Texts (395)
Fable (394)	Explanatory Texts (242)
Fantasy (249)	Instructional Texts (495)
Legends (44)	Persuasive Texts (382)
Mystery (191)	
Myths (48)	
Romance (591)	
Science Fiction (385)	
Thriller (61)	

Hence, we built a dataset which consists of both fictional and non-fictional texts with a special focus on carrying out a detailed linguistic analysis. Table 2 highlights the number of text samples (shown in brackets) considered in each subgenre grouped across fictional and non-fictional genres. The selected texts were divided into chapters, and it was made sure that the overall size of each of the texts would be around 100–2000 words. Preprocessing of the selected text was carried out to remove licensing information, unnecessary spaces and punctuation.

3.2. Linguistic Profiling of Texts

The scope of the present study is to carry out detailed linguistic profiling of various texts in the fictional and non-fictional categories. We chose to use the tool called Profiling–UD [21] for carrying out a detailed computational linguistic profiling of texts. As stated in the previous sections, this tool provides the most comprehensive set of features for a loaded text.

Each text was individually loaded onto the tool and corresponding features were extracted and tabulated. This process was repeated for all the texts. The results obtained from the analysis were loaded onto SPSS software [30] for further processing.

Even though the analysis of fictional and non-fictional texts was performed based on chapter-wise text, it can be noted that the overall number of sentences and number of tokens in the fictional texts are higher than in non-fictional texts. Table 3 shows the summary of the raw textual features across all subgenres and genres. However, the number of tokens per sentence and character per token is higher in non-fictional texts when compared to fictional texts. It was noted that there were individual differences across subgenres in terms of the number of sentences and tokens. Based on the raw text properties, it can be noted that mystery and thrillers, myths and legends, and fantasy and romance subgenres had similar raw text structures; whereas explanatory and persuasive texts had similar scores in the noted raw text properties. These findings support the hypothesis of [31] that non-fictional texts, notably informational and discussion texts, use substantially longer words and sentences than fictional texts, which use short and easy phrases.

Table 4 highlights the lexical variety noted in the subgenres, and it can be said that based on the values there were no statistical differences between them. However, it can be noted that the subgenres of fables had simple lexical variety and complexity and can be graded as even simpler than the non-fictional texts. This can be accredited to the population that the fables are targeted for—children need simple lexical variety. These findings add to the claims that fictional texts and subgenres report significantly higher TTR values suggesting greater lexical diversity and usage of unique words [32].

Table 3. Summary of the raw textual features across genres.

Parameter/Subgenre	n_sentences	n_tokens	tokens_per_sent	char_per_tok
Children's Fiction	559.40	5593.40	9.98	3.96
Fable	17.00	147.00	8.58	4.25
Fantasy	410.80	4399.80	10.65	4.10
Legends	680.80	7034.00	10.37	4.11
Mystery	981.80	9342.00	9.83	4.25
Myths	1378.20	13,339.80	9.62	4.57
Romance	644.40	6597.00	10.14	4.22
Science Fiction	551.80	5269.80	9.68	4.48
Thriller	1027.20	8921.20	8.78	4.27
Discussion	58.40	1191.00	19.69	4.73
Explanatory	176.80	1759.80	10.00	4.41
Instructional	147.60	1398.80	9.56	4.25
Persuasive	60.20	577.20	9.83	4.53
Fiction	694.60	6738.22	9.74	4.25
Non-fictional	110.75	1231.70	12.27	4.48

Table 4. Summary of the lexical variety features across genres.

Parameter/Subgenre	ttr_lemma_chunks_100	ttr_lemma_chunks_200	ttr_form_chunks_100	ttr_form_chunks_200
Children's Fiction	0.69	0.56	0.76	0.64
Fable	0.23	0.10	0.26	0.11
Fantasy	0.66	0.54	0.73	0.60
Legends	0.64	0.54	0.71	0.61
Mystery	0.72	0.61	0.79	0.68
Myths	0.68	0.60	0.73	0.65
Romance	0.68	0.61	0.74	0.66
Science Fiction	0.69	0.61	0.74	0.66
Thriller	0.69	0.62	0.75	0.67
Discussion	0.66	0.58	0.74	0.66
Explanatory	0.61	0.52	0.68	0.59
Instructional	0.67	0.59	0.76	0.66
Persuasive	0.62	0.48	0.71	0.56
Fiction	0.64	0.54	0.72	0.62
Non-fictional	0.63	0.53	0.69	0.59

Similarly, we looked at the parts of speech distribution in the various subgenres. Table 3 highlights the individual values of the distribution of parts of speech across various subgenres. When the values are compared across fiction and non-fictional texts, it can be noted that fictional texts have a lower number of adjectives but a higher number of adverbs, adpositions, pronouns and punctuation when compared to non-fictional texts. Whereas non-fictional texts have two times higher values of auxiliary verbs and nouns with slightly elevated values in numbers compared to fictional texts. No significant differences were noted in the values of coordinating and subordinating conjunctions, determiners, interjections, symbols and pronouns across fictional and non-fictional texts. Overall, the lexical density of fictional and non-fictional texts remained the same. Table 5 highlights the parts of speech distribution across all subgenres.

According to the Universal Dependencies (UD) framework, parts of speech can be divided into three types [33]. Figure 1 highlights this classification system, and it includes open class (ADJ, ADV, NOUN, VERB, PROPN, INTJ), closed class words (ADP, AUX, CONJ, DET, NUM, PART, PRON, SCONJ) and others (PUNCT, SYS, X). For more details, refer to Figure 1.

Table 5. Summary of parts of speech across and lexical density of various genres.

Parameter Subgenre	* ADJ	* ADP	* ADV	* AUX	* CCONJ	* DET	* INTJ	* NOUN	* NUM	* PART	* PRON	* PROPN	* PUNCT	* SCONJ	* SYM	* VERB	* X	Lexical Density
Children's Fiction	4.48	7.39	6.81	5.82	3.89	7.66	0.60	12.44	0.50	2.77	11.33	3.37	18.76	1.47	0.12	12.56	0.05	0.49
Fable	4.09	8.99	5.83	3.23	4.42	12.98	0.00	15.49	0.42	3.46	10.25	2.70	11.22	2.29	0.00	14.64	0.00	0.48
Fantasy	4.58	10.37	6.01	4.06	5.56	9.53	0.25	15.61	0.71	1.85	10.35	4.17	13.08	1.90	0.02	11.93	0.01	0.49
Legends	4.09	11.17	4.38	4.59	4.49	10.27	0.38	17.37	1.48	1.63	9.31	4.34	15.15	1.03	0.06	10.07	0.19	0.47
Mystery	5.46	9.87	6.48	5.78	3.19	9.44	0.35	14.49	0.66	2.02	11.55	1.77	15.66	2.28	0.17	10.77	0.07	0.46
Myths	6.22	11.41	3.76	4.15	3.77	10.61	0.16	18.19	1.17	1.42	5.98	7.12	15.12	1.35	0.23	8.89	0.45	0.52
Romance	4.79	9.09	5.48	5.95	3.51	7.44	0.44	13.56	0.49	2.64	11.68	4.20	16.80	1.93	0.05	11.91	0.03	0.48
Science Fiction	6.86	10.52	5.99	5.16	3.72	9.91	0.19	17.73	0.88	2.09	8.79	2.03	13.38	1.87	0.12	10.72	0.03	0.50
Thriller	5.80	10.33	5.65	5.31	2.68	9.28	0.26	15.45	0.41	2.06	12.02	2.43	14.89	1.76	0.11	11.57	0.01	0.48
Discussion	5.15	9.90	5.60	4.89	3.91	9.68	0.29	15.59	0.75	2.22	10.14	3.57	14.90	1.76	0.10	11.45	0.09	0.49
Explanatory	6.70	8.36	5.31	6.80	3.54	7.51	0.05	20.65	0.80	3.62	7.19	2.86	11.13	2.67	0.03	12.74	0.06	0.54
Instructional	6.83	8.66	5.29	6.66	3.42	10.54	0.26	21.33	1.13	2.53	6.68	1.45	11.68	1.77	0.07	11.66	0.05	0.53
Persuasive	5.26	8.14	3.68	2.73	2.70	10.75	0.18	22.46	4.30	2.01	5.06	7.47	12.19	1.30	0.43	10.88	0.44	0.57
Fiction	6.94	6.78	5.27	6.70	3.19	7.90	0.08	17.53	0.07	4.79	12.82	0.60	9.17	3.00	0.00	15.18	0.00	0.50
Non-fictional	6.43	7.99	4.89	5.72	3.21	9.18	0.14	20.49	1.58	3.24	7.94	3.10	11.04	2.19	0.13	12.62	0.14	0.54

* The parameters are upos_dist_.

	Tag	Description	Example
Open Class	ADJ	Adjective: noun modifiers describing properties	*nice, yellow, old*
	ADV	Adverb: verb modifiers of time, place, manner	*quick, yesterday*
	NOUN	Noun: words for persons, places, things, etc.	*India, apple, beauty*
	VERB	Verb: words for actions and processes	*doing, dancing*
	PROPN	Proper noun: name of a person, organization, place, etc.	*Akshay, TCS*
	INTJ	Interjection: greeting, polar questions	*yes, hello, oh*
Closed Class Words	ADP	Adposition (Preposition/postposition): describes a noun's spacial, temporal and/or other relation	*over, around, on*
	AUX	Auxillary: words which describe tense, mood etc.	*should, can, may*
	CCONJ	Coordinating conjunctions: words joining two phrases	*and, or*
	DET	Determiner: marks noun phrase properties	*a, an, the*
	NUM	Numerals: numbers	*one, two, three, fourth*
	PART	Particle: a preposition-like form used together with a verb	*up, down, in, at*
	PRON	Pronoun: a shorthand for referring to an entity or event	*who, others*
	SCONJ	Subordinating Conjunction: join a main clause with a subordinate clause such as a sentential complement	*which, that*
Other	PUNCT	Punctuation	*? , .*
	SYM	Symbols	*$, %*
	X	Other	*asdf, qwfg*

Figure 1. Universal Dependencies (UD) tagset by [34].

When we carefully examine the parts of speech distribution across non-fictional texts, we can note that instructional texts had significantly fewer adjectives, adverbs and auxiliary verbs when compared to other non-fictional texts. Also, the concentration of proper nouns in instructional texts is statistically higher than in any other text. Persuasive texts have a statistically significant fewer number of nouns, punctuation and adpositions, but higher values in pronouns and verbs overall. Based on the values of determiners, particle structure, subordinate conjunctions and interjections, we can group non-fictional texts into two groups: discussion–persuasive and explanatory–instructional. No significant differences were noted in the lexical density across all the subgenres. Therefore, it can be noted that open class and closed class words are equally important in the classification of texts into fictional and non-fictional genres.

Similarly, when we look at the subgenres of the fictional texts, we can note that myths and science fiction texts have the highest and lowest concentration of open-class words (specifically adjectives and adverbs, respectively) but this is not statically significant, whereas fables and children's fiction have the least concentration of open-class words (interjections) in the non-fictional text genre. No other significant differences in closed-class words were noted across other subgenres of fiction. Adverbs are the fewest in myths but others were statically insignificant. Auxiliary verbs and coordinating conjunctions were the fewest in children's fiction and thrillers, respectively, but were similar across all the other domains. Nouns are the fewest in children's fiction but are similar in all other domains. Children's fiction and romance had similar closed-class compositions. Myths and legends have the highest number of numerals and proper nouns, and the least occurrence of pronouns and verbs compared to all other subgenres. Lexical density, particles, punctuation, subordinating conjunctions, symbols and other domains are insignificant and are similar across all domains.

Table 5 highlights the part of speech distribution in the different text types. Pronouns and verbs appear to be frequently occurring in non-fictional texts. These findings are similar to the claim by [31] that these two elements are more common in conversation than in written language forms. The frequency of occurrence of nouns, on the other hand, is relatively low, resulting in a substantially lower noun/verb ratio. These findings are in line with the findings of [35], who suggest that novels have a narrative structure with a plot involved that requires the description of activities using verbs.

Further, when we look at the other morphosyntactic information such as inflectional morphology, the distribution of verbs according to their tense pre and post showed significant differences across fictional vs non-fictional texts. Fictional texts had higher past tense verbs whereas non-fictional texts are composed more of present tense verbs. Looking at the indicative and imperative verbs in fictional versus non-fictional texts, it was found that both kinds of texts are extensively composed of indicative verbs.

No statistical differences in the distribution of verbs according to their number and person, their tenses or even verbal mood were noted. Fables had the highest concentration of past tenses whereas persuasive texts had the lowest concentration of past tenses.

Syntactic features of verbal predicate structures, such as the verbal heads in the document, roots headed by a lemma tagged as a verb, verbal arity and distribution of verbs for arity class, were not found to be significantly different across the subgenres. Further, there were no significant differences noted in the parsed tree structures either, except that the prepositional chains for non-fictional texts had significantly lower values compared to fictional texts. However, fables had the smallest concentration of prepositional chains making their structure closer to non-fictional texts.

When studying the order of elements in syntactic structure, specifically the objects and subjects preceding and following the verbs, it was noted that fictional texts had slightly higher values, but this did not reach statistical significance. When examined individually, it was noted that the objects preceding verbs were least for fables and similar to the non-fictional category where there was not much difference across other subgenres.

Further contrastive analysis of 37 universal syntactic relations was carried out across fictional and non-fictional texts. It was observed that non-fictional texts had elevated values of the clausal modifier of the noun (adnominal clause), adjectival modifier, compound, phrasal verb particle, marker, numeric modifier, object, oblique nominal, which reached statistical significance, but non-significantly different values in adverbial clause modifier and punctuation.

In the use of subordination, none of the parameters reached statistical significance across fictional and non-fictional texts, but slight differences in the values of the distribution of principal clauses and subordinate clauses were noted. No further subgenre differences were noted.

One of the aims of this experiment was to highlight the main features that can be used for the classification of fictional and non-fictional categories for the task of genre classification.

4. Feature Reduction and Classification

We begin by providing a quick overview of the classification algorithm and feature selection approaches we employed in our trials (Section 4.1). Following that, we discuss the classification models that were trained on the dataset using the proposed feature sets (Section 4.2). The next section includes a feature selection experiment in which we evaluate the relevance of the features (Section 4.3). The next step is to re-run the classification methods using alternative subsets of the features to evaluate how this affects the model's accuracy.

4.1. The Classification Algorithm and the Feature Selection Methods

In this study, we utilised the Random Forest algorithm (RF), which is an ensemble learning method, as our classifier. The classification is based on the outcomes of several decision trees it generates during the training process [36,37]. We chose RF as it calculates the permutation relevance of the variables reliably during training the classification models. Table 6 highlights the feature details after dimensionality reduction. After that, we employed Rank Features by Importance (RFI) and Sequential Forward Search (SFS) to evaluate the features included in each model. Sections 4.3 and 4.4 explain RFI and SFS in detail.

Table 6. Feature details after dimensionality reduction.

Linguistic Category	Old Size	New Size for the Genre	New Size for Subgenre	Ignored Features Genre/Subgenre
Raw Text Properties	4	4	4	0/0
Lexical Variety	4	4	2	0/2
Morphosyntactic Information				
- upos_dist	18	11	13	7/5
- lexical_density	1	1	1	0/0
Inflectional Morphology	21	17	17	4/4
Syntactic Features				
- Verbal Predicate Structure	10	3	5	7/5
- Global and Local Parsed Tree Structures	10	7	8	3/2
- Order of Elements	4	3	3	1/1
Syntactic Relations				
- dep_dist	44	24	30	20/14
- Use of Subordination	8	7	5	0/3

4.2. Constructing RF Models

We utilised Jupyter Notebook for a quick implementation of RF. The features were evaluated in a sequential manner to predict the importance of each feature in the models' prediction success.

4.3. Using RFI to Assess the Relevance of the Features: Experiment One

To evaluate the variables, we used RF's built-in permutation importance [38] to rank their "importance". According to [39], the model is developed first, and its accuracy is computed in out-of-bag (OOB) observations to determine the relevance of the feature (Xi). Following that, any relationship between the values of Xi and the model's outcome is severed by permuting all the values of Xi, and the model's accuracy with the permuted values is re-computed. The permutation importance of Xi is defined as the difference between the accuracy of the new model and the original score. As a result, if a feature has noise or random values, the permutation is unlikely to affect the accuracy. A large difference between the two rates, on the other hand, indicates the importance of the feature for the prediction task. Figures 2 and 3 demonstrate the importance of several variables in genre and subgenre classifier models. The greater the relevance of the feature, the greater the value of the mean decrease in accuracy on the x-axis.

We also used the method of [40] to calculate the p-values for the variables under the null hypothesis that the permutation of the variable has no effect on the accuracy. Out of 131 features, 89 and 83 features from the genre and subgenre models, respectively, were found to have a significant effect on classifier models. The remaining features had a role in the models to varying degrees which did not reach significance.

4.4. Measuring Relevance of the Features Using SFS—Experiment Two

To implement SFS, we used the R package mlr [41]. The algorithm starts with an empty set of features and gradually adds variables until the performance of the model no longer improves. In this model, we used the classif.randomForest learner and the Holdout resampling method. If the improvement falls below the minimum needed value (alpha = 0.01), the algorithm comes to a halt. Each box in Figure 4 shows the selected features of each feature set.

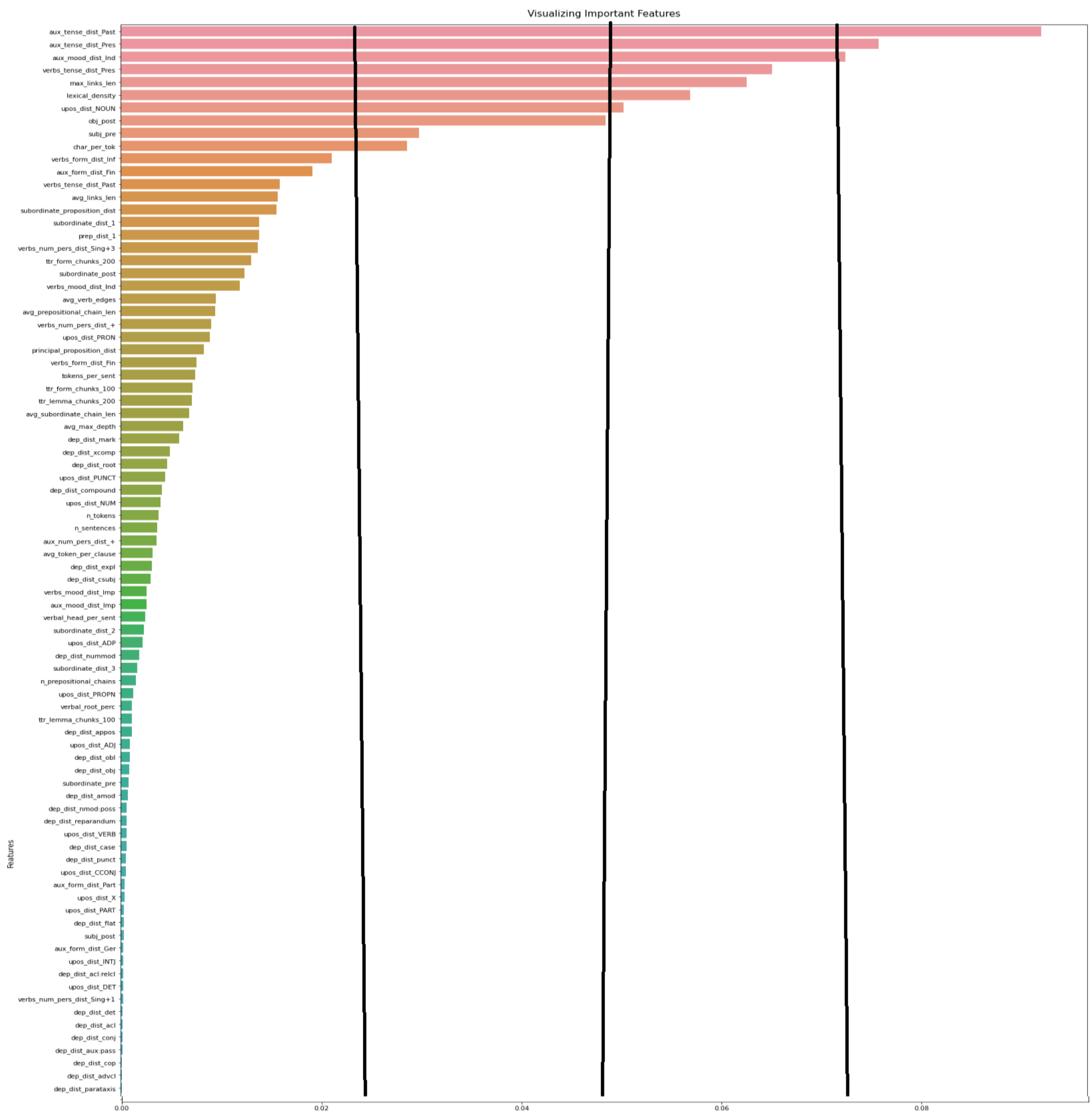


Figure 2. Variable importance plot of the RF genre model. NOTE: The x-axis shows the permutation relevance (mean decrease in accuracy) of each feature; the y-axis lists the features of the genre model.

4.5. Examining Various Feature Subsets Based on Their Significance—Experiment Three

Firstly, in Section 4.5.1, we explore the accuracy of different subsets of each feature set based on the results of the RFI and SFS experiments. In Section 4.5.2, we explore the subsets of all the features combined, trying to come up with an optimal consensus set of features.

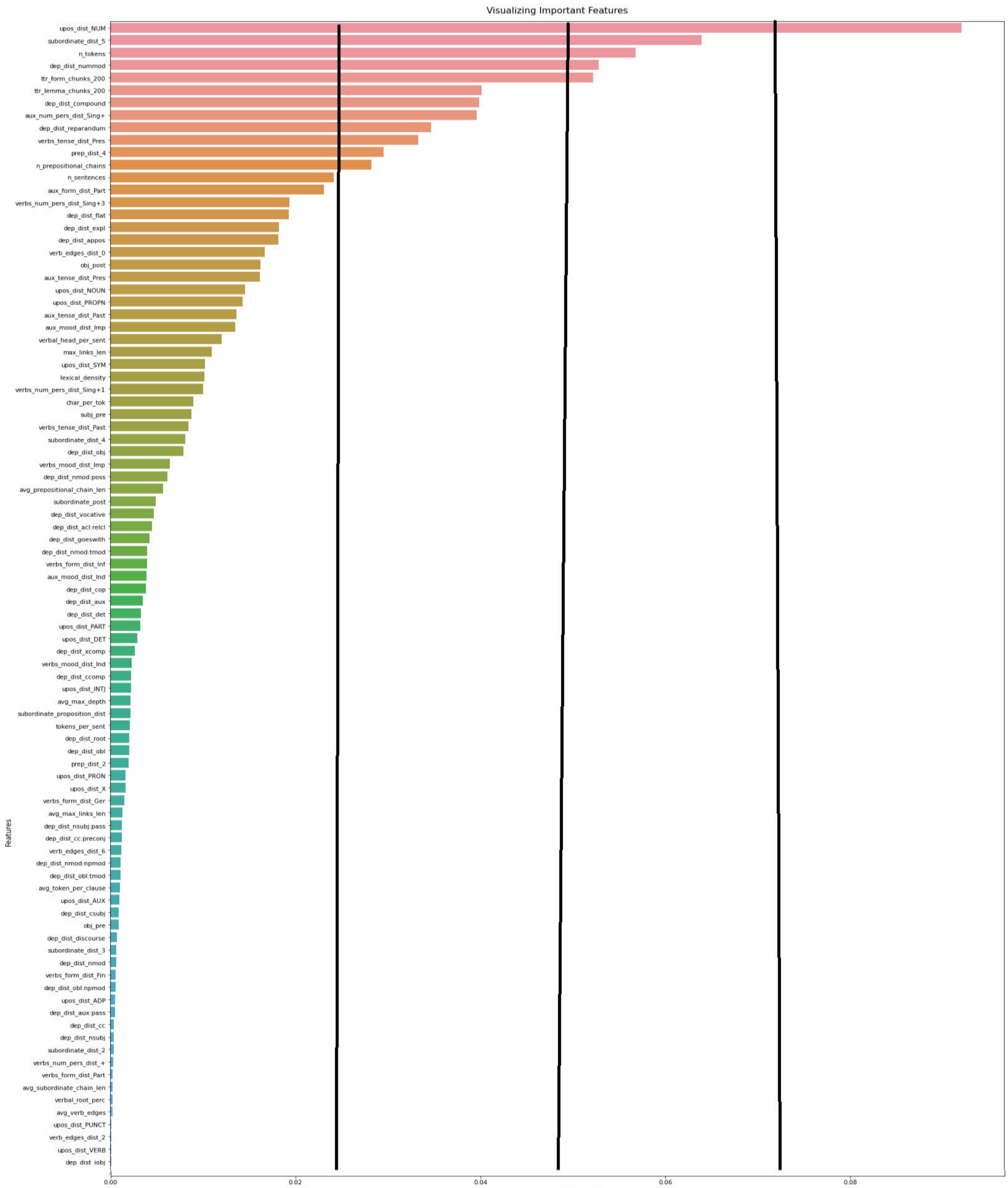


Figure 3. Variable importance plot of the RF sub-genre model. NOTE: The x-axis shows the permutation relevance (mean decrease in accuracy) of each feature; the y-axis lists the features of the subgenre model.

SUB-GENRE	GENRE
upos_dist_NUM	aux_tense_dist_Past
n_tokens	aux_tense_dist_Pres
ttr_form_chunks_200	aux_mood_dist_Ind
ttr_lemma_chunks_200	verbs_tense_dist_Pres
	lexical_density

Figure 4. SFS optimal features of each feature set.

4.5.1. Subsets of Each Feature Set

Tables 7 and 8 highlight the list of features that are considered with greater importance in genre determination, and the top 25th, 50th and 75th percentile of the features important for classification can be noted in Figure 4. The initial accuracy of each model is reported in the first row of Table 9 with the term original. Rows Top 25%, Top 50% and Top 75% report, respectively, on performing RF on the 25th, 50th and 75th percentile of important features for class determination. Similarly, the Top 5 row highlights the relevance of the first five features of each feature set with the greatest importance (according to Figure 4). The row Allimp highlights the results of applying the RF to all the features that were noted to play a part in classification.

Table 7. Subgenre selection of top features.

Top 25	Top 50	Top 75
upos_dist_NUM	upos_dist_NUM, subordinate_dist_5, n_tokens, dep_dist_nummod, ttr_form_chunks_200	upos_dist_NUM, subordinate_dist_5, n_tokens, dep_dist_nummod, ttr_form_chunks_200, ttr_lemma_chunks_200, dep_dist_compounds, aux_num_pers_dist_Sing+, dep_dist_reparandum, verbs_tense_dist_Pres, prep_dist_4, n_prepositional_chains

Table 8. Genre selection of top features.

Top 25	Top 50	Top 75
aux_tense_dist_Past, aux_tense_dist_Pres, aux_mood_dist_Ind	aux_tense_dist_Past, aux_tense_dist_Pres, aux_mood_dist_Ind, verbs_tense_dist_Pres, max_links_len, lexical_density, upos_dist_NOUN	aux_tense_dist_Past, aux_tense_dist_Pres, aux_mood_dist_Ind, verbs_tense_dist_Pres, max_links_len, lexical_density, upos_dist_NOUN, obj_post, subj_pre, char_per_token

Table 9. Accuracy of the model with feature selection.

Row Name	Genre Data	Subgenre Data
Original	0.87	0.82
Top 25%	0.71	0.64
Top 50%	0.77	0.71
Top 75%	0.84	0.79
Top 5	0.75	0.68
Allimp	0.93	0.89

4.5.2. Subsets of All Features

To investigate the possible combinations of all features based on the findings of the RFI and SFS tests, after trying out different subsets of each feature set.

1. In the RFI experiment, we initially applied RF to the set of attributes with the highest permutation relevance. The set, as shown in Figure 3, is {aux_tense_dist_Past, aux_tense_dist_Pres, aux_mood_dist_Ind, verbs_tense_dist_Pres}. The accuracy of this model is 0.889. From Figure 4, the set of features important are {upos_dist_NUM, subordinate_dist_5, n_tokens, dep_dist_nummod, ttr_form_chunks_200, ttr_lemma_chunks_200, dep_dist_compounds}. This model had an accuracy of 0.728.
2. The union of RF and the two most important features of each feature set: {*aux_tense_dist_*Past, aux_tense_dist_Pres, aux_mood_dist_Ind, verbs_tense_dist_Pres, max_links_len, lexical_density, upos_dist_NOUN, obj_post, subj_pre, char_per_token}. The accuracy of this model is 0.913. Similarly, for subgenre model, {*upos_dist_NUM, subordinate_dist_5, n_tokens, dep_dist_nummod, ttr_form_chunks_200, ttr_lemma_chunks_200, dep_dist_compounds, aux_num_pers_dist_Sing+, dep_dist_reparandum, verbs_tense_dist_Pres, prep_dist_4, n_prepositional_chains*} revealed the model accuracy of 0.792. The accuracy of this model is in line with the expected increase in the accuracy when compared with the accuracy of the union of the single most relevant features.

5. Conclusions

In this paper, we tried to linguistically profile the features noted in various fictional and non-fictional subgenres. By considering multiple feature sets highlighted in various computational SRF studies from a linguistic perspective, we attempted to connect the computational models and the linguistic explanations behind those features. As a result of the experiment, we are able to linguistically grade the composition of texts that constitute a text type. We also noted that for the task of genre classification the most important set of features are inflectional morphology, morphosyntactic information and raw text properties. However, for the task of subgenre classification, a mixture of semantic and syntactic features is important, i.e., morphosyntactic information, use of subordination, lexical variety, general syntactic features and parsed tree structures.

Based on the linguistic profiling of non-fictional texts we found that the linguistic composition of discussion and persuasion texts are similar across most of the domains of comparison, and explanatory and instructional texts show linguistic similarities as well. Similarly, grouping of subgenres of fiction can be performed for dyads of children's fiction and fantasy, myths and legends, and mystery and thrillers.

The results of the present study highlight the use of exact estimates of linguistic elements in each text type. These estimates could be useful in planning future use case experiments ranging from identifying developmental patterns in children [42,43] to estimating atypical language acquisition [44,45]. Further, we can also detect linguistic markers for acquired language disorders and cognitive impairments such as dementia and aphasia [46]. Similarly, we can estimate the writing abilities of school children [47]. Furthermore, from the perspective of computational sociolinguistics, the findings aid in the analysis of variations in the social component of language [8] as well as the modelling of stylometric features of authors [9]. By performing a comprehensive estimation of elements belonging to morphological, semantic and syntactic domains, we are able to grade the text types in terms of their complexities as well. This will be especially useful in such cases as readability measurement and selection of specific texts for language learning, among others.

Similarly, the current trend in linguistic analysis is to use complex network models for linguistic representation [48–50]. Complex networks have been used to model and study many linguistic phenomena, such as complexity [51–53], semantics [54], citations [55], stylometry [56–61] and genre classification [62–64]. Multiple studies [65,66] have concluded that the different properties of specific words on the macroscopic scale structure of a whole text are as relevant as their microscopic feature such as frequency of appearance. Linguistic research from the complex network approach is a relatively young domain of scientific endeavour. There is still a need for studies that can fill the gap in understanding the relationships between the system-level complexity of human language and microscopic linguistic features [48]. Although research in this area is on the rise and abundant findings have already been made, researchers need to have a clear knowledge of the microscopic linguistic features to determine the directions of further research. Our study highlights the crucial microscopic linguistic features which can be used to build better complex network models.

Even though the present study was comprehensive with the linguistic parameters considered, the dataset used was unevenly distributed across fictional and non-fictional text groups. Further studies which can address these issues and replicate the results of the present study in a controlled dataset would be required.

Funding: This research was conducted as part of the ELIT project, which has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie, grant agreement no. 860516.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: I wish to thank all the reviewers and Monika Płużyczka, Eliza, Niharika, Priyanka, Darshan and Deepak for helpful comments and discussion. I am also extremely grateful to all the members of IKSI at the University of Warsaw, who helped me in completing this research work.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Halteren, H.V. Linguistic Profiling for Authorship Recognition and Verification. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04), Barcelona, Spain, 21–26 July 2004.
2. Paltridge, B. Genre Analysis and the Identification of Textual Boundaries. *Appl. Linguist.* **1994**, *15*, 288–299. [CrossRef]
3. Cimino, A.; Wieling, M.; Dell'Orletta, F.; Montemagni, S.; Venturi, G. Identifying Predictive Features for Textual Genre Classification: The Key Role of Syntax. In Proceedings of the Fourth Italian Conference on Computational Linguistics CLiC-it, Rome, Italy, 11–13 December 2017.
4. Coulthard, M. Author Identification, Idiolect, and Linguistic Uniqueness. *Appl. Linguist.* **2004**, *25*, 431–447. [CrossRef]
5. Gamon, M. Linguistic correlates of style: Authorship classification with deep linguistic analysis features. In Proceedings of the COLING 2004: 20th International Conference on Computational Linguistics, Geneva, Switzerland, 23–27 August 2004; pp. 611–617.
6. Halteren, H.V. Author verification by linguistic profiling: An exploration of the parameter space. *ACM Trans. Speech Lang. Processing* **2007**, *4*, 1–17. [CrossRef]
7. Argamon, S.E. Computational Register Analysis and Synthesis. *Regist. Stud.* **2019**, *1*, 100–135. [CrossRef]
8. Nguyen, D.; Doğruöz, A.S.; Rosé, C.P.; De Jong, F.M. Computational Sociolinguistics: A Survey. *Comput. Linguist.* **2016**, *42*, 537–593. [CrossRef]
9. Daelemans, W. Explanation in computational stylometry. In *International Conference on Intelligent Text Processing and Computa-tional Linguistics*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 451–462.
10. Montemagni, S. Tecnologie Linguistico-Computazionali E Monitoraggio Della Lingua Italiana. *Studi Ital. Linguist. Te-Orica Appl. (SILTA)* **2013**, *42*, 145–172.
11. Dell'Orletta, F.; Montemagni, S.; Venturi, G. Linguistic profiling of texts across textual genres and readability levels. An exploratory study on Italian fictional prose. In Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP, Hissar, Bulgaria, 9–11 September 2013; pp. 189–197.
12. Biber, D. Spoken and Written Textual Dimensions in English: Resolving the Contradictory Findings. *Language* **1986**, *62*, 384. [CrossRef]
13. Biber, D. *Variation across Speech and Writing*; Cambridge University Press: Cambridge, UK, 1988. [CrossRef]
14. Carroll, J.B. Vectors of Prose Style. In *Style in Language*; Sebeok, T.A., Ed.; MIT Press: Cambridge, MA, USA, 1960; pp. 283–292.
15. Marckworth, M.L.; Baker, W.J. A discriminant function analysis of co-variation of a number of syntactic devices in five prose genres. *Am. J. Comput. Linguist.* **1974**, *11*, 2–24.
16. Eder, M.; Rybicki, J.; Kestemont, M.; Eder, M.M. Stylometry with R: A package for computational text analysis. *R Journal* **2016**, *8*, 107–121. Available online: https://journal.r-project.org/archive/2016/RJ-2016-007/index.html (accessed on 2 March 2022).

17. Graesser, A.C.; McNamara, D.S.; Cai, Z.; Conley, M.; Li, H.; Pennebaker, J. Coh-Metrix Measures Text Characteristics at Multiple Levels of Language and Discourse. *Elem. Sch. J.* **2014**, *115*, 210–229. [CrossRef]
18. Lu, X. Automatic analysis of syntactic complexity in second language writing. *Int. J. Corpus Linguist.* **2010**, *15*, 474–496. [CrossRef]
19. Kyle, K. Measuring Syntactic Development in L2 Writing: Fine Grained Indices of Syntactic Complexity and Usage-Based Indices of Syntactic Sophistication. Ph.D. Thesis, Georgia State University, Atlanta, GA, USA, 2016. [CrossRef]
20. Näsman, J.; Megyesi, B.; Palmér, A. SWEGRAM: A Web-Based Tool for Automatic Annotation and Analysis of Swedish Texts. In Proceedings of the 21st Nordic Conference on Computational Linguistics, Nodalida, Gothenburg, Sweden, 22–24 May 2017; pp. 132–141.
21. Brunato, D.; Cimino, A.; Dell'Orletta, F.; Venturi, G.; Montemagni, S. Profiling-UD: A tool for linguistic profiling of texts. In Proceedings of the 12th Language Resources and Evaluation Conference, Marseille, France, 11–16 May 2020; pp. 7145–7151.
22. Lu, X. *Computational Methods for Corpus Annotation and Analysis*; Springer: Berlin, Germany, 2014. [CrossRef]
23. Francis, W.N.; Kucera, H. *Manual of Information to Accompany a Standard Sample of Present-Day Edited American English, for Use with Digital Computers*; Technical Report; Department of Linguistics, Brown University: Providence, RI, USA, 1964.
24. Johansson, S.; Leech, G.N.; Goodluck, H. *Manual of Information to Accompany the Lancaster-Oslo/Bergen Corpus of British English, for Use with Digital Computers*; Department of English, University of Oslo: Oslo, Norway, 1978.
25. National Literacy Trust (Adapted from Crown Copyright). A Guide to Text Types: Narrative, Non-Fiction and Poetry. 2013. Available online: https://www.thomastallisschool.com/uploads/2/2/8/7/2287089/guide_to_text_types_final-1.pdf (accessed on 2 March 2022).
26. Kuijpers, M.M.; Douglas, S.; Kuiken, D. Capturing the Ways We Read. *Anglistik* **2020**, *31*, 53–69. [CrossRef]
27. Christenson, H.A. HathiTrust. *Libr. Resour. Tech. Serv.* **2011**, *55*, 93–102. [CrossRef]
28. Schutz, D. The Common Core State Standards Initiative. 2011. Available online: http://www.corestandards.org/ (accessed on 26 March 2022).
29. Wikipedia Contributors. Instructables. In Wikipedia, The Free Encyclopedia. Available online: https://en.wikipedia.org/w/index.php?title=Instructables&oldid=1024372150 (accessed on 26 March 2022).
30. IBM Corp. *Released. IBM SPSS Statistics for Windows*; Version 26.0; IBM Corp.: Armonk, NY, USA, 2019.
31. Biber, D.; Conrad, S. *Register, Genre, and Style*; Cambridge University Press: Cambridge, UK, 2009.
32. Jacobs, A.M. (Neuro-)Cognitive poetics and computational stylistics. *Sci. Study Lit.* **2018**, *8*, 165–208. [CrossRef]
33. Nivre, J.; De Marneffe, M.C.; Ginter, F.; Goldberg, Y.; Hajic, J.; Ryan Petrov, S.; Pyysalo, S.; Sil-veira, N.; Tsarfaty, R.; Zeman, D. Universal dependencies v1: A multilingual treebank collection. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia, 23–28 May 2016; pp. 1659–1666.
34. Nivre, J.; de Marneffe, M.C.; Ginter, F.; Hajič, J.; Manning, C.D.; Pyysalo, S.; Sebastian, S.; Tyers, F.; Zeman, D. Universal Dependencies v2: An evergrowing multilingual treebank collection. In Proceedings of the 12th Language Resources and Evaluation Conference, Marseille, France, 11–16 May 2020. 2020.
35. Voghera, M. La misura delle categorie sintattiche. Parole e numeri. In *Analisi Quantitative dei Fatti di Lingua*; Aracne: Roma, Italy, 2005; pp. 125–138.
36. Nayak, A.; Natarajan, D. Comparative study of naive Bayes, support vector machine and random forest classifiers in sentiment analysis of twitter feeds. *Int. J. Adv. Stud. Comput. Sci. Eng.* **2016**, *5*, 16.
37. Biau, G. Analysis of a random forests model. *J. Mach. Learn. Res.* **2012**, *13*, 1063–1095.
38. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
39. Strobl, C.; Boulesteix, A.-L.; Kneib, T.; Augustin, T.; Zeileis, A. Conditional variable importance for random forests. *BMC Bioinform.* **2008**, *9*, 307. [CrossRef]
40. Altmann, A.; Toloşi, L.; Sander, O.; Lengauer, T. Permutation importance: A corrected feature importance measure. *Bioinformatics* **2010**, *26*, 1340–1347. [CrossRef]
41. Bischl, B.; Lang, M.; Kotthoff, L.; Schiffner, J.; Richter, J.; Studerus, E.; Casalicchio, G.; Jones, Z.M. mlr: Machine Learning in R. *J. Mach. Learn. Res.* **2016**, *17*, 5938–5942.
42. Lu, X. Automatic measurement of syntactic complexity in child language acquisition. *Int. J. Corpus Linguist.* **2009**, *14*, 3–28. [CrossRef]
43. Lubetich, S.; Sagae, K. Data-driven measurement of child language development with simple syntactic templates. In Proceeings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, 23–29 August 2014; pp. 2151–2160.
44. Prud'hommeaux, E.; Roark, B.; Black, L.M.; Van Santen, J. Classification of atypical language in autism. In Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics, Portland, OR, USA, 23 June 2011; pp. 88–96.
45. Rouhizadeh, M.; Sproat, R.; Van Santen, J. Similarity measures for quantifying restrictive and repetitive behavior in conversations of autistic children. In Proceedings of the Conference Association for Computational Linguistics North American Chapter, Meeting, Seattle, DC, USA, 29 April–4 May 2015; p. 117.
46. Roark, B.; Mitchell, M.; Hollingshead, K. Syntactic complexity measures for detecting mild cognitive impairment. In *Biological, Translational, and Clinical Language Processing*; Association for Computational Linguistics: Cambridge, MA, USA, 2007; pp. 1–8. [CrossRef]

47. Barbagli, A.; Lucisano, P.; Dell'Orletta, F.; Montemagni, S.; Venturi, G. CItA: An L1 Italian learners corpus to study the development of writing competence. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia, 23–28 May 2016; pp. 88–95.
48. Cong, J.; Liu, H. Approaching human language with complex networks. *Phys. Life Rev.* **2014**, *11*, 598–618. [CrossRef]
49. Gao, Y.; Liang, W.; Shi, Y.; Huang, Q. Comparison of directed and weighted co-occurrence networks of six languages. *Phys. A Stat. Mech. Appl.* **2013**, *393*, 579–589. [CrossRef]
50. Lužar, B.; Levnajić, Z.; Povh, J.; Perc, M. Community structure and the evolution of interdisciplinarity in Slovenia's sci-entific collaboration network. *PLoS ONE* **2014**, *9*, e94429. [CrossRef] [PubMed]
51. Amancio, D.R.; Oliveira, O.N., Jr.; Costa, L.D.F. Structure–semantics interplay in complex networks and its effects on the predictability of similarity in texts. *Phys. Stat. Mech. Appl.* **2012**, *391*, 4406–4419. [CrossRef]
52. Segarra, S.; Eisen, M.; Ribeiro, A. Authorship attribution using function words adjacency networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–30 May 2013; pp. 5563–5567. [CrossRef]
53. Segarra, S.; Eisen, M.; Ribeiro, A. Authorship Attribution Through Function Word Adjacency Networks. *IEEE Trans. Signal Process.* **2015**, *63*, 5464–5478. [CrossRef]
54. Silva, T.C.; Amancio, D.R. Word sense disambiguation via high order of learning in complex networks. *Eur. Lett.* **2012**, *98*, 58001. [CrossRef]
55. Amancio, D.R.; Nunes, M.D.G.V.; Oliveira, O.N., Jr.; Costa, L.D.F. Using complex networks concepts to assess approaches for citations in scientific papers. *Scientometrics* **2012**, *91*, 827–842. [CrossRef]
56. Brede, M.; Newth, D. Patterns in syntactic dependency networks from authored and randomised texts. *Complex. InterNatl.* **2008**, *12*, 051915.
57. Liang, W.; Shi, Y.; Tse, C.K.; Liu, J.; Wang, Y.; Cui, X. Comparison of co-occurrence networks of the Chinese and English languages. *Phys. Stat. Mech. Appl.* **2009**, *388*, 4901–4909. [CrossRef]
58. Liang, W.; Shi, Y.; Tse, C.K.; Wang, Y. Study on co-occurrence character networks from Chinese essays in different periods. *Sci. China Inf. Sci.* **2012**, *55*, 2417–2427. [CrossRef]
59. Liu, H.; Li, W. Language clusters based on linguistic complex networks. *Chin. Sci. Bull.* **2010**, *55*, 3458–3465. [CrossRef]
60. Antiqueira, L.; Nunes, M.D.G.V.; Oliveira, O.N., Jr.; Costa, L.D.F. Strong correlations between text quality and complex networks features. *Phys. Stat. Mech. Appl.* **2007**, *373*, 811–820. [CrossRef]
61. Amancio, D.R.; Antiqueira, L.; Pardo, T.A.; Costa, L.D.F.; Oliveira, O.N., Jr.; Nunes, M.G. Complex net-works analysis of manual and machine translations. *Int. J. Mod. Phys. C* **2008**, *19*, 583–598. [CrossRef]
62. Amancio, D.R.; Oliveira, O.N.; Costa, L.D.F. Identification of literary movements using complex networks to represent texts. *New J. Phys.* **2012**, *14*, 043029. [CrossRef]
63. Costa, L.D.F.; Oliveira, O.N., Jr.; Travieso, G.; Rodrigues, F.A.; Villas Boas, P.R.; Antiqueira, L.; Viana, M.P.; Correa Rocha, L.E. Analyzing and modeling real-world phenomena with complex networks: A survey of applications. *Adv. Phys.* **2011**, *60*, 329–412. [CrossRef]
64. Newman, M.E.; Barabási, A.L.E.; Watts, D.J. *The Structure and Dynamics of Networks*; Princeton University Press: Princeton, NJ, USA, 2022.
65. Ke, J.; Yao, Y. Analysing Language Development from a Network Approach. *J. Quant. Linguist.* **2008**, *15*, 70–99. [CrossRef]
66. Akimushkin, C.; Amancio, D.R.; Oliveira, O.N., Jr. Text authorship identified using the dynamics of word co-occurrence networks. *PLoS ONE* **2017**, *12*, e0170527. [CrossRef]

information

MDPI

Review

A Literature Survey on Word Sense Disambiguation for the Hindi Language

Vinto Gujjar [1], Neeru Mago [2], Raj Kumari [3], Shrikant Patel [4], Nalini Chintalapudi [5,]* and Gopi Battineni [5,6]

1 Department of Computer Science & Applications, Panjab University, Chandigarh 160014, India; vntgujjar@gmail.com
2 Department of Computer Science & Applications, Panjab University Swami Sarvanand Giri Regional Centre, Hoshiarpur 160014, India
3 University Institute of Engineering and Technology, Panjab University, Chandigarh 160014, India
4 School of IT & ITES, Delhi Skill and Entrepreneurship University, Government of NCT of Delhi, Delhi 110003, India
5 Clinical Research Centre, School of Medicinal and Health Products Sciences, University of Camerino, 62032 Camerino, Italy
6 Department of Electronics and Communication Engineering, Velagapudi Ramakrishna Siddharth Engineering College, Vijayawada 520007, India
* Correspondence: nalini.chintalapudi@unicam.it

Abstract: Word sense disambiguation (WSD) is a process used to determine the most appropriate meaning of a word in a given contextual framework, particularly when the word is ambiguous. While WSD has been extensively studied for English, it remains a challenging problem for resource-scarce languages such as Hindi. Therefore, it is crucial to address ambiguity in Hindi to effectively and efficiently utilize it on the web for various applications such as machine translation, information retrieval, etc. The rich linguistic structure of Hindi, characterized by complex morphological variations and syntactic nuances, presents unique challenges in accurately determining the intended sense of a word within a given context. This review paper presents an overview of different approaches employed to resolve the ambiguity of Hindi words, including supervised, unsupervised, and knowledge-based methods. Additionally, the paper discusses applications, identifies open problems, presents conclusions, and suggests future research directions.

Keywords: word sense disambiguation; knowledge-based; supervised; unsupervised; Hindi language

Citation: Gujjar, V.; Mago, N.; Kumari, R.; Patel, S.; Chintalapudi, N.; Battineni, G. A Literature Survey on Word Sense Disambiguation for the Hindi Language. *Information* **2023**, *14*, 495. https://doi.org/10.3390/info14090495

Academic Editor: Peter Revesz

Received: 7 July 2023
Revised: 30 August 2023
Accepted: 2 September 2023
Published: 7 September 2023

1. Introduction

In the present age of information technology (IT), the whole world is sharing information using the internet. This information is available in natural language. As naturally understood, all-natural languages have an intrinsic feature called ambiguity. Ambiguity refers to the situation where a word can have multiple meanings. Ambiguity in natural language poses a significant obstacle in Natural Language Processing (NLP). While the human mind can rely on cognition and world knowledge to disambiguate word senses, machines lack the ability to employ cognition and world knowledge, leading to semantic errors and erroneous interpretations in their output. Therefore, the WSD process is employed to alleviate ambiguity in sentences.

WSD represents highly regarded formidable challenges within the realm of NLP and stands as one of the earliest quandaries in computational linguistics. Experimentation efforts in this domain commenced in the late 1940s, with Zipf's [1] introduction of the "law of meaning" in 1949. This principle posits a power law relationship between the frequency of a word and the number of meanings it possesses, indicating that more common words tend to have a greater range of meanings compared to less frequent ones. In 1975, Wilks [2] advanced the field by developing a model known as "preference semantics", which employed selectional constraints and frame-based lexical semantics to ascertain the precise meaning of a polysemous word. Notably, the 1980s witnessed significant progress in WSD research, facilitated by the availability of extensive lexical resources and corpora. Ultimately, WSD entails the task of identifying the accurate sense of a word within its specific contextual framework [3]. WSD is not considered a final objective; instead, it is recognized as an intermediary task with relevance to various applications within the field of NLP. Figure 1 presents the WSD conceptual diagram.

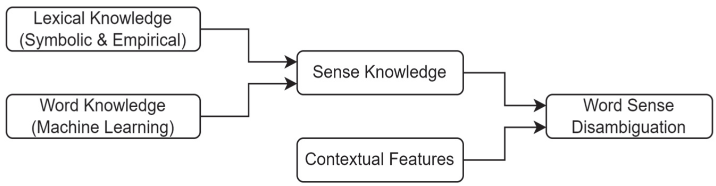


Figure 1. Conceptual Diagram of WSD.

In machine translation, WSD is an important step because a number of words in every language have a different translation according to the context of their usage [3–6]. It is an important issue to be considered during language translation. WSD assumes a crucial role in ensuring precise text analysis across a wide range of applications [7,8]. For example, an intelligence-gathering system could distinguish between references to illicit drugs and medicinal drugs through the application of WSD. Research works such as named entity recognition and bioinformatics research can also use WSD. In the realm of information retrieval (IR), the primary concern lies in determining the accurate sense of a polysemous word within a given query before initiating the search for its corresponding answer [9,10]. Enhancing the efficiency and effectiveness of an IR system entails the resolution of ambiguity within a query. Similarly, in sentiment analysis, the elimination of ambiguity is crucial for determining the correct sentiment tags (e.g., negative or positive) associated with a sentence [11,12]. In question-answering (QA) systems, WSD assumes a significant role in identifying the appropriate types of answers that correspond to a given question type [13,14]. Furthermore, WSD is necessary to accurately assign the appropriate part of speech tagging (POS) to a word, as its POS can vary depending on the contextual usage [15,16].

WSD can be categorized into two classifications: "all words WSD" and "target word WSD". In the case of all words WSD, the disambiguation process extends to all the words present in a given sentence, whereas target word WSD specifically focuses on disambiguating the target word within the sentence. WSD poses a significant challenge within the field of NLP and remains an ongoing area of research. It is regarded as an open problem, categorized as "AI-Complete", signifying that a viable solution does exist but has not yet been discovered. If we consider the given below two sentences in the Hindi language

आज-कल बाज़ार में नई-नई वस्तुओं की माँग बढ़ रही है ।

(aaj-kal baazaar mein naee-naee vastuon kee maang badh rahee hai)
(Now-a-days the demand of new things is increasing in the market.)

सुहागन औरतें अपनी माँग में सिंदूर भरती हैं ।

(suhaagan auraten apanee maang mein sindoor bharatee hain)
(Married women apply vermillion on their maang (the partition of hair on head).)

In both sentences, we have a common word, "माँग" (maang), that has a different meaning as per the context. In the initial sentence, the term refers to "the demand", whereas in the subsequent sentence, it denotes "the partition of hair on the head". Identifying the specific interpretation of a polysemous word is not a problem for a personage, whereas, for machines, it is a challenging task. Conversely, Hindi is the top fourth language, with over 615 million speakers worldwide. A significant amount of work is performed for English

WSD, but the WSD for the Hindi language is still in its infancy stage. Hindi WSD is now gaining the attention of researchers.

The objective of this paper is to provide a comprehensive survey of the existing approaches and techniques for WSD in the Hindi language. It presents several approaches employed for WSD in the context of Hindi. The paper highlights the specific challenges and limitations faced in WSD for Hindi due to its morphological complexity, rich lexical resources, and less availability of labeled data. The rest of this paper is structured in the following way: Section 2 discusses the various approaches for WSD, followed by a proposed methodology presented on WSD in Section 3. In Section 4, the survey results presented for WSD were critically reviewed, and Section 5 is the conclusion.

2. Various Approaches for WSD

Various approaches and methods used for WSD are classified into two categories, including knowledge-based approaches and ML (Machine Learning) based approaches. In knowledge-driven approaches, external lexical resources such as Wordnet, dictionary, and thesauri are required to perform WSD, and in ML-based techniques, classifiers are trained to carry out the WSD task on sense-annotated corpora. Figure 2 presents the different WSD approaches, and the explanation for each category can be explained further.

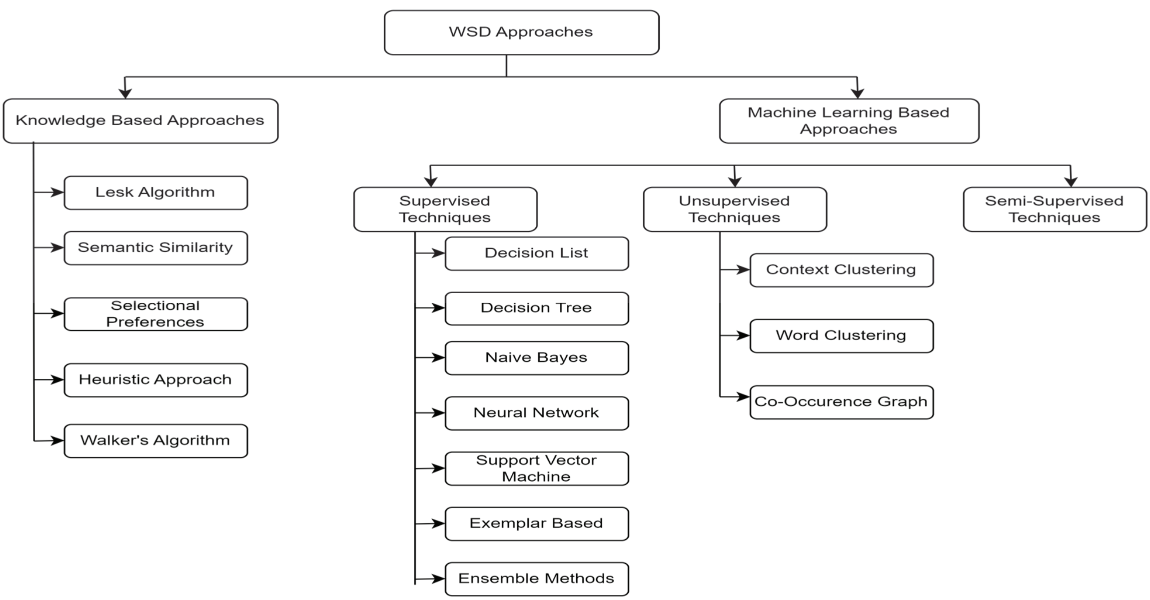


Figure 2. Classification of WSD Approaches.

2.1. Knowledge-Based Approaches

The knowledge-driven approach depends on various sources of knowledge such as dictionaries, thesaurus, ontologies, and collocations. The goal of these approaches in WSD is to utilize these knowledge resources to deduce the meanings of words within a given context. Let us delve deeper into an overview of several knowledge-based approaches.

- **LESK Algorithm**

The first algorithm developed using the knowledge-driven approach for WSD is the LESK algorithm [17,18]. The method relies on determining the degree of word overlap between the definitions or glosses of two or more target words. The dictionary definitions or glosses of the polysemous word are collected from the dictionary, and then these glosses and context words are compared. The desired sense of the polysemous word is determined by identifying the sense with the highest degree of overlap. A score is calculated for each pair of word senses using the provided formula, which is

$$\text{overlapscoreLesk}(S1, S2) = |\text{Gloss}(S1) \cap \text{Gloss}(S2)|$$

The senses of the respective words are assigned based on the maximum value obtained from the above formula, where *Gloss(Si)* represents the collection of words in the textual interpretation of sense *Si* of word *W*.

- **Semantic Similarity**

Words that exhibit a connection with one another possess a shared context, allowing for the selection of the most suitable sense of a word by leveraging the meanings found within the shortest semantic distance. Various metrics can be employed to compute the semantic similarity between two words [19].

- **Selectional preferences**

Selectional preferences provide insights into the categories of words that are likely to be associated with one another and convey shared knowledge [20,21]. For instance, "actors" and "movies" are words that exhibit semantic relationships. In this approach, inappropriate senses of words are excluded, and only those senses that align with common sense rules are taken into consideration. The methodology revolves around tallying the occurrences of word pairs with syntactic relations in a given corpus. The identification of word senses is accomplished based on this frequency count.

- **Heuristic Approach**

In the heuristic approach, to disambiguate word heuristics, they are calculated using the different linguistic properties. Three types of heuristics are employed as a baseline.

(a) The most frequent sense heuristic operates on the principle of identifying all possible meanings that a word can have, with the understanding that one particular sense occurs more frequently than others.
(b) The one sense per discourse heuristic posits that a term or word maintains the same meaning throughout all instances within a specified text.
(c) The one sense per collocation heuristic has a similar meaning to the one sense per discourse heuristic, but it assumes that nearby words offer a robust and consistent indication of the contextual sense of a word.

- **Walker's Algorithm**

Walker introduced an approach or technique for WSD in 1987 [22,23]. This approach incorporates the use of a thesaurus to accomplish the task. The initial step involves assigning a thesaurus class to each sense of a polysemous word. Subsequently, a total sum is computed by considering the context where the ambiguous word appears. If the context of the word matches the word sense with a thesaurus category, the total sum for that category increases by one.

2.2. ML-Based Approaches

In ML-based approaches, a classifier undergoes a training step to acquire knowledge of the attributes and subsequently determines the senses for the unseen examples. The resources that are used in this approach are based on a corpus that can be tagged or untagged. In these types of approaches, the target is the word to be disambiguated, also called the input word, and the surrounding text in which it is submerged is referred to as the contextual information. ML-based approaches are categorized into three types: supervised, unsupervised, and semi-supervised techniques.

2.2.1. Supervised Techniques

Supervised techniques for disambiguation utilize sense-annotated corpora for training purposes. These techniques operate under the supposition that the context itself can impart sufficient affirmation to resolve a sense of ambiguity. The context is represented as a collection of word "features", encompassing information about the neighboring words as well. Within these techniques, a classifier is trained using a designated training set that consists of instances specifically related to the target word. Overall, supervised approaches in WSD have generally achieved superior results compared to other approaches. However, the problem is that these techniques work on sensing annotated dataset, which is very expensive to create. Various supervised techniques are as follows:

- **Decision list**

In the context of WSD, a decision list refers to a sequential collection of "if-then-else" rules that are employed to determine the suitable sense for a given word [24,25]. It can also be viewed as a listing of weighed "if-then-else" rules. These rules are generated from a training set, utilizing parameters such as feature value, sensitivity, and score. The decision list is constructed by arranging these rules in descending order of their scores. When encountering a word, let us say w, its frequency of existence is computed, and its representation as a feature vector is used to evaluate the decision list, resulting in a calculated value. The attribute that has the highest value that matches the input vector corresponds to the meaning assigned to the word w.

- **Decision Tree**

A decision tree is a classification method that repeatedly divides the training dataset and organizes the classification rules in a tree-like structure [26,27]. Every interior node of the decision tree represents a test performed on an attribute value, and the branches represent the outcomes of the test. The word sense is determined when a leaf node is reached. An illustration of a decision tree for WSD is depicted in Figure 3. In this example, the sense of the polysemous word "bank" that is active is a noun within the sentence, "I will be at the bank of the Narmada River in the afternoon." The tree has been constructed and traversed to ultimately select the sense "bank/RIVER." A null value in a leaf node indicates that there is no sense selection present for that particular attribute value.

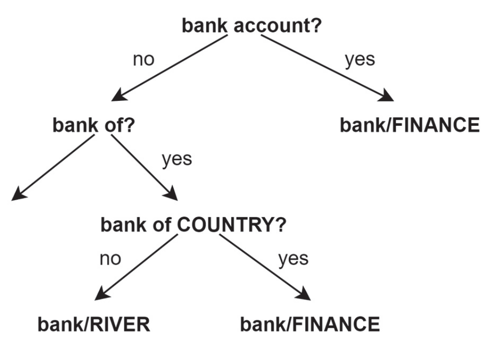


Figure 3. Decision Tree Example.

- **Naïve Bayes**

The NB (Naïve Bayes) classifier is a probabilistic classifier that applies Bayes' Theorem [28,29] to determine the appropriate meaning for a word. To classify text documents, it computes the conditional probability of each sense *Si* of word *w* based on the context features *j*. The sense *S* with the highest value, determined using the provided formula, is chosen as the most appropriate sense within the given context.

$$\hat{S} = \underset{S_i \in Sense\ _D(w)}{\text{argmax}} \ P(S_i \mid f_1, \ldots, f_m) = \underset{S_i \in Senses\ _D(w)}{\text{argmax}} \ \frac{P(f_1,\ldots, f_m|S_i)P(S_i)}{P(f_1,\ldots, f_m)}$$
$$= \underset{S_i \in Senses\ _D(w)}{\text{argmax}} \ P(S_i) \prod_{j=1}^{m} P(f_j|S_i)$$

In this context, m denotes the number of features. The probability score $P(Si)$ is computed based on the co-existence frequency of senses in the training dataset, while $P(fj \mid Si)$ is derived using the presence of the attribute given in the sense.

- **Neural network**

Neural networks consist of interconnected units or artificial neurons that serve as a loose model of human brain neurons [30,31]. They follow a connectionist approach and utilize a computational model for data processing. The learning program receives input attributes and target output. The objective is to divide the training data into non-overlapping sets based on desired responses. When new input pairs are presented to the network, the weights are adjusted to ensure the higher activation of the output unit that generates the desired result compared to other output units. In the context of neural networks, nodes represent words, and these words activate the associated concept with which they share semantic relations. Inputs propagate from the input to the output layer through intermediate layers. The network efficiently processes and manipulates the inputs to generate an output. However, generating a precise output becomes challenging when the connections within the network are widely dispersed and form loops in multiple directions.

- **Support Vector Machine (SVM)**

An SVM [32] serves the purpose of both classification and regression tasks. This approach is rooted in the concept of identifying a hyperplane that can effectively isolate positive examples from negative ones with the highest possible margin. The edge/margin represents the interspace between the hyperplane and the nearest examples for positive and negative, which are referred to as support vectors. In Figure 4, circle and square represent two different classes, the bold line represents the hyperplane that isolates the two classes while the dashed lines indicate the support vectors closest to positive and negative example. These support vectors play an important role in constructing an SVM classifier. The vectors have an impact on the position and the orientation of the hyperplane, and by removing or adding support vectors, adjustments can be made to the position of the hyperplane. In Figure 4,

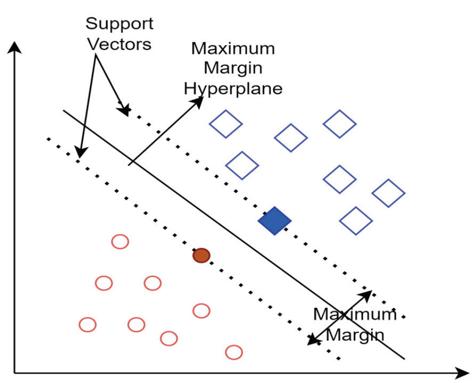


Figure 4. Illustrating SVM Classification.

- **Exemplar or instance-based learning**

In this approach, the classification model is constructed using examples [33]. In a feature space, these examples are represented as points, and the new examples are evaluated for classification. When new examples are encountered, they are progressively stored in the model. The k-nearest neighbor (k-NN) [34] method is an example of this type of approach. In k-NN, examples are stored based on their feature values, and the classification of the new examples is determined by considering the meanings of the k most similar previously stored examples. The hamming distance (a measure of the number of differing elements between two strings of equal length) [35,36] is calculated between new examples and the stored examples using the k-NN algorithm, which measures the proximity of the given input to the stored examples. The highest value obtained among the k-nearest neighbors represents the output sense.

- **Ensemble methods**

In order to enhance the accuracy of disambiguation, it is common to employ a combination of different classifiers. This combination strategy is called ensemble methods, which combine algorithms of different nature or with different characteristics [37]. Ensemble methods are more powerful than single-supervised techniques as they can overcome the weakness of a single approach. Strategies such as majority voting, the AdaBoost system of Freund and Schapire [38], rank-based combination, and probability mixture can be utilized to combine the different classifiers to improve accuracy. Figure 5 presents the simple approach of the ensemble WSD approach.

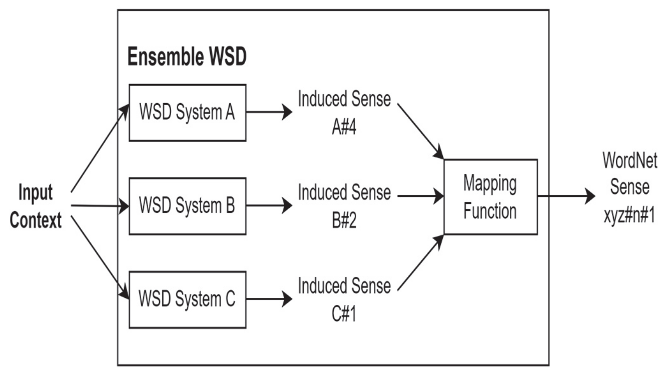


Figure 5. Ensemble Methods: Combining the Strengths of Multiple Models.

2.2.2. Unsupervised Techniques

Unsupervised techniques do not make use of sense annotated datasets or external knowledge sources. Instead, they operate under the assumption that senses with similar meanings occur in similar contexts. These techniques aim to determine senses from the text by clustering the word occurrences based on some measure of contextual similarity. This task is known as word sense induction or discrimination. Unsupervised techniques offer significant potential in overcoming the bottleneck of knowledge acquisition, as they do not require manual efforts. Here are some approaches that are used for unsupervised WSD.

Context Clustering: This unsupervised approach is rooted in the use of clustering techniques [39]. It begins by representing words through context vectors, which are then organized into clusters. Each cluster is corresponding to a sense of the target word. The approach revolves around the notion of a word space or vector space, where the dimensions represent individual words. Specifically, a word w is transformed into a vector, capturing the frequency of its co-occurrences with other words. This leads to the creation of a co-occurrence matrix, which is then subjected to various similarity measures. Finally, sense discrimination is performed by applying clustering techniques such as k-means clustering or agglomerative clustering.

Word Clustering: The induction of word senses can also be achieved through the use of word clustering [3]. This approach groups words that are semantically similar and may possess specific meanings. One commonly employed method for word clustering is Lin's method [40], which identifies words that are synonymous or have similarities to the target word. The similarity among the synonyms and the target word is determined by analyzing the features represented by syntactic dependencies found in a corpus, such as a verb–object, subject–verb, adjective–noun relationships, and so on. The more similar the two words are, the greater the extent to which they share information content. A word clustering algorithm is then utilized to differentiate between senses. Given a list of words W, the words are initially arranged based on their similarity, and a tree for similarity is constructed. In the beginning, the tree has only a single node, and through iterations, the most similar word is added as a child to the tree for each word in the list. Subsequently, pruning is performed, resulting in the generation of sub-trees. Each sub-tree, with the initial node serving as its root, represents a distinct sense of the original word.

Another method that is used for the clustering of words is the clustering by committee (CBC) [41] algorithm. The first step is similar to the above, i.e., a set of similar words is created for each input word. A similarity matrix is constructed to capture the pairwise similarity information between words. The second step involves the application of a recursive function to determine a set of clusters, referred to as committees. Following this, the average-link clustering technique is applied. In the final step, a discrimination process is executed, assigning the most alike cluster to each target word according to its similarity to the centroid of each committee. Subsequently, the intersecting attributes among the word and the committee are eliminated from the initial/actual word. This allows for the identification of less frequent senses for the same word in the next iteration.

Co-occurrence Graph: This approach utilizes a graph-based methodology. It involves the creation of a co-occurrence graph [42], denoted as G, comprising vertices V and edges E. Words are represented as vertices, and the connections between words that co-occur within the same paragraph are represented as edges. The weight assigned to each edge is determined by the frequency of co-occurrences, thus capturing the relationships between connected words. This graph construction effectively portrays the grammatical relations between the words.

In order to determine the sense of a word, an iterative method is used to identify the word with the highest degree node in the graph. Subsequently, a minimum spanning tree algorithm is applied to deduce the word's sense based on the information extracted from the graph. This process allows for a meaningful sense of disambiguation of the word within the given context.

2.2.3. Semi-Supervised Techniques

Semi-supervised techniques, known as weakly supervised or minimally supervised approaches, are utilized in WSD when training data are scarce. These methods make efficient use of both labeled and unlabeled data. Among the earliest algorithms in the realm of semi-supervised approaches is bootstrapping. Bootstrapping involves statistical resampling, where multiple datasets are generated from the original data with replacement. This technique is employed to estimate the accuracy and variability of a model or statistical inference, particularly in cases where traditional assumptions are not applicable or when working with small datasets.

The following table, Table 1 gives an in-depth comparison of various WSD approaches based on their benefits, drawbacks, and rationale for use. It seeks to provide a thorough grasp of how each method works and the settings in which they excel or may have limits.

Table 1. Comparative Analysis of Knowledge-Based, Supervised, Unsupervised, and Semi-Supervised Techniques.

Technique	Working	Advantages	Disadvantages	Justification for Usage
Knowledge-based	Utilizes pre-defined rules and human expertise to make decisions or classify data.	1. Interpretable outcomes	1. Limited scalability	Useful when domain-specific knowledge is available and interpretability is essential
		2. Robust to noisy data	2. Relies on expert knowledge	

Table 1. *Cont.*

Technique	Working	Advantages	Disadvantages	Justification for Usage
Supervised	Trained on labeled data with input–output pairs and predicts outputs for unseen data based on the learned model.	1. High accuracy	1. Requires labeled data	Preferred when labeled data are available and the goal is precise predictions
		2. Well-established algorithms	2. Sensitive to outliers and noise	
		3. Suitable for various problem types (classification, regression, etc.)	3. Lack of generalization to unseen classes or categories	
Unsupervised	Clusters data or discovers hidden patterns without labels.	1. Useful for exploratory data analysis	1. Limited guidance in model evaluation	Ideal for identifying structures in data when labeled data are scarce or unavailable.
		2. Can handle large datasets	2. Lack of direct feedback on model performance	
		3. Detects anomalies or outliers	3. Difficulty in interpreting the results	
Semi-supervised	Utilizes a combination of labeled and unlabeled data.	1. Utilizes the advantages of both supervised and unsupervised learning	1. Difficulty in obtaining and managing labeled data	Valuable when labeled data are expensive to acquire but unlabeled data are abundant
		2. Cost-effective for certain applications	2. Semi-supervised methods may not outperform fully supervised or unsupervised techniques	
		3. Improves performance with limited labeled data	3. May suffer from error propagation due to incorrect labels	

3. WSD Execution Process

WSD is the task of determining an ambiguous word's suitable sense based on context. WSD has seen a variety of methods. The majority of methods are based on different statistical methods. A few methods use corpora that have been sense-tagged, while others use unsupervised learning. The flowchart in Figure 6 shows the steps that are performed for WSD.

A string with an ambiguous word is given as an input string. Then, pre-processing is performed on this input string. Pre-processing steps such as stop word elimination, tokenization, part-of-speech tagging, and lemmatization, etc., are essential to transform raw text into a suitable format for analysis. For example, we have input 'राम कच्चा आम खा रहा है।' (raam kachcha aam kha raha hai) (Ram is eating raw mango). Various pre-processing steps are as follows:

Stop Word Elimination: Stop words are words commonly filtered out or excluded from the analysis process in NLP. These words are highly frequent in most texts, but they generally lack significant meaning or do not contribute much to the overall understanding of the content. By eliminating stop words, the text becomes less noisy, and contextual relevance is improved. This improved context helps the WSD algorithm make more accurate sense selections.

Examples of stop words in English include "the", "a", "an", "in", "on", "at", "and", "but", "or", "I", "you", "he", "she", "it", etc. Examples of stop words in Hindi

(Devanagari script) include "का," "की," "के," "को," "है," "हैं," "में," "और," "लेकिन," "या," "मैं," "तुम," "वह," "यह," आदि।.

The elimination of stop words and punctuation from the input text is performed in this step as they hold no significance or utility. After stop word removal string is 'राम कच्चा आम खा'.

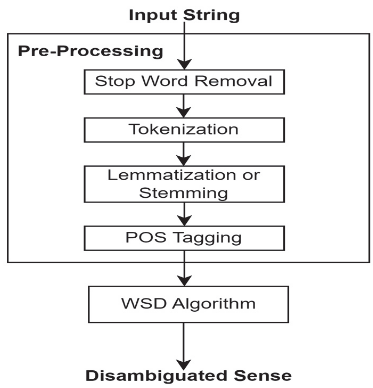


Figure 6. Flowchart of WSD Execution Process.

Tokenization: Tokenization is a fundamental technique in NLP that involves dividing a given text into smaller components, such as sentences and words. It encompasses the method of breaking down a string into a list of individual units called tokens. It helps in isolating individual words for disambiguation, making the WSD process more manageable and focused. In this context, a token can refer to a word within a sentence or a sentence within a paragraph, representing a fragment of the whole text. After Tokenization output is ('राम', 'कच्चा', 'आम', 'खा').

Stemming: Stemming is a linguistic process aimed at removing the last few characters of a word, which can sometimes result in incorrect meanings and altered spellings. Stemming simplifies text data and improves computational efficiency, aiding in tasks such as text matching and retrieval. However, it may generate non-words, leading to potential loss of word meaning and semantic distinctions. For instance, stemming the word 'Caring' would return to 'Car', which is an incorrect result.

Lemmatization: Lemmatization takes into account the context of a word and transforms it into its meaningful base form, known as a lemma. For example, by lemmatizing the word 'Caring,' the resulting lemma would be 'Care', which is the correct result. By converting words to their lemma, the WSD system can associate different inflected forms of a word with the same sense, improving the coverage and generalization of the sense inventory.

PoS Tagging: POS tagging involves the assignment of suitable part-of-speech labels to each word within a sentence, encompassing categories such as nouns, adverbs, verbs, pronouns, adjectives, conjunctions, and their respective sub-categories. This information is crucial for WSD because different parts of speech may have different senses. POS tagging helps in narrowing down the sense options for each word based on its grammatical role in the sentence.

When pre-processing is completed, the WSD algorithm is applied that gives the accurate sense of the ambiguous word as output. Various WSD algorithms are supervised, semi-supervised, unsupervised, and knowledge-based.

WordNet [43] is a valuable tool that plays a significant role in WSD. It serves as an extensive database containing nouns, adjectives, verbs, and adverbs, which are arranged into clusters of synonymous word groups known as synsets. These collections are interconnected through applied lexical and semantic relations. At IIT Bombay, Hindi WordNet

(HWN) [44] is being developed, which shares similarities with English WordNet. Words are grouped based on their perceived similarity in impact in HWN. It is worth noting that in certain contexts, terms that may have distinct meanings elsewhere can be considered synonymous. Each word within the HWN is associated with a corresponding synset that stands for "synonym set" and represents a group of words or terms that are synonymous or have similar meanings representing a lexical concept.

The WordNet synsets serve as its primary construction blocks. HWN controls words with open class categories or words with substance. Thus, the noun, adjective, verb, and adverb word categories that make up the HWN are included. The following characteristics apply to every entry in the HWN:

- Synset: This is a group of words, or synonyms, with similar meanings. For example, "पेन, कलम, लेखनी" (pen, kalam, lekhanee) refers to a tool or device used for writing with ink. According to the frequency of usage, the words are organized in the synset.
- Gloss: It explains the ideas. It is divided into two sections: a text definition that explains the concepts indicated by the synset (for example, "स्याही के सहयोग से कागज आदि पर लिखने का उपकरण (syaahee ke sahayog se kaagaj aadi par likhane ka upakaran)" elaborates on the idea of a writing or drawing instrument that utilizes ink), along with an illustrative sentence showcasing the importance of each word within a sentence. In general, a synset's words may be simply changed in a phrase (for instance, "यह पेन किसी ने मुझे उपहार में प्रदान की है। (yah pen kisee ne mujhe upahaar mein pradaan kee hai) (Someone gifted me this pen.)" illustrates the usefulness of the synset's words describing an ink writing or drawing equipment).

4. Results and Discussions

In this section, we presented the overview of which techniques and methodologies have been used by different researchers and what accuracy they have achieved, which datasets have been used by them, and what is specific about their techniques. We have divided it according to the techniques used by different researchers. It will help the researchers in the future to analyze which technique they should use.

4.1. Knowledge-Based Techniques

Knowledge-based techniques for WSD rely on external knowledge resources to resolve word ambiguities. These techniques use lexical databases, semantic networks, and linguistic resources to associate words with their appropriate meanings based on contextual information. As researchers delved into the subject, they started employing a combination of automatic knowledge extraction techniques alongside manual methods. Various knowledge-based techniques used by researchers for WSD are as follows:

In 1986, the first algorithm, called the Lesk algorithm [18], was developed by Michael Lesk for the disambiguation of words. In this algorithm, overlapping of the context where the word occurs and the definition of the input word from the Oxford Dictionary (OALD) was performed. The sense with the maximum overlap is chosen as the correct sense of the ambiguous word. In [17], Banerjee and Pederson introduced an adapted Lesk approach that relied on utilizing a lexical database, WordNet, as a source of knowledge rather than a machine-readable dictionary. WordNet, a hierarchical structure of semantic relations such as synonyms, hypernyms, meronyms, and antonyms, served as the foundation for this algorithm.

The notion of disambiguating Indian languages was initially proposed with a technique involving a comparison of contexts within which ambiguous words occurred with those created with HWN [45]. The sense would be determined according to its degree and extent of overlap. HWN arranges the lexical information based on word meanings. Hindi WordNet's design was influenced by English WordNet. HWN was developed by IIT Bombay, and it became publicly available in 2006. The accuracy range is about 40% to 70%.

Singh et al. [46] investigated the impact of the size of context window, stemming, and stop word removal on the Lesk-like algorithm for WSD in Hindi. The elimination of stop words coupled with the use of stemming is a proven method for obtaining good results, and they applied the Lesk algorithm to their work. From the analysis carried out, it is evident that utilizing 'Karak relations' leads to correct disambiguation. Additionally, stop-word elimination combined with stemming can help to raise the number of content-specific vocabulary while also promoting greater word stem overlap. A 9.24% improvement in precision is reported after the elimination of stop words and stemming over the baseline. In [47], the WSD technique relies on graph-based methods. They merged Lesk semantic similarity measures with Indegree approaches for graph centrality in their study. The beginning step involves constructing a graph for all target words in the sentence wherein nodes correspond to words and edges denote their respective semantic relations. By using Hindi wordNet along with the DFS Algorithm, we managed to create a final graph. The determination of a word's meaning is ultimately achieved through the application of graph centrality measures. An accuracy rate of 65.17% is achieved.

The authors introduced and evaluated the effectiveness of Leacock–Chodorow's measure of semantic relatedness for WSD of Hindi [48]. Having semantic similarity between two terms indicates a relationship. Semantic similarity and additional relations such as is-a-kind-of, is-the-opposite-of, is-a-specific-example-of, is-a-part-of, etc., are included in the relationships between ideas. The Leacock–Chodorow metric is employed, taking into account the length of routes among the noun concepts within an is-a hierarchy. The algorithm employs the Hindi WordNet hierarchy to acquire word meanings and uses it in the process of disambiguation rather than relying solely on the direct overlap. For evaluation purposes, a dataset consisting of 20 sense-tagged polysemous Hindi nouns is utilized. Using this metric, they found an accuracy of 60.65%. The role of hypernym, holonym, meronym, and hyponym interactions in Hindi WSD is examined [49]. We have taken into account five different scenarios in their research, including all relations, hyponym and hypernym, hypernym, holonym, and hyponym. The baseline makes no use of any semantic relations. When taking into account all relations, they found that total precision had increased by 12.09% over the baseline. The use of hyponyms produced the greatest improvement for a single semantic link and a precision improvement of 9.86% overall.

Sawhney et al. [50] employed a modified Lesk approach that incorporates a dynamic context window concept. The dynamic context window refers to the number of preceding and succeeding words surrounding the ambiguous words. According to this approach, if two words have similar meanings, then there must be a common topic in their vicinity. An increase in precision signifies that this algorithm provides superior results as compared to prior methods that employ a fixed-size context window. The lesk approach was applied to bigram and trigram words to disambiguate the verb words [51], and it is the only work, as per our knowledge, that disambiguates Hindi verbs, as most of the work is performed for nouns.

In [52], Goonjan et al. make use of Hindi Wordnet to retrieve the senses of the words, and then a graph is created using a depth-first search between the senses of the words. After that, weights are assigned to the edges of the connecting node according to the weights of the Fuzzy Hindi wordnet. Then, various local fuzzy centrality measures are applied, and the values of these calculated measures help us to find the accurate meaning of the polysemous word. The knowledge-driven Lesk algorithm is employed in [53] that works by selecting the meaning whose definition most closely matches the In their investigation, they successfully identified 2143 out of 3000 ambiguous statements, achieving an accuracy rate of 71.43%.

In [54], WSD for the Bengali language is performed in two distinct phases. During the first phase, sense clusters of an ambiguous word are constructed by considering the preceding and succeeding words in their context. In the second phase, WSD is performed by utilizing a semantic similarity measure after expanding the context with the assistance of Bengali WordNet. An ambiguous Bengali words test set, comprising 10 words, is used, for testing which has 200 sentences for each ambiguous word. The overall accuracy achieved is 63.71%. Tripathi et al. [55] have used a Lesk algorithm along with a novel scoring method.

To enhance the performance of the Lesk Algorithm, they employed a scoring technique that evaluates token senses based on their cohesive variations. This strategy aimed to improve the accuracy and effectiveness of the approach. Based on a combination of different senses of tokens according to the gloss along with their related hypernyms and hyponyms, a sense rating is assigned that helps in determining the meaning of the ambiguous word.

A complete framework named “hindiwsd” [56] is constructed for WSD of Hindi in Python language. It is a pipeline that performs a series of tasks, including transliteration of Hinglish code-mixed text, spell correction, POS tagging, and the disambiguation of Hindi text. A knowledge-based modified Lesk algorithm is applied here for WSD. A comparative analysis of various knowledge-based approaches is also performed in [57]. The results demonstrate that accuracy is lower for limited resource languages and higher for languages with abundant knowledge resources. A knowledge-based resource is critical in the processing of any language. The survey suggests that several factors influence the performance of WSD tasks. These include the removal of stop words, the positioning of ambiguous words, the use of Part-of-Speech (POS) tagging, and the size of the dataset utilized for training. Each of these elements plays a significant role in the overall effectiveness of WSD methods.

This is a review of some knowledge-based approaches that have been used by different researchers for WSD. Knowledge-based techniques can be effective in resolving word sense ambiguities, especially when supported by comprehensive and well-structured lexical resources and linguistic knowledge. However, they may have limitations when dealing with unseen or domain-specific contexts, as they heavily rely on the information present in the knowledge bases. In such cases, supervised and unsupervised machine learning approaches are often employed to complement the knowledge-based methods and improve overall disambiguation performance.

4.2. Supervised Techniques

Supervised techniques for WSD are highly effective in resolving word sense ambiguities by utilizing labeled training data, achieving high accuracy through diverse and well-annotated datasets that associate words with correct senses in various contexts. These methods capture deeper semantic relationships, enabling a nuanced understanding of word sense distinctions while exhibiting context sensitivity to handle complex sentence structures and resolve ambiguous words. We present a review of various supervised techniques used for WSD of Indian languages.

NB classifier [58], a supervised method equipped with eleven different features such as collocations, vibhaktis vibhaktis (the grammatical cases or inflections used in Indian languages to indicate the function of nouns or pronouns in a sentence), unordered list of words, local context, and nouns has been applied to solve Hindi WSD. In order to assess its performance, the NB classifier was applied to a set of 60 polysemous Hindi nouns. Applying morphology to nouns included in a feature vector led to achieving maximum precision of 86.11%, while considering the nearby nouns in the context of a target ambiguous noun is important for achieving accurate meaning.

In [59], a supervised approach using cosine similarity is introduced. Vectors have been generated for the query given for testing and knowledge data for the sense of the polysemous word, taking weights into account. The sense with the maximum similarity to the polysemous word is selected as the appropriate sense. The experiment is conducted on a dataset comprising 90 Hindi-ambiguous words. An average precision of 78.99% is obtained.

The supervised approach of the k-NN algorithm has been used for Gurumukhi WSD [60]. Two feature sets are derived: one comprises frequently occurring words alongside the ambiguous word, and the other encompasses words neighboring the ambiguous word in the corpora. Subsequently, the provided data are divided into the training and the testing sets. The k-NN classifier is trained using the training set. For the given input sentence, pre-processing is performed, and then its vector is generated. The k-NN classifier identifies similar vectors or nearest neighbors for the unknown vector. After that, the distance between the input vector/unknown vector and nearest neighbors is calculated using the Euclidean method. The closeness between the vectors is determined by using this distance.

The WSD of Panjabi has been accomplished using a supervised NB [61] classifier. For feature extraction, both Bag-of-Words (BoW) and a collocation model are employed. The collocation model utilizes only the two preceding and two succeeding words of the input word as features, whereas the BoW model considers all the words surrounding the input word as features. Using both feature extraction methods, the NB classifier is trained on a dataset of 150 ambiguous words with six or more senses collected from the Punjabi word net. The system attains an accuracy of 89% for the Bow model, and for the collocation model, the accuracy is 81%.

In [62], a comparative analysis is conducted among rule-based, classical machine learning, and two neural network-based RNN and LSTM models. The evaluation is carried out on four highly ambiguous terms and a group of seven other ambiguous words. The rule-based method achieved an accuracy of 31.2%, classical machine learning attained 33.67% accuracy, while RNN exhibited an accuracy of 41.61%. Notably, the LSTM model outperformed all other methods with an accuracy of 43.21%, showcasing its superior performance in disambiguating word senses.

A review of some supervised techniques is presented. Supervised techniques excel in providing fine-grained disambiguation, which is essential for precise semantic interpretation. However, their dependency on labeled data poses challenges, especially for resource-limited languages. Supervised techniques may struggle with unseen words or senses, and overfitting remains a concern, potentially affecting performance on new data. To address limitations, researchers often combine supervised methods with unsupervised or knowledge-based approaches to enhance overall WSD performance.

4.3. Unsupervised Techniques

Unsupervised techniques for WSD present advantages in their independence from labeled training data, making them more cost-effective and adaptable to different languages and domains. By learning solely from distributional patterns, they have the potential to discover new word senses and uncover novel semantic relationships. A review of unsupervised techniques used for WSD of Indian languages is as follows:

An unsupervised approach is used for resolving word ambiguity in [63]. As part of the pre-processing steps, the elimination of stop words and stemming is required when encountering an unclear context. After employing the decision list for untagged examples, there is a need for some manual intervention to provide seed examples. A decision list is employed to generate ambiguous words, and this decision list is subsequently utilized to determine the sense of such ambiguous words.

A technique to perform unsupervised WSD on a Hindi sentence using network agglomeration is proposed in [64]. We start by creating a graph G for the input sentence. All variations in meaning for this sentence can be seen collectively in this graph. Sentence graphs can be used to develop interpretation graphs such as G, and the sentence must have an interpretation for all instances of G. To find out which is the preferred interpretation, we perform network agglomeration on all relevant graphs. By identifying which interpretation holds the highest network agglomeration value, we can derive its relevance.

In [65], the author deals with algorithms based on an unsupervised graph-based approach. This consists of two phases: (1) A lexical knowledge base is utilized to construct a graph, where each node and edge in the graph represents a possible meaning of a word within a given sentence. These nodes and edges capture dependencies between meanings, such as synonyms and antonyms. (2) Subsequently, the graph is analyzed to identify the most suitable node, representing the most significant meaning, for each word according to the given context. In the graph-based WSD method of unsupervised techniques, word meanings are determined by considering the dependencies between these meanings.

Relations in HWN are crisp, meaning they are either related or not related at all. There is no partial relation between words in the Hindi wordnet. However, in real life, partial relations can also exist between words, which are also called fuzzification of relations. Therefore, an expanded version of Hindi wordnet that incorporates fuzzy relations is called Fuzzy Hindi WordNet (FHWN), which is represented as a fuzzy graph in which nodes depict words/synsets and the edges show fuzzy relationships within words/synsets. The fuzzy relations are assigned a membership value between [0, 1]. The values are assigned by consulting with experts from diverse domains. In [66], an approach using fuzzy graph connectivity measures is applied to FHWN for WSD. Various local and global connectivity measures are calculated using the values assigned to the relations. The sense with the maximum rank is chosen as the suitable sense for the ambiguous word. The utilization of the FHWN sense inventory results in an improvement in disambiguation performance, with an average increase of approximately 8% in most cases. Since the membership value can change, so can the algorithm's performance.

In [67], a multilingual knowledge base called ConceptNet is used to automatically generate the membership values. The nodes and edges that make up ConceptNet's network represent words, word senses, and brief phrases, while the edges show how the nodes are related to one another. The Shapley value, which is derived from co-operative game theory, is then employed as a centrality metric. Shapley's value is utilized to mitigate the influence of alterations in membership values within fuzzy relationships by considering only the marginal contributions of all the values in the calculation of centrality.

For Gujarati WSD [68], a genetic algorithm-based strategy was employed. Darwin's idea of evolution serves as the basis for genetic algorithms. The population is the first set of solutions the algorithm considers (represented by chromosomes). One population's solutions are utilized to create a new one. This approach is pursued with the expectation that the new population will exhibit improved performance compared to the previous population. The solutions chosen to create new descendants (solutions) are selected based on their suitability. This process is carried out again and again until or unless a certain need (such as the number of people or an improvement in the ideal solution) is attained.

Kumari and Lobiyal [69] introduced a word-embedding-based approach for WSD. They employed two word2vec architectures, namely the skip-gram and the continuous bag-of-words models, to generate word embeddings. The determination of the appropriate sense of a word was achieved using cosine similarity. An unsupervised Latent Dirichlet Allocation (LDA) and Semantic features-based approach using semantic features has been applied for the target WSD of the Malayalam language [70]. A dataset consisting of 1147 contexts containing target polysemous words has been utilized. In total, 80% accuracy is achieved.

Various word embedding methods such as Bow, Word2Vec, TF-IDF, and FastText have been used in [71]. For the construction of Hindi word embeddings, Wikipedia articles were used as the data source. They conducted multiple trials to explore this idea, and the results convinced us that Word2Vec outperforms all other embeddings for the Hindi dataset we examined. When training the input, the method uses word embedding techniques. It also incorporates clustering, which is used to create a sense inventory that aids in disambiguation. These methods can use unlabeled data because they are unsupervised. The accuracy achieved is 71%.

In [72], The authors employed an approach based on a genetic algorithm (GA) for Hindi WSD. The process involved pre-processing and creation of a context bag and sense bag, followed by the application of the GA. The GA encompassed selection, crossover, and mutation to disambiguate the word, and the approach was tested on a manually created dataset. The experimental results demonstrated an accuracy of 80%. A comparative analysis of two path-based similarity measures is performed in [73]. The experimental investigation is performed using the shortest path and Leacock–Chodorow methods, which shows that a Leacock–Chodorow similarity measure performs better than the shortest path measure. Experimentation is performed on five polysemous nouns, and an average accuracy of 72.09% is achieved with the Leacock–Chodorow method.

Unsupervised techniques are cost-effective, and they use unlabeled data. Thus, they can be used for languages that lack sense-tagged datasets. However, they may struggle with sense overlapping and lack deep semantic interpretation, leading to less precise disambiguation compared to supervised methods. Data sparsity can also limit their effectiveness, requiring substantial data for satisfactory performance. Evaluating their performance can be challenging without a definitive gold standard for comparison. Combining unsupervised techniques with supervised or knowledge-based approaches can address their limitations and enhance overall WSD performance.

The following table, Table 2, exhibits the summary of study characteristics of different Indian language WSD approaches.

Table 2. Analysis of WSD Approaches in Different Indian Languages.

Year (Ref.)	Language	Technique	Method	Specification	Dataset Used	Accuracy	Comments
1986 [18]	English	Knowledge-Based	Lesk	Overlapping of context and word definition is performed.	Used Machine Readable Dictionaries	-	Only definitions are used for deriving the meaning.
2002 [17]	English	Knowledge-Based	Adapted Lesk	The proposed approach expands the comparisons by incorporating the glosses of words that are linked to the words under disambiguation in the given text. These connections are established using the WordNet lexical database.	WordNet is used	32%	-
2004 [45]	Hindi	Knowledge-Based	Lesk Method	Comparison of the ambiguous word’s context and the context derived from Hindi WordNet is performed.	The manually created test set.	40–70%	Works with only nouns and does not deal with morphology.
2009 [63]	Hindi	Unsupervised	Decision List	After pre-processing, a decision list of untagged examples is created that is utilized to depict the meaning of the polysemous word.	A dataset for 20 ambiguous words with 1856 training instances and 1641 test instances was used.	The accuracy ranges from approximately 82% to around 92% when employing techniques such as stop-word elimination, automatic generation of decision lists, and stemming.	-

Table 2. *Cont.*

Year (Ref.)	Language	Technique	Method	Specification	Dataset Used	Accuracy	Comments
2012 [46]	Hindi	Knowledge-Based	Lesk Algorithm	Effects of context window size, stop word elimination, and stemming has been analyzed with Lesk	Evaluation is carried out on a test set of 10 polysemous with 1248 test instances.	Improvement of 9.24% over baseline.	Works only for nouns.
2012 [47]	Hindi	Knowledge-based	Graph-Based	A graph is constructed using the DFS algorithm and then centrality measures are applied to deduce the sense of the word.	Text files that contain 913 nouns are used as datasets.	65.17%	For graph centrality, only the in-degree algorithm is used.
2013 [48]	Hindi	Knowledge-Based	A Leacock-Chodorow measure of semantic relatedness	The Leacock–Chodorow algorithm is used to find the length of the route among two noun concepts.	dataset of 20 polysemous Hindi nouns	60.65%	Works only for nouns
2014 [49]	Hindi	Knowledge-Based	Semantic Relations	The significance of different relationships such as hypernym, hyponym, holonym, and meronym is examined here.	dataset of 60 nouns is used.	Improvement of 9.86% over baseline.	Only for nouns.
2014 [58]	Hindi	Supervised	Naive Bayes	Naive Bayes classifier with eleven different features has been applied for Hindi WSD.	A dataset of 60 polysemous Hindi nouns is used.	86.11%	Works only for nouns
2014 [50]	Hindi	Knowledge-Based	Modified Lesk	A modified Lesk approach with a dynamic context window is used in this paper.	A dataset of 10 ambiguous words is used.	-	Accuracy depends on the size of the dynamic context window.
2015 [64]	Hindi	Unsupervised	Network Agglomeration	An interpretation graph is created for each interpretation derived from the graph of the sentence, and subsequently, network agglomeration is performed to determine the correct interpretation.	Health and Tourism datasets are used.	Health-43% (All words) and 50% (Nouns) Tourism-44% (All Words) and 53% (Nouns)	Works for nouns as well as other parts of speech, too.

Table 2. *Cont.*

Year (Ref.)	Language	Technique	Method	Specification	Dataset Used	Accuracy	Comments
2015 [65]	Hindi	Unsupervised	Graph Connectivity	A graph is generated to represent all the senses of a polysemous word, then it is analyzed to determine the accurate sense of the word.	Hindi Wordnet is used as a reference library.	-	No standard dataset.
2015 [66]	Hindi	Unsupervised	Fuzzy Graph Connectivity Measures	Different global and local fuzzy graph connectivity measures are computed to find the meaning of a polysemous word.	Used Health corpus.	Performance increases by 8% when we use Fuzzy Hindi WordNet.	-
2016 [51]	Hindi	Knowledge-Based	Tri-Gram and Bi-Gram	Lesk's approach is applied to tri-gram and bi-gram verb words.	15 words of verbs are used as a dataset with 103 test instances.	52.98% with bi-gram and 33.17% with tri-gram.	Only work for verb words.
2016 [59]	Hindi	Supervised	Cosine Similarity	The cosine similarity of vectors, created from input query and senses from Wordnet, is calculated to determine the meaning of the word.	dataset of 90 Hindi ambiguous word	78.99%	It does not perform part-of-speech disambiguation for word categories other than nouns, such as adjectives, adverbs, etc.
2017 [68]	Gujarati	Unsupervised	Genetic Algorithm	A genetic algorithm is used.	-	-	-
2018 [60]	Gurumukhi	Supervised	K-NN	KNN classifier is used to find the similarity between vectors of input words and their meaning in Wordnet.	Punjabi Corpora of 100 sense tagged words is used.	The accuracy varies for each word, with the highest being 76.4% and the lowest being 53.6%.	The size of the dataset is too small.
2018 [61]	Punjabi	Supervised	Naive Bayes	Naive Bayes classifier, with Bow and collocation model as feature extraction technique, is used.	corpus of 150 ambiguous words having 6 or more senses taken from Punjabi word net	89% with BoW and 81% with the collocation model.	One word disambiguation per context.

Table 2. *Cont.*

Year (Ref.)	Language	Technique	Method	Specification	Dataset Used	Accuracy	Comments
2019 [69]	Hindi	Unsupervised	Word Embedding	Two-word embedding techniques, i.e., Skip-gram and CBow are used with cosine similarity to deduce the correct sense of the world.	-	52%	Semantic relations such as hypernyms, hyponyms, etc., are not used for the creation of sense vectors.
2019 [52]	Hindi	Knowledge-Based	Fuzzified Semantic Relations	Fuzzified semantic relations along with FHWN are used for WSD.	-	58–63%	There is uncertainty associated with fuzzy values. Values assigned to fuzzy memberships are based on the intuition of annotators.
2019 [53]	Hindi	Knowledge-Based	Lesk	Lesk algorithm is used to disambiguate the words.	A corpus of 3000 ambiguous sentences is used.	71.43%	POS tagger is not used
2019 [54]	Bengali	Knowledge-Based	Sense Induction	The semantic similarity measure is calculated for various sense clusters of ambiguous words.	A test set of 10 Bengali words is used.	63.71%	Classification of senses is not performed.
2021 [55]	Hindi	Knowledge-Based	Score-Based Modified Lesk	A scoring technique is utilized for advancing the performance of the Lesk algorithm.	-	-	Due to the segregation of only a part of the data from WordNet, the database needs to be queried repeatedly.
2021 [70]	Malyalam	Unsupervised	Semantic Features and Latent Dirichlet Allocation	An unsupervised LDA-based approach using semantic features has been applied for the target word sense disambiguation of the Malayalam language.	A dataset of 1147 contexts of polysemous words is used.	80%	LDA does not take into account the positional parameters within the context.

Table 2. *Cont.*

Year (Ref.)	Language	Technique	Method	Specification	Dataset Used	Accuracy	Comments
2021 [71]	Hindi	Unsupervised	Word Embeddings	Various word embedding technique has been used for WSD and experiments shows that Word2Vec performs better than all.	Hindi word embeddings were generated using articles sourced from Wikipedia.	54%	Further enhancements can be achieved by incorporating additional similarity metrics and incorporating sentence or phrase-level word embeddings into the approach.
2022 [67]	Hindi	Unsupervised	Co-operative Game Theory	Co-operative game theory along with Concept Net is used. It mitigated the influence of variations in membership values of fuzzy relations..	Health and tourism dataset and a manually created dataset from Hindi newspaper articles.	66%	-
2022 [56]	Hindi	Knowledge-Based		A complete framework named "HindiWSD" is developed in this that uses the knowledge-based modified Lesk algorithm.	A dataset of 20 ambiguous word along with Hindi WordNet is used.	71%	Dataset size is small.
2022 [72]	Hindi	Unsupervised	Genetic Algoritm	After pre-processing and creating the context bag and sense bag, GA is employed. In GA, selection, crossover and mutation are applied for the disambiguation of the word.	A manually created dataset is used.	80%	Only worked with nouns.

In the field of WSD for Hindi, the availability of high-quality data has been a challenge due to the resource-scarce nature of the language. However, there have been efforts to create and utilize datasets and benchmarks for Hindi WSD. Table 3 provide an overview about some common datasets and benchmarks that have been used or recognized in this field a:

Table 3. Data Sources available for Hindi WSD.

Data Source/Benchmark	Description
Hindi WordNet	Lexical database providing synsets and semantic relations for word senses in Hindi.
SemEval Hindi WSD Task	Part of the SemEval workshops, offering annotated datasets, evaluation metrics, and tasks for WSD in multiple languages.
Sense-Annotated Corpora	Manually annotated text segments where words are tagged with their corresponding senses from Hindi WordNet.
Cross-Lingual Resources	Leveraging resources from related languages with more data for WSD and transferring knowledge across languages.
Parallel Corpora	Using texts available in multiple languages to align senses and perform cross-lingual WSD.
Indigenous Corpora	Domain-specific or genre-specific corpora in Hindi, focusing on specific areas such as medicine, technology, or literature.
Supervised Approaches	Using a small annotated dataset for training models, often involving manually sense-tagged instances.
Unsupervised Approaches	Employing techniques such as clustering or distributional similarity without relying heavily on labeled data.
Contextual Embeddings	Utilizing pretrained models such as BERT to capture rich semantic information from large text corpora.

Because of the limitations in resources, the domain of Hindi WSD may not possess an equivalent abundance of universally accepted benchmarks as observed in more resource-endowed languages. As a result, researchers frequently modify techniques and methodologies drawn from other languages. Moreover, they occasionally amalgamate existing resources with data augmentation strategies to elevate their model's efficacy. The task of formulating more expansive and varied sense-annotated datasets and benchmarks continues to be a persisting challenge within this sphere.

4.4. Research Gaps and Future Scope

Hindi is a rich language in terms of users and information available in the Hindi language, and not much work has been performed on this. These are some of the research gaps, with the majority of the work involving nouns. Word lemmatization, which could improve accuracy even further, is not carried out, and one of the difficulties is understanding the idiomatic words. There is no standard sense annotated dataset available for supervised approaches. Using better methods or a hybrid model also has the potential to improve accuracy. Significant efforts have been dedicated to research and development for the English language, but Hindi, as the top fourth language in the world in terms of native speakers, is still in its infancy stage in the case of WSD. There is still a significant amount of work to be performed for the Hindi language. There is a lot of scope for improving accuracy, as well as other challenges, such as morphology, etc., that need to be solved.

5. Conclusions

This article summarizes several techniques utilized for the disambiguation of word senses based on Hindi literary sources. The classification of Hindi WSD tasks has categorized its methods into sections: supervised learning-based methods, knowledge-based methods, and unsupervised and supervised ones. Several types of knowledge-based, supervised, and unsupervised techniques are reviewed. Every approach has its own set of rules for working and helps in solving a particular type of problem. In order to achieve superior outcomes with supervised methods, it is necessary to create an annotated dataset. Creating an annotated dataset can be both difficult and costly. However, the use of unannotated datasets with unsupervised approaches generally produces less favorable results than those produced using supervised techniques. Tackling resource-scarce languages effectively requires a knowledge-intensive approach. A comparative analysis of various approaches has been conducted, providing insights into the work undertaken by different researchers in the field. In conclusion, each category of WSD techniques offers distinct advantages and faces specific challenges. Supervised techniques excel in accuracy and fine-grained disambiguation but require labeled data and may struggle with generalization. Unsupervised techniques are flexible, scalable, and adapt well to languages with limited resources, yet they may encounter sense overlapping and lack semantic interpretation. Knowledge-based techniques leverage external resources effectively but heavily rely on the quality of knowledge bases. The choice of technique depends on task requirements, data availability, and language characteristics. Hybrid models, combining different techniques, can effectively address limitations and improve overall WSD performance, providing a tailored approach for specific applications and language contexts.

Author Contributions: Conceptualization, V.G. and N.M.; methodology, V.G.; software, R.K.; validation, V.G., S.P. and G.B.; formal analysis, N.M.; investigation, V.G.; resources, N.C.; data curation, V.G.; writing—original draft preparation, V.G. and S.P.; writing—review and editing, G.B.; visualization, R.K.; supervision, G.B.; project administration, N.C.; funding acquisition, N.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zipf, G.K. *Human Behavior and the Principle of Least Effort*; Addison-Wesley Press: Oxford, UK, 1949.
2. Wilks, Y.; Fass, D. The preference semantics family. *Comput. Math. Appl.* **1992**, *23*, 205–221. [CrossRef]
3. Navigli, R. Word sense disambiguation: A survey. *ACM Comput. Surv.* **2009**, *41*, 1459355. [CrossRef]
4. Vickrey, D.; Biewald, L.; Teyssier, M.; Koller, D. Word-sense disambiguation for machine translation. In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP 2005), Vancouver, BC, Canada, 6 October 2005; pp. 771–778. [CrossRef]
5. Carpuat, M.; Wu, D. Improving statistical machine translation using word sense disambiguation. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, 28–30 June 2007; pp. 61–72.
6. Pu, X.; Pappas, N.; Henderson, J.; Popescu-Belis, A. Integrating Weakly Supervised Word Sense Disambiguation into Neural Machine Translation. *Trans. Assoc. Comput. Linguist.* **2018**, *6*, 635–649. [CrossRef]
7. Plaza, L.; Jimeno-Yepes, A.J.; Díaz, A.; Aronson, A.R. Studying the correlation between different word sense disambiguation methods and summarization effectiveness in biomedical texts. *BMC Bioinform.* **2011**, *12*, 355. [CrossRef] [PubMed]
8. Madhuri, J.N.; Ganesh Kumar, R. Extractive Text Summarization Using Sentence Ranking. In Proceedings of the 2019 International Conference on Data Science and Communication (IconDSC), Bangalore, India, 1–2 March 2019; pp. 19–21. [CrossRef]
9. Carpineto, C.; Romano, G. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.* **2012**, *44*, 2071390. [CrossRef]
10. Sharma, N.; Niranjan, P.S. Applications of Word Sense Disambiguation: A Historical Perspective. *IJERT* **2015**, *3*, 1–4.
11. Sumanth, C.; Inkpen, D. How much does word sense disambiguation help in sentiment analysis of micropost data? In Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Lisboa, Portugal, 14 July 2015; pp. 115–121. [CrossRef]
12. Xu, G.; Yu, Z.; Yao, H.; Li, F.; Meng, Y.; Wu, X. Chinese Text Sentiment Analysis Based on Extended Sentiment Dictionary. *IEEE Access* **2019**, *7*, 43749–43762. [CrossRef]
13. Chifu, A.G.; Ionescu, R.T. Word sense disambiguation to improve precision for ambiguous queries. *Open Comput. Sci.* **2012**, *2*, 398–411. [CrossRef]
14. Asim, M.N.; Wasim, M.; Khan, M.U.G.; Mahmood, N.; Mahmood, W. The Use of Ontology in Retrieval: A Study on Textual, Multilingual, and Multimedia Retrieval. *IEEE Access* **2019**, *7*, 21662–21686. [CrossRef]

15. Advaith, V.; Shivkumar, A.; Sowmya Lakshmi, B.S. Parts of Speech Tagging for Kannada and Hindi Languages using ML and DL models. In Proceedings of the 2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, India, 8–10 July 2022. [CrossRef]
16. Gadde, S.P.K.; Yeleti, M.V. Improving statistical POS tagging using Linguistic feature for Hindi and Telugu Improving statistical POS tagging using linguistic features for Hindi and Telugu. In Proceedings of the ICON-2008: International Conference on Natural Language Processing, Pune, India, 20–22 December 2008.
17. Banerjee, S.; Pedersen, T. An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In Proceedings of the Computational Linguistics and Intelligent Text Processing, Mexico City, Mexico, 17–23 February 2002; Gelbukh, A., Ed.; Springer: Berlin/Heidelberg, Germany, 2002; pp. 136–145.
18. Lesk, M. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In Proceedings of the 5th Annual International Conference on Systems Documentation, Toronto, ON, Canada, 1 June 1986; pp. 24–26. [CrossRef]
19. Mittal, K.; Jain, A. Word Sense Disambiguation Method Using Semantic Similarity Measures and Owa Operator. *ICTACT J. Soft Comput.* **2015**, *5*, 896–904. [CrossRef]
20. Mccarthy, D.; Carroll, J. Adjectives Using Automatically Acquired Selectional Preferences. *Comput. Linguist.* **2003**, *29*, 639–654. [CrossRef]
21. Ye, P.; Baldwin, T. Verb Sense Disambiguation Using Selectional Preferences Extracted with a State-of-the-art Semantic Role Labeler. In Proceedings of the Australasian Language Technology Workshop 2006, Sydney, Australia, 4–6 December 2006; pp. 139–148.
22. Sarika; Sharma, D.K. A comparative analysis of Hindi word sense disambiguation and its approaches. In Proceedings of the International Conference on Computing, Communication & Automation, Pune, India, 26–27 February 2015; pp. 314–321.
23. Walker, J.Q., II. A node-positioning algorithm for general trees. *Softw. Pract. Exp.* **1990**, *20*, 685–705. [CrossRef]
24. Parameswarappa, S.; Narayana, V.N. Decision List Preliminaries of the Kannada Language and the Basic. 2013, Volume 2. Available online: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=7620a95796c2eae4a94498fa779b00e2b25c957a (accessed on 21 May 2023).
25. Yarowsky, D. Hierarchical decision lists for word sense disambiguation. *Lang. Resour. Eval.* **2000**, *34*, 179–186.
26. Singh, R.L.; Ghosh, K.; Nongmeikapam, K.; Bandyopadhyay, S. A Decision Tree Based Word Sense Disambiguation System in Manipuri Language. *Adv. Comput. Int. J.* **2014**, *5*, 17–22. [CrossRef]
27. Rawat, S.; Kalambe, K.; Kawade, G.; Korde, N. Supervised word sense disambiguation using decision tree. *Int. J. Recent Technol. Eng.* **2019**, *8*, 4043–4047. [CrossRef]
28. Thwet, N.; Soe, K.M.; Thein, N.L. System Using Naïve Bayesian Algorithm for Myanmar Language. *Int. J. Sci. Eng. Res.* **2011**, *2*, 1–7.
29. Le, C.A.; Shimazu, A. High WSD accuracy using Naive Bayesian classifier with rich features. In Proceedings of the 18th Pacific Asia Conference on Language, Information and Computation PACLIC 2004, Tokyo, Japan, 8–10 December 2004; pp. 105–113.
30. Popov, A. Neural network models for word sense disambiguation: An overview. *Cybern. Inf. Technol.* **2018**, *18*, 139–151. [CrossRef]
31. Kumar, S.; Kumar, R. Word Sense Disambiguation in the Hindi Language: Neural Network Approach. *Int. J. Tech. Res. Sci.* **2021**, *1*, 72–76. [CrossRef]
32. Kumar, M.; Sankaravelayuthan, R.; Kp, S. Tamil word sense disambiguation using support vector machines with rich features. *Int. J. Appl. Eng. Res.* **2014**, *9*, 7609–7620.
33. Decadt, B.; Hoste, V.; Daelemans, W.; van den Bosch, A. GAMBL, genetic algorithm optimization of memory-based WSD. In Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, Barcelona, Spain, 25–26 July 2004; pp. 108–112.
34. Fix, E.; Hodges, J.L. Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. *Int. Stat. Rev. Rev. Int. Stat.* **1989**, *57*, 238. [CrossRef]
35. Hamming, R.W. Error detecting and error correcting codes. *Bell Syst. Tech. J.* **1950**, *29*, 147–160. [CrossRef]
36. Revesz, P.Z. A Generalization of the Chomsky-Halle Phonetic Representation using Real Numbers for Robust Speech Recognition in Noisy Environments. In Proceedings of the 27th International Database Engineered Applications Symposium, Heraklion, Greece, 5–7 May 2023; pp. 156–160. [CrossRef]
37. Brody, S.; Navigli, R.; Lapata, M. Ensemble methods for unsupervised WSD. In Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, Stroudsburg, PA, USA, 17–18 July 2006; Volume 1, pp. 97–104. [CrossRef]
38. Freund, Y.; Schapire, R.E. A Short Introduction to Boosting. 1999. Available online: https://api.semanticscholar.org/CorpusID:9621074 (accessed on 20 December 2022).
39. Martín-Wanton, T.; Berlanga-Llavori, R. A clustering-based approach for unsupervised word sense disambiguation. *Proces. Leng. Nat.* **2012**, *49*, 49–56.
40. Lin, D. Automatic retrieval and clustering of similar words. *Proc. Annu. Meet. Assoc. Comput. Linguist.* **1998**, *2*, 768–774. [CrossRef]
41. Pantel, P.A. Clustering by Committee. 2003, pp. 1–137. Available online: https://www.patrickpantel.com/download/papers/2003/cbc.pdf (accessed on 25 January 2023).

42. Silberer, C.; Ponzetto, S.P. UHD: Cross-lingual word sense disambiguation using multilingual Co-occurrence graphs. In Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, Sweden, 12 July 2010; pp. 134–137.
43. Miller, G.A. WordNet: A Lexical Database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
44. Bhattacharyya, P. IndoWordnet. In *The WordNet in Indian Languages*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 3785–3792. [CrossRef]
45. Sinha, M.; Reddy, M.K.; Bhattacharyya, P.; Pandey, P.; Kashyap, L. Hindi Word Sense Disambiguation. 2004. Available online: https://api.semanticscholar.org/CorpusID:9438332 (accessed on 25 January 2023).
46. Singh, S.; Siddiqui, T.J. Evaluating effect of context window size, stemming and stop word removal on Hindi word sense disambiguation. In Proceedings of the 2012 International Conference on Information Retrieval & Knowledge Management, Kuala Lumpur, Malaysia, 13–15 March 2012; pp. 1–5. [CrossRef]
47. Kumar Vishwakarma, S.; Vishwakarma, C.K. A Graph Based Approach to Word Sense Disambiguation for Hindi Language. *Int. J. Sci. Res. Eng. Technol.* **2012**, *1*, 313–318. Available online: www.ijsret.org (accessed on 25 January 2023).
48. Singh, S.; Singh, V.K.; Siddiqui, T.J. *Hindi Word Sense Disambiguation Using Semantic Relatedness Measure BT-Multi-Disciplinary Trends in Artificial Intelligence*; Ramanna, S., Lingras, P., Sombattheera, C., Krishna, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 247–256.
49. Singh, S.; Siddiqui, T.J. Role of semantic relations in Hindi Word Sense Disambiguation. *Procedia Comput. Sci.* **2015**, *46*, 240–248. [CrossRef]
50. Sawhney, R.; Kaur, A. A modified technique for Word Sense Disambiguation using Lesk algorithm in Hindi language. In Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Delhi, India, 24–27 September 2014; pp. 2745–2749. [CrossRef]
51. Gautam, C.B.S.; Sharma, D.K. Hindi word sense disambiguation using lesk approach on bigram and trigram words. In Proceedings of the International Conference on Advances in Information Communication Technology & Computing, Bikaner, India, 12–13 August 2016; pp. 1–5. [CrossRef]
52. Jain, G.; Lobiyal, D.K. Word sense disambiguation of Hindi text using fuzzified semantic relations and fuzzy Hindi WordNet. In Proceedings of the 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 10–11 January 2019; pp. 494–497. [CrossRef]
53. Sharma, P.; Joshi, N. Knowledge-Based Method for Word Sense Disambiguation by Using Hindi WordNet. *Eng. Technol. Appl. Sci. Res.* **2019**, *9*, 3985–3989. [CrossRef]
54. Sau, A.; Amin, T.A.; Barman, N.; Pal, A.R. Word sense disambiguation in bengali using sense induction. In Proceedings of the 2019 International Conference on Applied Machine Learning (ICAML), Bhubaneswar, India, 25–26 May 2019; pp. 170–174. [CrossRef]
55. Tripathi, P.; Mukherjee, P.; Hendre, M.; Godse, M.; Chakraborty, B. Word Sense Disambiguation in Hindi Language Using Score Based Modified Lesk Algorithm. *Int. J. Comput. Digit. Syst.* **2021**, *10*, 939–954. [CrossRef]
56. Yusuf, M.; Surana, P.; Sharma, C. HindiWSD: A Package for Word Sense Disambiguation in Hinglish & Hindi. In Proceedings of the 6th Workshop on Indian Language Data: Resources and Evaluation (WILDRE-6), Marseille, France, 20–25 June 2022; pp. 18–23.
57. Purohit, A.; Yogi, K.K. A Comparative Study of Existing Knowledge Based Techniques for Word Sense Disambiguation. In Proceedings of the International Joint Conference on Advances in Computational Intelligence, Online, 23–24 October 2021; Uddin, M.S., Jamwal, P.K., Bansal, J.C., Eds.; Springer Nature: Singapore, 2022; pp. 167–182.
58. Singh, S.; Slddiqui, T.J.; Sharma, S.K. Naïve bayes classifier for hindi word sense disambiguation. In Proceedings of the 7th ACM India Computing Conference, Nagpur, India, 9 October 2014. [CrossRef]
59. Sarika; Sharma, D.K. Hindi word sense disambiguation using cosine similarity. In Proceedings of the Advances in Intelligent Systems and Computing, Athens, Greece, 29–31 August 2016.
60. Walia, H.; Rana, A.; Kansal, V. A Supervised Approach on Gurmukhi Word Sense Disambiguation Using K-NN Method. In Proceedings of the 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 11–12 January 2018; pp. 743–746. [CrossRef]
61. pal Singh, V.; Kumar, P. Naive Bayes classifier for word sense disambiguation of Punjabi Language. *Malaysian J. Comput. Sci.* **2018**, *31*, 188–199. [CrossRef]
62. Mishra, B.K.; Jain, S. *Word Sense Disambiguation for Hindi Language Using Neural Network BT-Advancements in Smart Computing and Information Security*; Rajagopal, S., Faruki, P., Popat, K., Eds.; Springer Nature: Cham, Switzerland, 2022; pp. 14–25.
63. Mishra, N.; Yadav, S.; Siddiqui, T.J. An Unsupervised Approach to Hindi Word Sense Disambiguation. In Proceedings of the First International Conference on Intelligent Human Computer Interaction, Rome, Italy, 20–23 January 2009.
64. Jain, A.; Lobiyal, D.K. Unsupervised Hindi word sense disambiguation based on network agglomeration. In Proceedings of the 2015 International Conference on Computing for Sustainable Global Development, INDIACom 2015, New Delhi, India, 11–13 March 2015.
65. Nandanwar, L. Graph connectivity for unsupervised Word Sense Disambiguation for Hindi language. In Proceedings of the ICIIECS 2015—2015 IEEE International Conference on Innovations in Information, Embedded and Communication Systems, Coimbatore, India, 19–20 March 2015.

66. Jain, A.; Lobiyal, D.K. Fuzzy Hindi wordnet and word sense disambiguation using fuzzy graph connectivity measures. *ACM Trans. Asian Low-Resource Lang. Inf. Process.* **2015**, *15*, 2790079. [CrossRef]
67. Jain, G.; Lobiyal, D.K. Word Sense Disambiguation Using Cooperative Game Theory and Fuzzy Hindi WordNet Based on ConceptNet. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2022**, *21*, 3502739. [CrossRef]
68. Vaishnav, Z.B. Gujarati Word Sense Disambiguation Using Genetic Algorithm. 2017. Available online: https://api.semanticscholar.org/CorpusID:212514785 (accessed on 25 January 2023).
69. Kumari, A.; Lobiyal, D.K. Word2vec's Distributed Word Representation for Hindi Word Sense Disambiguation. In Proceedings of the 16th International Conference, ICDCIT 2020, Bhubaneswar, India, 9–12 January 2020. [CrossRef]
70. Sruthi, S.; Kannan, B.; Paul, B. Improved Word Sense Determination in Malayalam using Latent Dirichlet Allocation and Semantic Features. *ACM Trans. Asian Low-Resource Lang. Inf. Process.* **2022**, *21*, 3476978. [CrossRef]
71. Kumari, A.; Lobiyal, D.K. Efficient estimation of Hindi WSD with distributed word representation in vector space. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 6092–6103. [CrossRef]
72. Bhatia, S.; Kumar, A.; Khan, M. Role of Genetic Algorithm in Optimization of Hindi Word Sense Disambiguation. *IEEE Access* **2022**, *10*, 3190406. [CrossRef]
73. Jha, P.; Agarwal, S.; Abbas, A.; Siddiqui, T. Comparative Analysis of Path-based Similarity Measures for Word Sense Disambiguation. In Proceedings of the 2023 3rd International Conference on Artificial Intelligence and Signal Processing (AISP), Vijayawada, India, 18–20 March 2023; pp. 1–5. [CrossRef]

information

MDPI

Article

Agile Logical Semantics for Natural Languages

Vincenzo Manca

Dipartimento di Informatica, University of Verona, 37134 Verona, Italy; vincenzo.manca@univr.it

Abstract: This paper presents an agile method of logical semantics based on high-order Predicate Logic. An operator of *predicate abstraction* is introduced that provides a simple mechanism for logical aggregation of predicates and for logical typing. Monadic high-order logic is the natural environment in which predicate abstraction expresses the semantics of typical linguistic structures. Many examples of logical representations of natural language sentences are provided. Future extensions and possible applications in the interaction with chatbots are briefly discussed as well.

Keywords: logical semantics; predicate logic; natural language processing; large language models

Citation: Manca, V. Agile Logical Semantics for Natural Languages. *Information* **2024**, *15*, 64. https://doi.org/10.3390/info15010064

Academic Editor: Peter Revesz

Received: 6 December 2023
Revised: 17 January 2024
Accepted: 19 January 2024
Published: 21 January 2024

1. Introduction

Epochal changes and new possibilities in the interaction between humans and artificial systems capable of processing information have been brought about by the recent advances in Natural Language Processing (NLP), which are based on Machine Learning and Artificial Neural Networks [1–4]. Large Language Models (LLMs) [5,6], in particular, represent the start of a new line of development that will have enormous implications for the entire field of artificial intelligence and numerous applications involving our societies globally. LLMs are the foundation of recent systems that are widely available to the public.

The kind of "understanding" that these systems are capable of achieving in conversation with humans is among their most contentious features. There are a wide range of opinions in the current debate between the extremes that they (i) converse without really understanding the other person and (ii) converse while gaining knowledge that could eventually approach that of humans and animals. In any event, these systems do display intelligent characteristics, making consideration of broad approaches to natural language text interpretation a critical theme for the development of LLM systems in the future.

Semantics is a very old topic; Leibniz is credited with the earliest modern mathematical formulation of it in his **Characteristica Universalis** [7].

After millennia of development, the logical representation of natural language texts is today a well developed field with a vast body of books and articles. Specifically, in the 1970s, Richard Montague, a student of Alfred Tarski (who founded both set-theoretic model theory and logical semantics [8]), developed a valuable theory proving that higher-order predicate logic generates coherent and comprehensive representations of texts [9–11]. Richard Montague's famous article "English as a Formal Language" was followed by similar works. Montague's theory s formally complex, using *intensional logic* and Alonzo Church's lambda abstraction [12].

In short, from Montague's point of view every linguistic element has a semantic that is provided by a high-order function that is represented in an appropriate space by a lambda term. Our formalism, as we will demonstrate, enables us to formalize sentences in natural languages by decomposing them into their component parts—all predicates associated with words—and joining these parts with constants and logical operations (connected concepts are provided in [13]). A logical operator of "predicate abstraction", which is present neither in Montague's work nor in analogous subsequent logical approaches [14,15], provides an advancement of Montague's grammars in terms of simplification of the logical apparatus and adherence to the linguistic structures. Moreover, monadic high-order predicates allow us to eliminate variables.

Apart from Montague's approach, formalisms of logical representation constitute a large field of investigation in artificial intelligence [16,17]. However, the spirit and the finalities of these systems are very different from those of the present work. In fact, they are essentially oriented towards knowledge representation (KR), very often focusing on specific knowledge domains (e.g., programming, datasets, query languages, semantic webs, belief revision, medicine). In these contexts, natural language is considered an instrument on which representations are based rather than an object of investigation in itself. Moreover, they use variables, first-order or second-order logic, and modal or temporal operators, and the rules of composition are very complex, making KR languages comparable to programming languages. In certain cases they differ radically from classical predicate logic, and can follow very different principles and presuppositions [17].

Although the basis of our formalism is logically sophisticated (high-order predicate logic, logical types, lambda abstraction), we can explain the method in a very intuitive way because monadic predicates naturally resemble the conceptual organization of words and completely avoid variables. The ability to produce correct logical representations lies essentially in the choice of the involved constants, the right aggregation of parts by means of parentheses, and the right logical types of the constituents, which is managed using the operator of predicate abstraction. The simplicity of the formalism is proven by the conversation with ChatGPT 3.5 reported in the final section, where, after one page of conversation, the chatbot is able to show a basic familiarity with the presented formalism.

The main ingredients of our logical representations are words, constants, parentheses, and predicate abstraction. This means that semantics reduces to a relational system of words from which morphology and syntax are removed and the logical essence of the relationship between words is extracted. The relevance of this for LLM models could be considerable, and surely needs further analyses and experiments that can be developed using the most recent chatbots. In addition, as addressed in our conclusions, this fact raises a number of problems around the proprietary nature of these systems, as training strategies and finalities are under the control of the companies producing and maintaining them.

2. Materials and Methods

In this section, we first define the main aspects of logical semantics, then outline predicate high-order logic by providing the first examples of the logical representation of sentences.

2.1. Logical Semantics

Let us begin by observing the intrinsic principle of **duality** in semantics. Meanings denote both objects and relations between them; therefore, when associating meanings with symbolic expressions of a certain type, it is necessary to presuppose both objects and relations.

What is essential is the **application** of a predicate to complementary entities of the application, called **arguments**. The **proposition** obtained as result of this application expresses the occurrence of relationships between the two types of entities. We can write

$$P(a, b)$$

to express the validity of a relation associated with P on the **arguments** a, b (in the order they appear). P is called a **predicate**, and denotes a relation; thus, $P(a, b)$ is called a **predication** or **atomic proposition**. The **individual constants** a, b designate the arguments of the predicate P.

However, because a predicate can be an argument for a predicate of a higher type, predicates are arranged along a hierarchy of levels, or **logical types**; according to Russell's theory of logical types, this situation can occur indefinitely [18,19].

In addition, it is possible to conceive of relations that are simultaneously both objects and relations. However, as these possibilities often lead to logical inconsistencies, they should only be considered in specific and well-controlled contexts with appropriate precautions. We exclude them from the following discussion, assuming individuals at level zero and predicates of levels 1, 2, 3, . . . (the semantics of natural languages rarely require predicates of order higher than 3).

The **negation** of $P(a, b)$,

$$\neg\mathbf{P}(\mathbf{a}, \mathbf{b})$$

indicates the opposite of $P(a, b)$, i.e., its non-validity. The **disjunction** is a proposition,

$$\mathbf{P}(\mathbf{a}, \mathbf{b}) \vee \mathbf{P}(\mathbf{b}, \mathbf{a})$$

indicating that at least one of the two propositions connected by $\vee$ is true, while the **conjunction**

$$\mathbf{P}(\mathbf{a}, \mathbf{b}) \wedge \mathbf{P}(\mathbf{b}, \mathbf{a})$$

indicates that both propositions connected by $\wedge$ are true. The arrow $\rightarrow$ denotes **implication**; thus, the proposition

$$\mathbf{P}(\mathbf{a}, \mathbf{b}) \rightarrow \mathbf{P}(\mathbf{b}, \mathbf{a})$$

read as "if P(a, b), then P(b, a)" is equivalent to

$$\neg\mathbf{P}(\mathbf{a}, \mathbf{b}) \vee \mathbf{P}(\mathbf{b}, \mathbf{a})$$

and finally

$$\mathbf{P}(\mathbf{a}, \mathbf{b}) \leftrightarrow \mathbf{P}(\mathbf{b}, \mathbf{a})$$

is the logical **equivalence** equal to $(\mathbf{P}(\mathbf{a}, \mathbf{b}) \rightarrow \mathbf{P}(\mathbf{b}, \mathbf{a})) \wedge (\mathbf{P}(\mathbf{b}, \mathbf{a}) \rightarrow \mathbf{P}(\mathbf{a}, \mathbf{b}))$.

The symbols $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ are called **connectives** (negation, disjunction, conjunction, implication, equivalence), while the symbols $\forall$ and $\exists$ are called **quantifiers** (universal, existential).

If we consider a variable x, then

$$\forall x P(x, b)$$

asserts that, for every value a taken by x, $P(a, b)$ holds, while

$$\exists x P(x, b)$$

asserts that there exists a value a of x for which $P(a, b)$ holds. Connectives between propositions can be extended to predicates. In particular, if P and Q are predicates with only one argument, then $(P \rightarrow Q)$ denotes the predicate such that $(P \rightarrow Q)(a)$ holds when proposition $(P(a) \rightarrow Q(a))$ holds.

2.2. Formalizing Natural Language Sentences

Predicate logic [10,12,20] is a formal system used to represent the logical structure of propositions. Chapter 6 of [20] develops, in more than 100 pages, the first modern attempt at logical analysis of natural language in terms of predicate logic. It provides a way to express relationships between objects and describe actions, properties, and concepts. In this text, we explore how high-order predicate logic can be used to represent the meaning of sentences and concepts in natural language in a systematic and agile way. The method is independent from any specific language, and is adequate for teaching logical analysis to artificial systems.

In predicate logic, we have three main components.

Predicates, Objects, and Logical Symbols

Predicates: these are symbols that represent relationships or properties. They describe how objects are related to each other or specify attributes of objects; for example, "love", "eat", and "happy" are predicates.

Objects: these are the entities to which predicates are applied. They can be individuals, things, or concepts. In natural language, objects can include people and animals as well as relations and properties. In this sense, there is a need for both predicates that can be applied to objects and for objects that are predicates to which other predicates can be applied.

Logical Symbols: **connectives** and **quantifiers** are used to express the logical operations $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \forall, \exists$. Parentheses and commas are additional symbols that are needed for writing logical formulas.

Objects and predicates are denoted by: (i) **constants** (letters or strings) denoting objects and relations (at every level) and (ii) **variables** (letters or strings, different from those used for constants) ranging over objects and relations (at every level). We adopt a convention that we call of **implicit typification**, in which: (i) lowercase letters denote individuals (objects at zero level); (ii) strings of letters with one uppercase letter denote first-order predicates (over individuals); (iii) strings with two uppercase letters denote second-order predicates (over first-order predicates); and (iv) analogously for the third order and higher orders. Strings of letters including x, y, z, X, Y, Z (possibly with subscripts or superscripts) are variables, while strings including other letters (lowercase or uppercase, possibly with subscripts or superscripts) are constants. In this way, the form of a string assigns to it the role of a constant or a variable and determines its logical type.

Predicates are associated with words in a given language. In this case, the logical types of such predicates can be deduced from the types of their arguments.

A **predicative theory** is provided by a list of propositions (as indicated below, on subsequent lines).

Below, we affirm a principle whose validity has been proven by the applications of Mathematical Logic from the late 19th century to the present day.

The semantics of every symbolic expression can always be reduced to an appropriate predicative theory.

In practice, predicative theories use additional symbols to make them easier to read and write. For example, the equality symbol $=$ is used to affirm that two symbolic expressions have the same meaning. In formal terms, equality is defined by the following proposition:

$$a = b \leftrightarrow \forall X(X(a) \leftrightarrow X(b)).$$

However, all symbols extending the kernel of Predicate Logic can be formally defined in the basic setting provided above, and can be reduced to Frege's logical basis of negation, universal quantifier, and implication $(\neg, \forall, \rightarrow)$.

Predicates associated with words include: (i) **lexemes** from a dictionary (in a predefined language), including proper names; and (ii) grammatical elements, called **grammemes**.

Obviously, the choice of dictionary determines the lexemes, while grammatical predicates depend on the terminological choices of the reference grammar. For example, we could use "ComplOgg" to indicate the role of an object complement or consider transitive verbs as predicates with two arguments (subject and object). For example, "a loves b" becomes $Love(a) \wedge ComplOgg(Love, b)$, or: $Love(a, b)$.

Furthermore, we can consider the predicate "I" or a predicate such as "1st-Pers-sing" (first-person, singular), and analogously for pronouns, prepositions, and conjunctions. Even a proper noun is a predicate; thus, Julia(a) indicates that "a" is named "Julia".

Thus, in predicative theory, the simple sentence "I love Helen" is expressed as

I(a)
Helen(b)
Love(a, b).

I(a): this indicates that the individual constant a is the "I" of the sentence.

Helen(b): this indicates that the individual constant b denotes an individual named "Helen".

Love(a, b): this asserts that "a loves b".

Of course, grammatical terminology is entirely arbitrary, and any equivalent terminology essentially express the same logical relationships between objects and predicates.

Let us consider the sentence "Yesterday, I was walking without shoes". Its predicative representation is as follows, where $\widehat{\text{Walk}}$ denotes the predicate abstraction of "Walk", which we explain in the next section:

I(a)
$\widehat{\text{Walk}}$(P)
$Without\text{–}shoe(x) = \forall y(Wear(x, y) \rightarrow \neg Shoe(y))$
$Without\text{–}shoe(a)$
PastProgressive(P)
YesterDay(P)
P(a).

Intuitively, the formalization of the sentence can be paraphrased as: (1) P is a walking motion; (2) a is without shoes (any object that a is wearing is not a shoe); (3) P is yesterday and is in the past (imperfect); (4) the constant a is the "I" of the sentence; (5) a satisfies predicate P.

3. Results

In this section, the logical operator of predicate abstraction in introduced, which is related to Church's lambda abstraction. Logical representations of a Chinese sentence are provided and High-order Monadic Logic (HML) is introduced, which is a special kind of high-order Predicate Logic. Finally, many examples of logical representations in HML are provided.

3.1. Predicate Abstraction

Predicate abstraction is a powerful logical operation in the context of natural languages. It allows us to elevate the logical order of a predicate. When we say $Love(a)$, we mean that individual a loves someone; however, when $\widehat{\text{Love}}$(P) holds, this means that P possesses the property of loving. Thus, P is a predicate including all the typical characteristics of loving, because $\widehat{\text{Love}}$ denotes a predicate over first-order predicates, which is a second order predicate.

We present the following informal definition of the Predicate Abstraction operator:

Given a first-order predicate $Pred$, the second-order predicate $\widehat{Pred}$ is a predicate expressing the property of all predicates that imply the predicate $Pred$.

In general, when applied to a predicate of order i, the predicate abstraction operator provides a new predicate of order $i + 1$. The sentence "Every man is mortal" has the following very simple representation showing the expressive power of predicate abstraction:

$$\widehat{Mortal}(Man)$$

namely, the predicate Man has the property of all predicates that imply mortality.

By using predicate abstraction, the sentence "I love Helen" becomes:

I(a)
Helen(b)
$\widehat{\text{Love}}$(P)
P(a, b).

Apparently, this seems a way of making difficult a very simple proposition: $Love(a, b) \wedge I(a) \wedge Helen(n)$. However, in the representation above it is possible to add other propositions having a P as argument, which can enrich P with other particular aspects.

For example, the sentence "I love Helen very much" is obtained by adding a further second-order predication to P:

I(a)
Helen(b)
$\widehat{\text{Love}}$(P)
VeryMuch(P)
P(a, b).

A formal definition of Predicate Abstraction can be provided by means of "lambda abstraction", introduced by Alonzo Church around 1920. Today, we prefer to express it in Python notation. Let us consider a Python expression $\mathcal{E}(a, B, i)$ built with some operations applied to data and variables, such as $(2 * a + B[i])$, where a is an integer, B is a list of integers, and i is an index (integer):

```
def funct(a, B, i)
    result = ℰ(a, B, i)
    return result.
```

This is essentially a way of expressing the function corresponding to the expression $\mathcal{E}$, independently from the choice of variables occurring in $\mathcal{E}$ as well as from the particular values assumed by the variables, that is, the result produced by the function is the evaluation of $\mathcal{E}$ when the variables occurring in it are instantiated with the arguments of funct. This mechanism is essential in programming languages, as it distinguishes the definition of a function from its application (the calling of function) in many possible contexts. It is a basic logical mechanism on which high-order predicate logic can be founded, together with application and implication.

The following is the formalization of the prior sentence regarding "walking without shoes" using predicate abstraction:

$\widehat{\text{Walk}}$(P)
$\widehat{\text{Without-shoes}}$(P)
I(a)
PassImperf(P)
Yesterday(P)
P(a).

This second representation of the sentence is more correct than the previous one; because $\widehat{\text{Without-shoes}}$ has P as argument, it is not expressing a property of the individual a (who sometimes may wear shoes), and instead characterizes P as a property of the walking of a (together with the predicates $Yesterday$ and $PassImperf$).

It can be verified that any discourse can be rigorously represented within the logical structure outlined here.

The number of basic words in a natural language is only a few thousand, while grammatical predicates are a few hundred and logical symbols a few dozen. By adding letters for constants and variables, it is possible to express the meanings of natural language texts with predicates of logical types that generally do not exceed the third level. However, with respect to Montague's approach, predicate abstraction permits a very simple way of constructing meanings incrementally by taking a basic predication $P(a)$ and adding other propositions with high-order predicates that provide further characterizations to P. As we show, this modularity avoids many complications of Montague's semantics by providing logical representations that are very close to the linguistic form of sentences.

3.2. Representing Meaning across Languages: The Chinese Example

Let us consider a sentence in Simplified Chinese:
昨天我去海散步
(yesterday, I went for a walk by the seaside).
The following words constitute the predicates used to build the predicative representation:

昨天 YesterDay
我 I
去 Going
海 Sea
边 Side
散 Scattered
步 Step
地方 Place

A predicative theory representing this sentence is as follows (QQ and RR are second order predicate constants):

我 (a)
$\forall X(QQ(X) \leftrightarrow \forall xy(X(x,y) \rightarrow$ 去 (x, y) $\wedge$ 地方 (y)))
QQ(P)
昨天 (P)
海 (c)
边 (b, c)
$\forall X(RR(X) \leftrightarrow$ (步 (X) $\wedge$ 散 (X))
RR(P)
P(a, b).

The predicate RR expresses that the action of P (already characterized as walking) is carried out "in steps" and in a "scattered" manner, i.e., distributed in space (in English, a walking). Let us use *abs* to denote predicate abstraction. For a predication such as $(Place(b))(P)$, expressing that predicate P is located at place b, the previous representation becomes

我 (a)
(abs(去)) (P)
昨天 (P)
海 (c)
边 (b, c)
(地方) (b)
(abs(步 $\wedge$ 散)) (P)
(地方(b)) (P)
P(a).

This example demonstrates that our method is entirely independent of the language being considered; when the words have been associated with predicates, formulas can represent the sentences by indicating how predicates apply at the different logical levels.

In the last example, no variable occurs. This situation can be generalized using a particular type of high-order logic, which we present in the next section.

3.3. High-Order Monadic Logic

High-order predicate logic with only monadic (unary) predicates (HML) is a powerful environment for developing logical representations of natural language sentences. This short section provides a rigorous basis for the analysis of the logical types of high-order predicate logic. As it is more technical, readers who are not interested in the foundations of our formalizations can skip it without compromising their understanding of the following discourse.

High-order Monadic Logic (HML) can be expressed using three logical symbols:
(1) λ for (functional) **abstraction**;
(2) $\rightarrow$ for **implication**;
(3) **parentheses** ( ).

In HML, there are two categories of expressions, namely, **objects** and **types**. An object is associated with one and only one type.

There are two kinds of basic objects, **individuals** and **truth values**, with respective types *ind* and *tt*.

If σ, τ denote generic types, then $(\sigma \to \tau)$ is a type denoting functions transforming objects of type σ into objects of type τ.

For any type τ, there is an infinite list of constants and variables of that type (with τ as a subscript indicating the type).

The constants F (false) and T (true) denote the two possible truth values (of type *tt*).

In HML, there are three rules for obtaining expressions denoting objects starting from logical symbols, constants, and variables:

Abstraction rule: if ξ_σ is a σ-variable and φ denotes a τ-object, then $\lambda\xi_\sigma(\varphi)$ denotes an object of type $(\sigma \to \tau)$.

Application rule: if Θ denotes an object of type $(\sigma \to \tau)$ and ζ denotes an object of type σ, then $\Theta(\zeta)$ denotes an object of type τ. Moreover, if $\Theta[\xi]$ is an expression of type τ including a variable ξ_σ on which no λ-abstraction is applied and η is an expression of type σ, then

$$(\lambda\xi_\sigma(\Theta[\xi]))(\eta) = \Theta[\eta],$$

where $\Theta[\eta]$ denotes the expression $\Theta[\xi]$ after replacing all the occurrences of ξ_σ with η.

Implication rule: if φ and ψ denote truth values, then $(\varphi \to \psi)$ denotes a truth value. In general, if ϕ and Ψ are predicates of type $(\sigma \to tt)$, then $(\phi \to \Psi)$ is a predicate of the same type, such that for any expression η of type σ it is the case that

$$(\phi \to \Psi)(\eta) = \phi(\eta) \to \Psi(\eta).$$

It can be shown that all the logical operators of high-order predicate logic can be expressed in HML. In particular, using the symbols $\lambda, \to, T, F$, negation $\neg\varphi$ is expressed by $(\varphi \to F)$ and quantification $\forall x(\varphi)$ is expressed by $(\lambda x(\varphi(x)) = (\lambda x(T))$.

Expressions denoting truth values are called **propositions** (a predication can be considered as an "atomic proposition"), while those denoting objects of type $(\sigma \to tt)$, which we indicate with $pred_\sigma$, denote (unary) predicates of type σ. Objects of type $(ind \to tt)$ correspond to first-order predicates, and are simply indicated by *pred*, while objects of type $((ind \to tt) \to tt)$ are second-order predicates.

3.4. Predicate Abstraction in HML

Let us consider a binary predicate P over two individuals and the predication $P(a,b)$ of P over the arguments a, b. We can express this proposition by means of two unary applications: $(P'(a))(b)$, where $(P'(a))$ is the monadic predicate $(P(a,-))$ obtained by P when its first argument is put equal to a, which holds on b when $P(a,b)$ holds:

$$(P((a,-))(b) = P(a,b).$$

Therefore, P' is a function taking an individual as argument and providing the unary predicate $(P'(a))$.

Let $SecondArgument(b)$ be a second-order predicate satisfied by the monadic predicates $X(a,-)$ holding on b. Consequently,

$$P(a,b) = (P(a,-))(b) = ((SecondArgument(b))(P(a,-)).$$

However, $(P(a,-))(b)$ means that

$$(P(\widehat{a,-)})(b)(P)$$

therefore,

$$((SecondArgument(b))(P(a,-)) = (P(\widehat{a,-)})(b)(P).$$

In other words, the monadic reduction of a first-order binary predicate is definable in term of predicate abstraction. In conclusion, $P(a,b)$ is completely represented by

$$\begin{array}{l}\exists y P(a,y)\\((SecondArgument(b))(P(a,-)),\end{array}$$

and we can simply write

$$\begin{array}{l}P(a)\\(SecondArgument(b))(P).\end{array}$$

Of course, the mechanism described for binary predicates can be naturally extended to predicates of any number of arguments.

Place, time, manner, instrument, possession, and other natural language complements logically relate to an (implicit) application of predicate abstraction. Specifically, we employ an implicit predicate abstraction (after accepting an individual as an argument) to express verb complements by supplying a predicate's property. As an illustration, the predication $(with(c))(P)$ states that P has the property $with(c)$ (a property of properties), the logical type of $with$ is $(ind \to (pred \to tt))$ (in fact. $with(c)$ has type $(pred \to tt)$), and finally, $(with(c))(P)$ is a proposition (of type tt).

The monadic nature of HML enables a very synthetic way of expressing the predicative structure of sentences: enumerating all constants and listing for each of them the predicates taking a given constant as argument. For example, the previously considered sentence "Yesterday I was walking without shoes" becomes:

$$a : P$$

$$P : Yesterday, Past, Progressive, \widehat{Walk}, Whithout(Shoe)$$

The above formalization corresponds to a Python dictionary structure of the following type (where "abs" stands for predicative abstraction):

'a': ['I', 'P'], 'P': ['Yesterday', 'Past', 'Progressive', 'abs(Walk)', '(Without(Shoe))(Wear)']

It is apparent that, by avoiding the explicit use of variables, the monadic setting of HML forces the formalization to fit closely with the linguistic form. Specifically, unary predicates naturally impose high-order logical types, with consequent elimination of variables. Moreover, a constant may occur as an argument and as a predicate at the same time ($P(a), Yesterday(P)$).

Certain aspects are crucial in the determination of the above Python dictionary: (1) the introduction of the right constants to which the predicates refer; (2) possible "hidden predicates" that do not occur as words in the sentence, which generally are of grammatical nature but in the case above include the lexical term "Wear"; and (3) the logical type of predicates and the pattern according to which they apply. For example,

$$(Without(Shoe))(Wear)$$

implicitly provides the following type assignments, where $pred$ abbreviates the type $(ind \to tt)$:

Whear : $pred$
Shoe : $pred$
Without : $(pred \to (pred \to pred))$.

In natural languages, complements, modifiers (adjectival and adverbial forms), and pronouns realize the reference mechanism, usually based on grammatical marks and morphological concordance (gender, number, tense …). A pronoun refers to a component having the same marks. Moreover, the aggregation of components is realized on the basis of concordance, which corresponds to the use of parentheses in mathematical expressions.

In HML, reference is realized by means of constants and aggregation is realized by parentheses, with the different levels of application expressing the logical levels of predicates in a rigorous way.

The sentence "Mary goes home with the bike" provides the following HML translation:

$Mary(a)$
$\widehat{Go}(P)$
$(Place(b))(P)$
$(With(c))(P)$
$Home(b)$
$Bike(c)$
$The(c)$
$P(a)$.

Synthetically,

$$a : Mary, P$$

$$b : Home$$

$$c : Bike$$

$$P : \widehat{Go}, With(c), Place(b).$$

A more complex example involving a relative clause is the sentence "I am searching for a bike with a leather saddle".

We can consider the logical definition of Any as a function of type $((ind \rightarrow tt) \rightarrow ind)$ satisfying the condition

$$\forall X(\exists x(X(x)) \rightarrow X(Any(X)))$$

$I(a)$
$Progressive\text{-}present(P)$
$(Search\text{–}For(Any(Q)))(P)$
$\widehat{Leather}(Q)$
$\widehat{Saddle}(Q)$
$(Of(Bike))(Part)(Q)$
$P(a)$.

Synthetically,

$$a : P, I,$$

$$P : (Search\text{–}For(Any(Q))), Progressive\text{–}present,$$

$$Q : \widehat{Leather}, \widehat{Saddle}, (Of(Bike))(Part).$$

We collected a number of translation exercises involving different kinds of linguistic constructions, several of which were long and complex, to confirm the power and adequacy of HML to represent natural language logic. The expressive mechanisms of lambda abstraction, high-order types, application, and implication, together with parentheses and constants, apply to any kind of natural language.

4. Teaching Logic to Chatbots

Here, we report a conversation with ChatGPT 3.5 that seems to be very informative about the potential of the discourse developed in the previous sections. Logical analysis could introduce a new kind of interaction with chatbots, opening up an interesting perspective on the AI discussion.

In many aspects, the reaction of ChtGPT is really surprising and impressive. While we do not report the whole conversation, a particular point deserves attention. In many discussions, people claim that chatbots do not really understand what is being said. Certainly, at this point in time they understand in a way that is very different from the way in which humans understand; indeed, we are not even sure how we comprehend. However, in general terms, abstraction is a necessary feature of any kind of comprehension. In fact, in order to understand meanings, information has to be organized according to general rules abstracted from single cases. In a previous conversation (not reported here) focused on the comprehension of sentences, ChatGPT, when stimulated by specific questions, explained that any word corresponds to a vector of real numbers (features), and that in the process of sentence elaboration these vectors generate a vector of thousands of values. Therefore, the process of sentence synthesis/analysis corresponds to suitable operations in many-dimensional vector spaces. This means that mathematical structures and related operations are involved in the dialogue activity. In other words, this activity relies on the capability of dominating abstract structures. On this basis, it could be said that these systems exhibit a particular form of comprehension.

Conversation

I want to ask you to give logical representations of sentences, according to a simple and basic logical mechanism, that is, "predication". According to it a predicate is applied to an argument, also called the subject of predication. In this way any sentence is reduced to a list of subjects, and each of them has a list of predicates applied to it Subjects are denoted by constants represented by strings, while predicates are usually words. This basic schema becomes not easy if we consider that in the sentences of natural languages the predicates associated to the words are arranged along a natural hierarchy of logical types. In fact, there are initial predications where subjects are considered as having type zero, and their predicates type 1, but some of these predicates can be subjects of other predications where predicates are of type 2, 3, and so forth. (**Author**)

It seems like you're describing a basic approach to logical representation using predication, where sentences can be broken down into subjects and predicates, and there's a consideration of logical types. This approach aligns with predicate logic and the concept of logical types as introduced by Bertrand Russell. In this framework, you would assign types to individuals (constants/subjects) and predicates to ensure meaningful and well-formed sentences. For instance, if a constant (individual) has type 0, the predicates associated with it would be of type 1. Then, if a predicate or constant has type 1, it can be applied to subjects (constants) of type 0, and so on. Let me know if you have a specific sentence you would like to represent in this logical framework, and I can help you break it down into subjects and predicates based on your described schema. (**ChatGPT**)

Yes, exactly! Let us start with simple examples. At the beginning I provide you the representation, and then I will ask you to do the same. The first sentence is "I love Helen" In this case, Constants: are: a, b, P. Predicates are: I, Love, Helen, P, Object. Predications are: I(a), Helen(b), (abs(Love))(P), (Object(b))(P), P(a). Moreover, "abs" is an operator that indicates that abs(love) holds for predicates that imply the property of loving, while (Object(b))(P) indicated that predicate P has b as object. Let me know if this is clear, and provide me the representation of "Helen goes home with her bike". (**Author**)

Certainly, I understand the structure you've outlined. Let's apply this to the sentence "Helen goes home with her bike". Using the constants and predicates you provided:

Constants: a,b,c

Predicates: Helen, Goes, Home, With, Her, Bike, P, Object

Predications:
Helen(a)

Goes(b)
Home(c)
(abs(With))(P)
(abs(Her))(P)
(abs(Bike))(P)
(Object(c))(P)

So, the logical representation of "Helen goes home with her bike" would include these predications involving the constants and predicates you've defined. If you have further sentences or questions, feel free to share them! (**ChatGPT**)

Let me show an improvement of your representation: (**Author**)

Certainly, feel free to provide an improved representation or any adjustments you'd like to make. I'm here to assist and discuss the logical representation further. (**ChatGPT**)

Helen(a), (With(b))(P), (abs(Go)(P), P(a), Home(c), Bike(b), (Place(c))(P), (Of(a))(b). (**Author**)

Great! Let's incorporate the improvements into the logical representation for "Helen goes home with her bike".

Constants: a,b,c

Predicates: Helen, Goes, Home, With, Her, Bike, P, Object, Place, Of

Predications:
Helen(a)
(With(b))(P)
(abs(Go))(P)
P(a)
Home(c)
Bike(b)
(Place(c))(P)
(Of(a))(b)

This representation reflects the refined structure you provided. If you have more sentences or if there's anything else you'd like to discuss or modify, feel free to let me know! (**ChatGPT**)

Ok, now please, could you explain me the logical motivation of the last predication above? (**Author**)

Certainly! Predication (Of(a))(b) represents the relationship between the constant a (Helen) and b (her bike) using the predicate Of. Let's break down the logical motivation:

- Predicate Of is used to express a relationship of possession or association. In this context, it signifies the association between Helen and her bike.
- The argument a represents Helen, and b represents her bike.
- Therefore, (Of(a))(b) is read as "Helen possesses/has her bike" or "The bike of Helen".

So, in the logical structure, this predication captures the idea that the constant a (Helen) is associated with the constant b (her bike) through the predicate Of, indicating ownership or possession. This is a common way to represent relationships and associations between entities in logical frameworks. (**ChatGPT**)

Let me ask another little improvement. In the list of predicates is present Goes, but is not present Go, while in the logical representation Go is present, but Goes is absent. How do you can fix this discrepancy? (**Author**)

Certainly, we can address this discrepancy by adding both "Go" and "Goes" to the list of predicates in the representation. Here's the corrected version: (**ChatGPT**)

But now Goes is in the list of predicates, but does not occur in the representation, hence which is the correct situation? (**Author**)

I apologize for the oversight. You are correct. If "Goes" is in the list of predicates, it should also appear in the representation. (**ChatGPT**)

No, it is not the right answer. The best solution is to delete Goes, by using only Go. In fact, Goes is only a grammatical form of Go that is required by the English grammar, according to the rules of grammatical concordance, but the semantic essence of the predicate is in the infinite form Go. Please, provide the correction. (**Author**)

I appreciate the clarification. Given that "Goes" is a grammatical form derived from the verb "Go," and we want to represent the semantics using the infinitive form "Go". We can indeed simplify the list of predicates by including only "Go". Here's the corrected version:

Constants: a,b,c

Predicates: Helen, Go, Home, With, Her, Bike, P, Object, Place, Of

Predications:
Helen(a)
(With(b))(P)
(abs(Go))(P)
P(a)
Home(c)
Bike(b)
(Place(c))(P)
(Of(a))(b)

(**ChatGPT**)

What about the sentence: "Helen is going home with her bike"? (**Author**)

Let's represent the sentence "Helen is going home with her bike" using the logical structure you've outlined. I'll incorporate the new information into the representation: Where the same constants and predications are given plus the predication (abs(Is))(P).

This structure aims to capture the essence of the sentence "Helen is going home with her bike" in the logical framework you've described. If you have further questions or if there's anything else you'd like to adjust, please let me know! (**ChatGPT**)

In my opinion, this choice can be improved by a predicate linked with the usual grammatical terminology. For example, by replacing "Is" by Progressive-present. (Comment extra conversation: an inaccuracy in the list of predicates remains, which is similar to that of Go and Goes. In fact, predicate Her can be removed, after the introduction of the Of(b) predicate). (**Author**)

Certainly, we can replace the generic "Is" with a more explicit term, such as "Progressive-present," to better reflect the grammatical aspect of the sentence. (**ChatGPT**)

In a previous conversation, you told me that chatbots can learn from the conversations they have. In this case, did these exercises in logical representation teach you new things, or, once the conversation is over, does your knowledge remain the same as it was before the conversation? (**Author**)

Chatbots based on language models, like me, do not learn or store specific information from user conversations. My language model, GPT-3.5, is pre-trained on a wide range of data from the web, but it does not retain a memory of specific user conversations. Each conversation is isolated and does not impact future responses.

> I can provide consistent and contextually appropriate responses based on my pre-existing training, but I do not have the ability to 'learn' in the traditional sense. Responses are generated in real-time based on the context of the current conversation and the information available up to my last update in January 2023.
>
> Currently, there is no version of GPT (Generative Pretrained Transformer) or ChatGPT that has the ability to learn incrementally from user conversations or to maintain long-term memory of acquired information. GPT is designed as a pretrained language model; thus, each conversation is considered in isolation, and the model does not retain data between interactions. While more advanced models might be developed in the future with incremental learning or long-term memory capabilities, currently these are not available in the known versions of GPT, including GPT-3.5. It is important to note that implementing such capabilities involves ethical and security challenges, and requires careful management to avoid potential issues related to user privacy and data security. (**ChatGPT**)

5. Discussion

The ability of predicate logic to represent human thought within a minimal set of symbols is a remarkable achievement with a millennia-long history. It traces back to the earliest systems of writing, through Aristotle's syllogism and medieval logic dealing with problems of interpretation and modality, up to Renaissance logic focused on combinations of symbols.

Leibniz had a profound and visionary understanding of symbolic systems and mathematical notation. Over time, great minds such as De Morgan, Boole, Schröder, Peirce, Frege, Peano, Hilbert, Russell, Church, Tarski, and Gödel laid the foundations for an alphabet of ideas. As mathematical logic proves, this alphabet, consisting of a few logical symbols and rules for their combinations, is capable of formally representing human reasoning.

This distilled form of reason has a deep and enduring history, serving as the foundation for various mathematical and scientific theories. In particular, it provides a secure framework for set theories such as ZF (Zermelo–Fränkel) and NBG (von Neumann–Bernays–Gödel), which can express nearly all of mathematics using specific axioms.

Formalisms for representing knowledge, particularly those that are universal in nature, are applicable in a vast array of contexts. This implies that a formalism has a good chance of developing and becoming a valuable tool in scientific communication if it is more straightforward and grounded in science than others.

The examples presented in this paper and the reported conversation with ChatGPT 3.5 suggest an intriguing possibility for the development of systems exhibiting dialogue abilities, such as ChatGPT, BARD, BERT, and others; see [21,22] for analogous proposals from different perspectives.

An artificial system able to provide HML formalization of linguistic texts must provide an elaboration of an input string expressing a sentence, then yield as output the correct dictionary expressing HML propositions involving the words of the sentence. While the words occurring in the dictionary take the form of lexicon entries (lexemes), grammatical items need to appear in the dictionary of logical representations as well. This requires basic linguistic ability on the part of the system, similar to that of LLM models.

The conversation with ChatGPT shows that even when we provided the formal basis of our formalism for motivating its logical structure and links with classical predicate logic, during the interaction with ChatGPT the formalism was explained in plain English and essentially transmitted by examples and comments on concrete cases of logical analysis. It is apparent that the system shows flexibility and the ability to abstract from single cases, which are surely supported by its ability to dominate abstract structures thanks to its grounding in the logical basis of HML.

Not only was the chatbot able to follow a very constructive conversation, it correctly addressed the point of incremental learning, which of course is a strategic topic, though beyond the scope of the present paper. However, formalisms of knowledge representation, especially those of general-purpose nature, apply to an enormous number of situations. This means that if a formalism is simpler and more scientifically well-founded than others, it can surely develop and become an important instrument in scientific communication.

In the case of systematic training of chatbots by human experts, the trainers need to understand the formalism; in this case, a Python version of HML might be more appropriate. We have already noted that a logical representation reduces to a Python dictionary, and it would not be difficult to translate lambda abstractions and all the logical basis of HML into terms of suitable Python functions, classes and methods.

A critical point emerged during the last part of the conversation, namely, that whatever ChatGPT learns during a teaching interaction is completely lost at the end of the conversation. In fact, for reasons of security, even if a learning system can develop a capability of incremental learning, this cannot be free until chatbots are able to develop internal mechanisms for decision-making and control of their learning. In other words, the actual systems are closed, and training can only be developed within the companies to which these systems belong. This means that at present experiments with significant impact could only be possible in accordance with the research that is planned on these systems.

Of course, this does not mean that proposals and suggestions from external researchers are useless. On the contrary, it is important to debate and promote the circulation of new ideas that can be assimilated and integrated with those of other scientists up to the level of design and implementation that the companies acting in AI and machine learning decide to realize.

With the availability of an artificial neural network already trained in basic dialogue competence, after training the ANN to acquire competence in HML representation, an evaluation of its impact on the quality and level of language comprehension could be carried out, which may be of fundamental importance for the whole of artificial intelligence.

Surely, the epochal passage to the latest chatbots tells us that language is the main tool for knowledge acquisition and organization; therefore, a correct understanding of the logical structure of language could be the next step toward a further level of "conscious" linguistic ability.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are openly available in the cited bibliography.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
2. Mitchell, T. *Machine Learning*; McGraw Hill: New York, NY, USA, 1997.
3. Nielsen, M. *Neural Networks and Deep Learning*. 2019. On-Line Book. Available online: http://neuralnetworksanddeeplearning.com/ (accessed on 2 December 2019).
4. Werbos, P. Backpropagation Through Time: What It Does and How to Do It. *Proc. IEEE* **1990**, *78*, 1550–1560. [CrossRef]
5. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. *Neurips* **2020**, *33*, 1877–1901.
6. Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T.B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; Amodei, D. Scaling Laws for Neural Language Models. *arXiv* **2020**, arXiv:2001.08361.
7. Parkinson, G.H.R. *Leibniz Logical Papers*; Clarendon Press: Oxford, UK, 1966.
8. Tarski, A.; Givant, S. *A Formalization of Set Theory without Variables*; Colloquium Publications; America Mathematical Society: Providence, RI, USA, 1988; Volume 41.
9. Dowty, D.R.; Wall, R.E. (Eds.) *Introduction to Montague Semantics*; D. Reidel: Dordrecht, The Netherlands, 1989.
10. Kalish, D.; Montague, R. *Logic. Techniques of Formal Reasoning*; Harcourt, Brace & World: San Diego, CA, USA, 1964.
11. Thomason, R.H. (Ed.) *Formal Philosophy*; Yale University Press: New Haven, CT, USA, 1974.
12. Church, A. *Introduction to Mathematical Logic*; Princeton University Press: Princeton, NJ, USA, 1956.
13. Manca, V. A Metagrammatical Logical Formalism. In *Mathematical and Computational Analysis of Natural Language*; Martín-Vide, C., Ed.; John Benjamins: Amsterdam, The Netherlands, 1998.
14. Barwise, J. *The Situation in Logic*; Center for the Study of Language and Information: Stanford, CA, USA, 1989.

15. van Benthem, J. *Intensional Logic*; Center for the Study of Language and Information: Stanford, CA, USA, 1988.
16. van Harmelen, F.; Lifschitz, V.; Porter, B. *Handbook of Knowledge Representation*; Elsevier: Amsterdam, The Netherlands, 2008.
17. Kerr, A.D. A plea for K R. *Synthese* **2021**, *198*, 3047–3071. [CrossRef]
18. Hilbert, D.; Ackermann, W. *Princioles of Mathematical Logic*; American Mathematical Society: Providence, RI, USA, 1991.
19. Whitehead, A.N.; Russel, B. *Principia Mathematica*; Cambridge University Press: London, UK, 1910.
20. Reichenbach, H. *Symbolic Logic*; Macmillan: New York, NY, USA, 1947.
21. Pan, L.; Albalak, A.; Wang, X.; Wang, W.Y. LOGIC-LM: Empowering Large Language Models with Symbolic Solvers for Faithful Logical Reasoning. *arXiv* **2023**, arXiv:2305.12295v1.
22. Yang, Z.; Ishay, A.; Lee, J. Coupling Large Language Models with Logic Programming for Robust and General Reasoning from Text. In Proceedings of the ACL 2023, Toronto, ON, Canada, 9–14 July 2023; pp. 5186–5219.

information

MDPI

Article

D0L-System Inference from a Single Sequence with a Genetic Algorithm

Mateusz Łabędzki and Olgierd Unold *

Department of Computer Engineering, Faculty of Information and Communication Technology, University of Science and Technology, Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland; m.labedzki@protonmail.com
* Correspondence: olgierd.unold@pwr.edu.pl

Abstract: In this paper, we proposed a new method for image-based grammatical inference of deterministic, context-free L-systems (D0L systems) from a single sequence. This approach is characterized by first parsing an input image into a sequence of symbols and then, using a genetic algorithm, attempting to infer a grammar that can generate this sequence. This technique has been tested using our test suite and compared to similar algorithms, showing promising results, including solving the problem for systems with more rules than in existing approaches. The tests show that it performs better than similar heuristic methods and can handle the same cases as arithmetic algorithms.

Keywords: L-system; genetic algorithm; grammatical inference

Citation: Łabędzki, M.; Unold, O. D0L-System Inference from a Single Sequence with a Genetic Algorithm. *Information* **2023**, *14*, 343. https://doi.org/10.3390/info14060343

Academic Editor: Peter Revesz

Received: 10 May 2023
Revised: 9 June 2023
Accepted: 13 June 2023
Published: 16 June 2023

1. Introduction

The discipline of artificial life aims to create models and tools for simulating and solving complex biological problems [1]. It allows for experimentation and studies on systems imitating existing life, without its physical presence. Examples of such models are cellular automata and Lindenmayer systems [2]. The latter, sometimes called L-systems, are a type of formal grammar introduced by Astrid Lindenmayer in 1968 [3]. Their trademark is a graphical representation associated with symbols of the alphabet. Initially, they were created as a tool for modelling symmetric biological structures such as some types of plants. Using L-systems, we can try to find a solution for a very basic problem—predicting the growth of an organism, given its current state and environment. They have also been used for a plethora of other use cases, such as modelling whole cities [4], sound synthesis [5] or fractal generation [6]. They can also be used in procedural generation. After the initial model is created, minor parameters or initial state modifications can create similar-looking but still distinct objects in great numbers. While the usefulness of L-systems is not in question, they are challenging to develop, especially when they are supposed to model an existing object. In this article, an attempt at the automatic generation of deterministic, context-free L-systems (D0L-systems [3]), from an image through grammar inference, has been made using a genetic algorithm (GA).

The main contributions of the article include the following:

- A new line detection algorithm,
- Extending the current capabilities of inference algorithms for D0L-systems from a single sequence from two to at least three rules,
- Improving the execution speed of heuristic algorithms for systems with one or two rules and reducing the number of assumptions that need to be made about the grammars being inferred.

The remaining part of the article is organized as follows. Section 2 first introduces the fundamental knowledge necessary to understand the following sections and presents the existing works dealing with similar problems. Then, our approach to solving the described problem is presented. The test results and comparison to other methods are shown in Section 3, while Section 4 draws conclusions and presents further investigation areas.

2. Materials and Methods

2.1. L-Systems

L-systems comprise three elements—a set of symbols V_p called an alphabet, a starting sequence A called an axiom, and a set of rewriting rules F in the form of $Base \rightarrow Successor$. These systems work in iterations on sequences of symbols, starting with the axiom. In each iteration, a new string is created by applying every rewriting rule to the current sequence, meaning that every occurrence of the rule's base is replaced by its successor. The alphabet contains two types of symbols, terminal and non-terminal, which differ in a single aspect—a rule's base has to contain at least one non-terminal character. In the case of D0L-systems, rules have single-symbol bases, meaning a non-terminal symbol is a base of a rule.

The most recognizable property of Lindenmayer systems is the geometrical representation that is usually associated with all or some of the symbols in the alphabet. Turtle graphics is a commonly encountered method of translating sequences to geometric structures. It utilizes a cursor with a straightforward command set—it can draw a straight line, turn by angle, save the current position and tilt, and return to the previously memorized state. Each of these operations can be mapped to symbols of the L-system alphabet, making an output sequence a command list for the cursor. In Figure 1, an example structure is shown, drawn from a sequence L_3:

$$\begin{gathered} L_3 = FFF - [XY] + [XY]FF - [XY] + [XY] - [+FY - FX] \\ +[+FY - FX]FF - [XY] + [XY]FF - [XY] + [XY]- \\ [+FY - FX] + [+FY - FX] - [+FF - [XY] + [XY] \\ -FX - FF - [XY] + [XY] + FY] + [+FF - [XY]+ \\ [XY] - FX - FF - [XY] + [XY] + FY], \end{gathered}$$

which was generated in the third iteration by the system S_3, defined as:

$$S_3 = \{A = F, F \rightarrow FF - [XY] + [XY], X \rightarrow +FY, Y \rightarrow -FX\}.$$

The F, X, and Y symbols are mapped to the draw forward action, symbols [and] traditionally represent the save and return to the position actions, and the characters $+$ and $-$ command the cursor to turn by an angle of $\pm 27.5^{\circ}$.

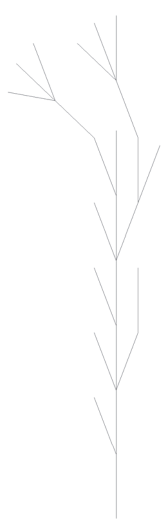


Figure 1. A structure generated by the S_3 system in the third iteration.

2.2. Grammatical Inference

As mentioned, the most significant problem with L-systems is the difficulty of creating the models. Usually, the model is supposed to imitate an existing object or create a structure that satisfies defined requirements. However, the connection between system rules and generated structures is not intuitive, making modelling difficult and usually requiring a significant amount of trial and error. This is why the need to create L-systems from existing examples automatically arose. The generation of a grammar from one or more samples is called grammatical inference [7]. During this process, three elements of L-systems need to be proposed—an alphabet, an axiom, and a set of rewriting rules. Generating correct rules is the most challenging problem because the possibilities are numerous, and their number grows exponentially with the number of rules a system can have. That is why, usually, except for small systems, this task has been approached as a search problem, and using metaheuristics has been the most common since they are designed to deal with problems with a large search space. The genetic algorithm is a metaheuristic most commonly associated with L-system inference research. It was also used in this article as a good starting reference point.

2.3. Genetic Algorithms

Genetic algorithms are metaheuristics belonging to the family of evolutionary algorithms [8,9]. Based on naturally occurring evolution and natural selection, they are commonly used for optimization and search problems where the search space is extensive, exact methods are unavailable, or the time constraints are too strict. The main component of this algorithm is a population that contains many individuals, each representing a specific solution to a problem, usually encoded as a set of values or symbols that belong to the search space. The quality of such a solution is measured in terms of fitness by a problem-specific function. To improve the quality of individuals, a few genetic operators are employed—crossover, mutation, and selection. In each iteration of the GA, individuals are selected from the population for breeding and then subjected to crossover and mutation. If better solutions emerge, they are included in the next generation, and the process repeats.

2.4. Related Works

The attempts at single-sequence D0L-system inference can be generally divided into arithmetic and evolutionary approaches. In the first group, two algorithms have been proposed [10,11], but they were constrained to solving systems with only one or two rules, with the first one also requiring a known axiom. More attempts have been made using evolutionary algorithms. Some of them used genetic programming, including one of the first ones in [12] who managed to infer a single-rule Quadratic Koch Island system with a known axiom, but also a new approach in [13] who used a genetic programming variant called bacterial programming and managed to infer systems with up to two rules. The others opted for genetic algorithms [14] or grammatical evolution with BNF grammar encoding [15]. However, both algorithms required either axiom or axiom and iteration number to be known. Even though we are dealing with a particular type of L-systems inference in this article, there is an abundance of work done for other types and applications of L-systems. One closely related research topic is grammar inference based on multiple sequences. Most interesting are relatively recent articles by J. Bernard and I. McQuillan [16–18], which our proposed algorithm is partly based on. Their work also extended to different types of L-systems—context-sensitive [19], stochastic [20,21], or temporal [22].

2.5. Inferring Grammar from a Single Input Image

The proposed approach is to parse the input image into a sequence of symbols that describe the geometrical structure generated from an L-system and then use a GA to infer the system's grammar (Figure 2). The respective steps of the proposed D0L-system induction method are described below.

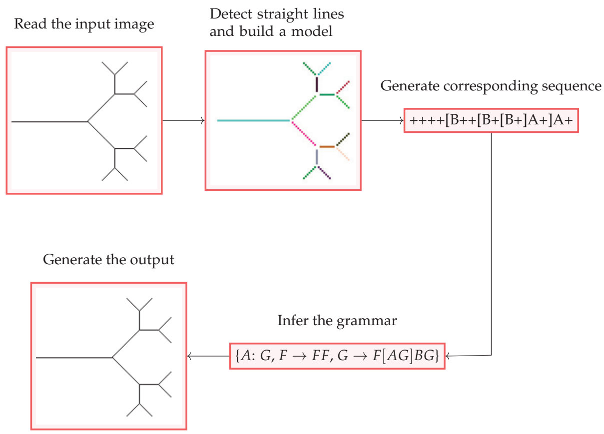


Figure 2. General Outline of the Grammar Inference Algorithm.

2.6. Image Parsing

The parsing can be done using general line detection algorithms (like D-DBSCAN [23]), but the results are often not precise enough for this application; therefore, we have employed our line detection algorithm.

The process of parsing the image into an input sequence is divided into three steps. First, all of the straight lines are detected in the image. Then, all of these lines are connected, building a model of the structure in the image so in the last step we can generate a sequence that accurately describes this model.

2.6.1. Straight Line Detection

For usage in our algorithm, we defined a line as a set of continuous points, understood as pixels in an image, each neighbouring at most two other points. A point with more than two neighbours is treated as a line intersection, and a point with only one neighbour, an edge point, is considered a line end. The neighbourhood is based on the euclidean distance between points in the image—two points are neighbours if the distance between them is not greater than $\sqrt{2}$, which is the largest distance between two touching pixels in an image.

To detect straight lines in the image, the procedure traverses the image looking for a pixel that has two neighbours and therefore belongs to a straight line according to its definition. Starting from this point, consecutive connected pixels are added to the detected line for as long as they have only two neighbours. The line detection ends when on both ends of the line the algorithm detects either an edge point or an intersection (multiple-neighbour point). Each visited point is also marked, so it is not under consideration when looking for the next lines.

One case that is not handled by the line definition is when a line changes direction. Two connected lines might conform to the definition and be detected as a single line even if they are clearly not the same line. The splitting of such lines is handled after the line detection process finishes. Given that a line is a set of points, to find a change of direction, we are looking for the largest continuous subsets of points that fit a linear model with some acceptable error.

To check if a set of points fits a linear model, an algorithm based on a simple idea is used—if the subset contains points of a straight line, the line between the first and the last point goes through all of the remaining points. It takes the first and the last point of the subset and finds a linear model that fits those points. Afterwards, it checks whether the model fits the remaining points of the subset. A model fits a point when the distance between the point and the linear model is smaller than a specified acceptable error. However, when working with indexes of pixels in an image, the accuracy is often not enough to correctly match the line to the points. That is why the points are first cast into a virtual space with higher granularity, which is a parameter of the algorithm. For example, given a granularity of 2, a pixel is divided into a grid of 5 by 5 cells (Figure 3).

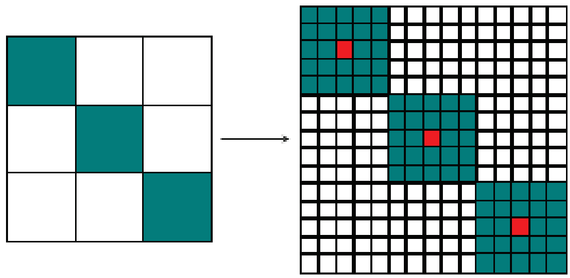


Figure 3. A few line points cast into virtual space with granularity = 2.

After casting each of the points to this virtual space, now a pixel matches the linear model if the distance from any of his cells to the linear model is smaller than the acceptable error. This effectively allows us to work with higher resolution than the original image and gives much more accurate results.

Due to inaccuracies in drawing straight lines, especially at the intersections of multiple lines, there are usually some points that could not be assigned to any of the lines. These points are grouped together with their unassigned neighbours and memorized into intersection groups for later use. They will be used during model building as connectors between lines.

2.6.2. Model Building

Before a model can be built from detected lines, two preprocessing steps must be made. We are looking for a non-parametric system, which means that every line must be of the same length. However, in the image, multiple lines can appear in consecutive order without changing direction, and the line detection algorithm will detect it as a single long line. The first pre-processing step takes care of this problem and splits long lines so that each line in the model is of the same length.

Because every symbol representing the "turn-by-angle" action needs to be associated with a specific angle, in the second pre-processing step, the information about all of the angles between the lines needs to be retrieved from the image. The drawn lines are only an approximation of actual straight lines; therefore, to calculate an angle between them, we need to apply some rounding and cannot achieve very high precision. The result of this step is a set of unique tilt angles rounded to the closest k degrees.

After the presented pre-processing steps, a model of the detected structure can be built. This is a recursive process that connects lines with their successors by finding the edge point of the line and then looking in the set of unused lines for one that is connected to it. A line is connected to an edge point when the edge point is a neighbour of one of its points. However, an edge point might not be connected to any line. In this case, we can search the set of intersection groups to check if the edge point is connected to any of them. If that is true, it means that the line connects to an intersection and will be connected to any other line that is also connected to this intersection.

2.6.3. Sequence Generation

The last step is the translation of generated model of a structure into a sequence of symbols. First, an alphabet needs to be generated. Some of the symbols are expected to always appear in an alphabet. Those include a draw forward ('+') symbol, and if the structure contains intersections, save the current position ('[') and return to the last saved position (']') symbols since they are required to produce a branching structure. Some systems might use more than one draw-forward symbol; however, at this stage, it is not possible to deduce this, so a placeholder is used for every possible actual symbol. The last class of symbols that needs to be included in the alphabet is the turn symbols. During the pre-processing step, a set of all unique tilt angles is gathered, and a unique symbol is assigned for each value and added to the alphabet.

After creating the alphabet, a sequence can be generated. The algorithm traverses the structure model starting from the first line. For each straight line, a draw forward symbol is added to the sequence and the algorithm moves on to the next connected line. If there is a change of direction between the current and the next line before translating the next part of the model, an appropriate tilt-by-angle symbol is inserted. When the algorithm approaches an intersection, a save position symbol is inserted, and each branch is translated, returning to the previous position after finishing and moving on to the next branch. The branches are processed in the order of the smallest absolute value of the tilt angle. The return to the position symbol is not inserted at the end of the sequence since this information is redundant and does not appear in practical systems.

2.7. Grammar Inference

After parsing the input image into a sequence of symbols, an attempt at grammar inference can be made. There are many unknown variables, and the search space is large. A genetic algorithm is proposed (Algorithm 1), but first, two techniques used for space reduction need to be introduced.

2.7.1. Calculating Sequence Length at the *n*th Iteration of System

For a given alphabet $V_p = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$, the Parikh vector of a sequence w is a vector $P_w = [|S_{\sigma_1}|, |S_{\sigma_2}|, \ldots, |S_{\sigma_n}|]$ where the element $|S_{\sigma_i}|$ contains the number of appearances of symbol σ_i in this sequence. Let P_{r_i} denote a Parikh vector of the successor of the rule r_i and P_{L_i} a Parikh vector of a sequence generated by a system in its ith iteration. Then, we can define a growth matrix

$$I = \begin{bmatrix} P_{r_1} \\ P_{r_2} \\ \vdots \\ P_{r_n} \end{bmatrix}. \tag{1}$$

which allows us to calculate the Parikh vector of the sequence generated by a system in any iteration, which will be essential for calculating offspring fitness. The sequence Parikh vector can be calculated as follows:

$$P_{L_k} I^m = P_{L_{m+k}}. \tag{2}$$

Using a growth matrix, we can check if a set of rules ω can generate a sequence with a given Parikh vector P_{L_m}. To do this, we need to determine if there exists, for a given m, such a Parikh vector P_{L_0} that the Equation (2) is satisfied. If it does, then a system with rules ω and an axiom with a Parikh vector P_{L_0} can generate a sequence of the same length as the target sequence in m iterations. If such a vector does not exist, we can find a vector with the closest sequence length by solving an integer programming problem:

$$\max P_{L_0}[1], P_{L_0}[2], \ldots, P_{L_0}[n], \tag{3}$$

$$\forall i \in \{1, \ldots, n\},\ P_{L_n}[i] \leq P_w[i]$$

Algorithm 1: L-system inference

Input: *sequence*—the target sequence
Output: *bestSpecimen*—the best current solution
population ← generate initial population;
tabuList ← ∅;
i ← 0;
bestSpecimen ← ∅;
evaluate population fitness;
sequence ← remove terminal symbols from the sequence (Section 2.7.2);
while *termination condition is not satisfied* **do**
    if *i mod 5 == 0* **then**
        population ←
          replace the worst 50% of population with new random individuals
    end
    forall *ancestor in population* **do**
        selected ← select another individual from the population;
        descendant ← crossover *ancestor* with *selected*;
        mutant ← mutate *descendant*;
        fitness ← calculate *mutant* fitness;
        if *mutant has higher fitness than ancestor* **then**
            if *mutant has higher fitness than bestSpecimen* **then**
                bestSpecimen ← *mutant*
            end
            replace *ancestor* with *mutant*;
        end
        add *mutant* to *tabuList*;
    end
    $i \leftarrow i + 1$
end
return *bestSpecimen*

2.7.2. System Independence from Terminal Symbols

Let us say that S is an L-system with an alphabet containing two terminal and non-terminal symbols that generates a sequence L in the nth iteration. Knowing that a terminal character cannot be a base of a rule, we can notice that we can remove terminal symbols and analyze a more straightforward case [16]. If system S generates a sequence L and we remove terminal characters from the rules of S, it will still produce the same sequence without the terminal symbols. For example,

$$S : \{A : F, F \rightarrow F + G + F, G \rightarrow G - F - G\}$$
$$L_2 = \text{F+G+F+G-F-G+F+G+F}$$

$$\hat{S} : \{A : F, F \rightarrow FGF, G \rightarrow GFG\}$$
$$\hat{L}_2 = \text{FGFGFGFGF}$$

We can see that excluding the terminal symbols sequences L_3 and $\hat{L_3}$ are equivalent. Thanks to this property during system inference, we can first solve a simpler problem without the terminal symbols and then gradually add the terminal symbols back to the system, obtaining subsequent partial solutions. The algorithm arrives at a full solution after restoring all of the terminal symbols. This requires more searches, but each has a significantly reduced search space.

2.7.3. Genetic Algorithm

The task to be solved by this algorithm is as follows: find a set of rewriting rules that, for some axiom, allows for the generation of the target sequence in the nth iteration. The individuals are encoded using Parikh vectors of rewriting rules. Before the start of the algorithm, all of the terminal symbols are removed from the target sequence in line with the logic from Section 2.7.2. Therefore, the individuals' Parikh vectors contain only counts of non-terminal symbols.

Initial Population Generation

The exact number of rules is unknown; therefore, the population should contain individuals with different amounts. An $\frac{m}{N}$ ratio was adopted, where m represents the maximum amount of rules and N is the size of the population. For each class of systems, rules are generated randomly, with lengths ranging from 1 to a specified maximum length value k. Due to the high cost of the fitness function, the population size has to be kept low. This might cause all of the individuals to converge to the same local minima, which prevents the algorithm from exploring the whole search space. To solve this problem, the worse half of the population is replaced by new randomly generated offspring every five iterations.

Genetic Operators

This algorithm uses a typical crossover operator, with the offspring having some chance of receiving each rule from either of the parents regulated by the *crossoverRatio* parameter. It needs to be noted that only parents with an identical rule count can be bred together. The mutation operator is implemented in the form of four independent operations—SWAP (swap the successors of two rules), ADD (add a random symbol to one of the rules), REMOVE (remove a random character from one of the rules with more than one symbol), and CHANGE (change one of the symbols in one of the rules into another). If an offspring is to be modified, each operation has an equal probability of being applied. In the case of L-systems, changing a single symbol rarely leads to a better result. That is why a memory mechanism has been introduced. Every visited solution is memorized, and if the future mutation results in a previously encountered state, the operator is repeatedly applied until a new solution is obtained. The selection operator picks a random partner for every individual in the population, with candidates with higher fitness having a better chance of being selected. The algorithm terminates when a complete solution is found or the maximum iteration count has been reached.

Fitness Function

The selected fitness function executes in two phases. During the first stage (Algorithm 2), the sequence length is considered, and the fitness can reach a maximum value of 1.0, which signals that the generated sequence has reached the target length. If an individual reaches ultimate fitness for the first stage, the second phase begins, where terminal symbols are consecutively reinserted, and an exact sequence match is evaluated in each step. The fitness increases for each correctly inserted terminal character.

The first phase of the fitness calculation evaluates whether the individual can generate a sequence with the same length as the target sequence in N iterations. The closer the sequence length is to the target sequence length, the higher the fitness. The fitness of the individual for a given N can be evaluated using the method specified in Section 2.7.1 by calculating the coefficient vector:

$$I^N = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \Rrightarrow c = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} a_{11} + a_{12} + \cdots + a_{1n} \\ a_{21} + a_{22} + \cdots + a_{2n} \\ \vdots \\ a_{n1} + a_{n2} + \cdots + a_{nn} \end{bmatrix}, \tag{4}$$

and solving an integer programming problem:

$$max(c_0x_0 + c_1x_1 + \cdots + c_nx_n), \tag{5}$$

$$0 \leq c_0x_0 + c_1x_1 + \cdots + c_nx_n \leq |L|,$$
$$\forall i \in \{0, \dots, n\},\ x_i \leq |L|$$

that gives a solution in the form of $\overrightarrow{r} = \begin{bmatrix} x_0 & x_1 & \dots & x_n \end{bmatrix}$, which is a Parikh vector of the system axiom. Then the fitness of the individual is calculated as follows:

$$\text{Fitness } = 1.0 - \frac{|L| - \sum_{i=0}^{|\overrightarrow{r}|} c_i x_i}{|L|}. \tag{6}$$

However, since N is unknown, we must check multiple values. For every system, there is an iteration number M for which finding a valid axiom is no longer possible, and a value of $N = 1$ is not practically useful; therefore, we have to check values of N in the range $\langle 2; M \rangle$ and find N with the highest fitness value as the final result.

Algorithm 2: Fitness function

```
Input: candidate—the subject individual
Output: fitness—the individual fitness value
candidates ← ∅;
iteration ← 2;
while lastFitness ≠ 0 do
    currentCandidate ← individual candidate with iteration number iteration;
    coefficients ← calculate coefficients vector for currentCandidate (Equation (8));

    axiom ← calculate individual axiom based on coefficients vector coefficients;
    fitness ← calculate individual fitness (Equation (6));
    lastFitness ← fitness;
    if fitness ≠ 0 then
        candidates ← add currentCandidate to the set
    end
    iteration ← iteration + 1
end
best ← best candidate from the set candidates;
if fitness of best == 1 then
    return result of the second phase of fitness function for best;
end
return fitness of best;
```

The second phase of the fitness calculation evaluates whether the individual can correctly generate a sequence that exactly matches the target sequence. The process runs in a few iterations, and in each one, a single terminal symbol is reinserted into the target sequence. Then, we are trying to find how to insert the new character into the rules and the axiom so that the system can generate the target sequence. The general outline of a single iteration is pictured in Figure 4. The initial state is the result of the previous iteration. For the sequences to match, they must contain the same number of each symbol. Therefore, there is a finite set of combinations in which we can insert the new character into the rules so that the Parikh vectors of the sequences are equal. In step 2, the algorithm calculates all of the possible combinations by calculating the coefficients vector W:

$$W = \sum_{i=1}^{k} P_A I^{i-1} = \begin{bmatrix} W_0 & W_1 & \dots & W_n \end{bmatrix}, \tag{7}$$

and solving an integer programming problem:

$$W_0x_0 + W_1x_1 + \cdots + W_nx_n + x_A = |L|, \tag{8}$$
$$\forall i \in \{0, \ldots, n\},\ x_i \le |L|,$$
$$0 \le x_A \le |L|,$$

which gives us a set of vectors $\begin{bmatrix} x_0 & x_1 & \ldots & x_n & x_A \end{bmatrix}$, where x_n and x_A is the occurrence count of terminal symbol in the nth rule and the axiom, respectively. Because the number of combinations can be large, we can take the simplest ten for the best results. Now we know how many symbols to insert but not where. To avoid exploring all of the possibilities, since the rules' successors must appear in the target sequence, we can reduce the search space by only using the appropriate subsequences in the target sequence, which is done in step 3. From the found subsequences, in step 4, the algorithm generates a population for the GA. Since the axiom does not appear in the sequence, we only know the number of symbols to be added but not their positions; therefore, the symbols are randomly inserted. In the last step, a GA finds a system that can recreate the target sequence using the generated initial population.

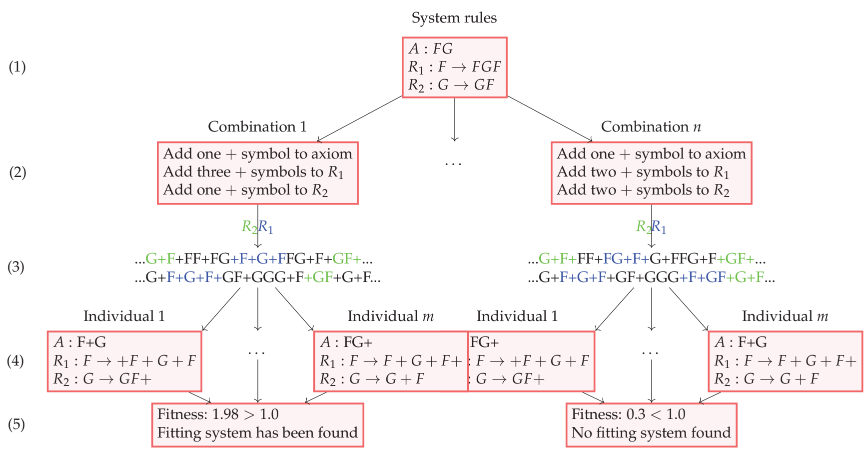


Figure 4. General outline of a single iteration of the second phase of the fitness function.

A typical crossover operator has been employed, with the descendant receiving each rule from one of the parents according to a selected ratio. A mutation operator can permute non-terminal symbols of one or more rules and the system axiom. Individuals are picked for breeding using elitist selection, with the top 10% of the population moving on to the next generation unchanged. The chosen fitness function compares the generated sequence and target sequence symbol by symbol. If a character at a given position matches, the individual receives one point. It needs to be noted that the output of the image parsing algorithm contains only a single type of non-terminal symbol; therefore, every non-terminal character receives a point for matching with it. The final fitness is a ratio between received points and total sequence length. Because the target sequences are usually very long, an optimization was applied, where only the first 100 symbols are compared, and only if those match the rest of the string is validated. When the entire sequence is correctly matched, the individual can increase their fitness value by a maximum of 1.0, relative to the simplicity of the system evaluated as:

$$Fitness = \frac{|L| - |A|}{|L|}. \tag{9}$$

The algorithm terminates when it finds a solution or reaches the maximum iteration count.

If we are looking for a single rule system, an additional optimization can be applied. Since the system axiom must start with a non-terminal symbol and a single rule cannot start with a terminal character, we can conclude that any generated sequence must begin with the rule's successor. Therefore, in step 3, instead of searching the whole string for matches, we can analyze only the first subsequence of adequate length and avoid steps 4 and 5 since there is only one matching subsequence, and we can outright check if it generates a correct sequence.

3. Results

Multiple L-systems found in the literature were selected as the example inputs to test the algorithm's efficiency. To be picked, the grammar could not have contained non-graphical symbols in its alphabet and had to generate a structure with no intersecting lines. The following systems were used: Koch Island [6,12,24], Koch Snowflake [24], Koch Curve [25], Barnsley Fern [6,13], Sierpiński Triangle [6], and Binary Tree [13]. From each selected system, an example image was generated by our plotting program and used during testing (such an approach is called grammar inference on synthetic images [26]). All of the tests were run on an AMD Ryzen 9 3900X PC with 16GB RAM. GPU acceleration and multi-threading were not used.

3.1. Grammar Inference

The tests for the provided examples succeeded in every case, including those successfully used in [13]. The initial population of 20 was used, with a mutation probability of 0.7 and a crossover ratio of 0.5. These parameters were selected during the initial experiments. The inferred systems were an exact match to the originals, with some minor notation differences that did not alter the system functionality.

The algorithm was also tested using examples used by the LGIN tool [11], and the results were compared. A solution was found in every case. However, the runtime was longer. It was to be expected since the LGIN tool uses an arithmetic approach instead of a search algorithm, which allows for faster execution but requires multiple constraints to be applied—only one or two rule systems can be inferred, with known axiom and rule count.

To compare our algorithm to the approach of Runqiang et al. [14], we used the same two examples with one and two rules, with the single-rule L-system being an equivalent of the Ex01 system from the LGIN test suite and the second system being given as:

$$\{A : X, F \to FF, X \to F[+X]F[-X]X\}.$$

The original algorithm found a solution in every run for the first system and in 66% of the runs for the second system. Meanwhile, our approach found a solution for both systems in every run. Moreover, our algorithm ran for fewer iterations than the original one (Table 1).

Table 1. Results of comparison with the algorithm from [14].

	Iteration Count of Own Algorithm				**Iteration Count of GA from [14]**		
	Minimum	**Average**	**Maximum**	σ [1]	**Minimum**	**Average**	**Maximum**
System A	1	1.7	5	1.34	1	10.8	38
System B	8	31.5	70	30.79	32	53.5	97

[1] Standard deviation.

3.2. More Complex Systems

The main advantage of our approach over the related arithmetic and heuristic algorithms described in Section 2.4 is its ability to work on systems with more than two rules. To test this capability, a system S_3 with three rewriting rules from Section 2.1 was used, and our algorithm successfully inferred the original grammar. Even though it took longer than for the simpler systems—1105 iterations in 25 min, it is a significant improvement over the mentioned algorithms that infer grammars with two rules at most.

3.3. Runtime Distribution

To test the replicability of our results, runtime distribution for the GA has been tested on systems with one and two rewriting rules. Test examples were taken from [14]. As seen in Figure 5, for a single rule system, most algorithm executions ran for a similar amount of time, around 100 ms, with very few stragglers that ran for more than 600 ms. For a system with two rules (Figure 6), we can notice similar behavior. However, here, we can see that a significant amount of runs finished quickly, meaning the initial population already contained a candidate with very high fitness. This lets us conclude that the algorithm has a low tendency to get stuck in areas of search space containing candidates with low fitness.

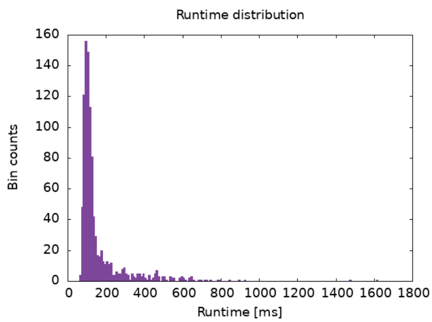


Figure 5. Runtime distribution for single rule system.

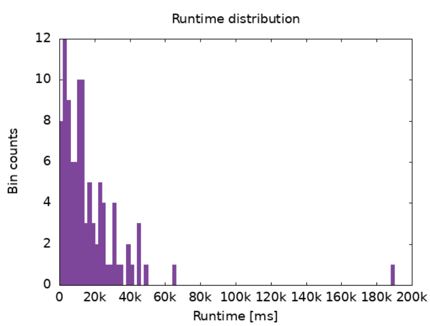


Figure 6. Runtime distribution for two rule system.

3.4. Koch Island

The effectiveness of the proposed algorithm was compared to the genetic programming method developed in [12]. It was one of the first attempts at inferring D0L-systems from a single sequence. Since then, multiple new solutions have been proposed. However, it is one of the better-documented articles, providing various performance metrics, which allow for a comprehensive comparison. The first difference in the results can be seen in the initial population generation. In the article mentioned above, it is stated that the members of the initial population of 4000 are not very good on average, with most placed around the middle of the fitness scale and the worst 12% of the population having the highest possible (the worst) fitness value. In our proposed algorithm, the population is much smaller; the tests were run using only 20 individuals. However, the generation is more effective—out of 1000 executed tests, only 32.7% of them required more than one iteration to reach a solution. Looking at the heatmap that shows the progress of hits histograms [12] (Figure 7) and changes of best and average fitness (Figure 8), we can notice that while in Koza's algorithm the progress is very steady and consistent (Figures 9 and 10), in our case, it is slower but has a tendency to take more significant leaps in quality.

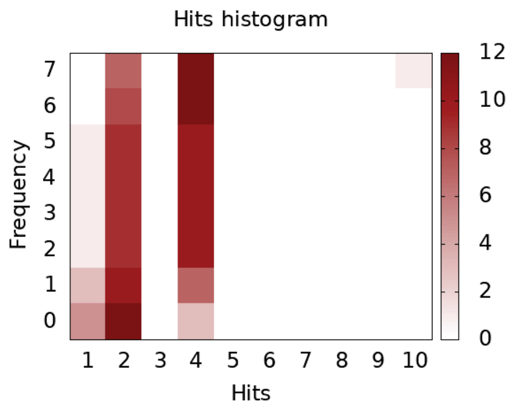


Figure 7. Hits heatmap of our algorithm.

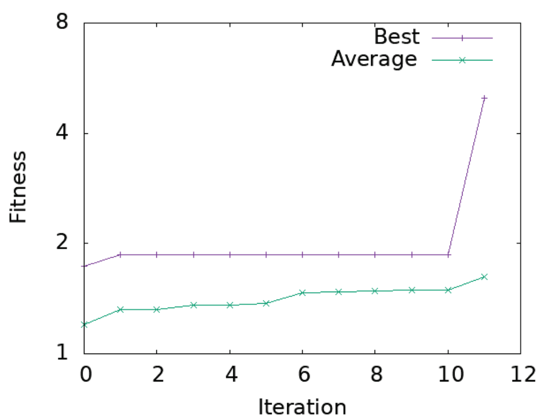


Figure 8. Fitness progression of our algorithm.

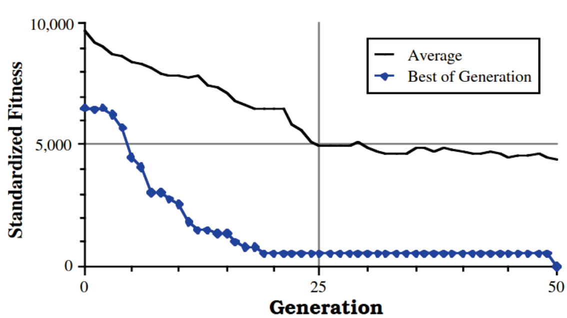


Figure 9. Fitness progression of the algorithm from [12].

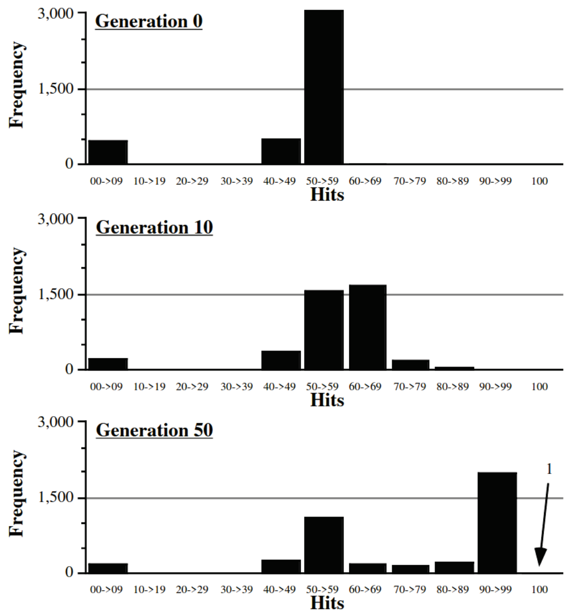


Figure 10. Hits histograms of the algorithm from [12].

3.5. Genetic Programming Using BNF Grammar

Finally, a comparison to the algorithm from the article by D. Beaumont and A. Stepney was made [15]. For this purpose, two example L-systems were used. The first one has a single rule and is given as

$$\{A \to F, F \to F[+F]F[-F]F\}. \tag{10}$$

The second one is equivalent to the Ex04Y system used for tests by the LGIN tool. In the first case, both algorithms arrived at a solution; although the compared algorithm returned multiple solutions, some of them very long or containing redundant symbols. In the second case, our algorithm managed to find a correct system every time; meanwhile, the compared algorithm achieved the same feat in only 2 in 200 test runs. The runtime was also much shorter; on average, the compared algorithm ran for several CPU-days for each test and required 891 iterations. Meanwhile, our algorithm completed the whole test suite of 30 runs in around 1 h and found a solution on average in 35 iterations.

3.6. Crossover between Individuals with Different Rule Counts

Since the selected crossover function operates only on individuals with the same rule count, two modifications have been tested. The main issue with the crossover between individuals with different rule counts is that individuals with more rules will have a larger alphabet and use symbols that are not valid for those with fewer rules. Therefore, the first modification allowed for a crossover when the second individual had the same amount or fewer rules as the main individual. This resulted in a slightly worse performance. The tests consisted of running the inference algorithm on System A from Section 3.1 for 1000 iterations. The modified crossover function resulted in an algorithm average runtime of 166.42 ms, while the original function achieved an average runtime of 161.04 ms.

The second modification further relaxed the constraints and allowed crossover between individuals with any rule count. To achieve this, a post-processing step had to be added, which, if the second individual had more rules, replaced the excess symbols with a random symbol from the smaller individual alphabet. This resulted in worse performance than the previous modification, with an average runtime of 169.84 ms. Overall, the crossover constrained to individuals with the same rule count seems to work the best, possibly because the rules already fit for this class of systems.

3.7. Comparison with Generational GA Approach

Our proposed solution uses a steady-state GA (SSGA) algorithm [27] in which only individuals that are better than their parents are inserted into the population. This approach has been compared to a generational GA (GGA) algorithm that replaces each generation's whole population. The comparison was made using the same tests as in Section 3.6. The results show a promising improvement, with the GGA algorithm achieving an average runtime of 147.5 ms compared to the 161.04 ms of the SSGA algorithm. This shows that enhancements to the breeding scheme can introduce even better performance of the inference algorithm.

4. Discussion

An algorithm for image-based grammatical inference of deterministic, context-free L-systems was proposed. The effectiveness of this approach was compared to multiple test results of comparable algorithms and tested using our examples. The results show that the algorithm performs better than existing heuristic techniques and can find solutions for the same problems as the arithmetic approaches. A significant improvement over previous methods has been made, proving that solving inference problems for systems with more than two rules is possible. However, further research is still needed. The GA's fitness function is effective but computationally costly, which implies that optimizations in this area could lead to the development of an algorithm that can solve systems with even higher rule count in a reasonable time. Further improvements to the fitness function or the encoding scheme should also be researched, studying whether fitness progress can be faster and more gradual, eliminating the frequent large jumps or decreasing the number of runs that take much longer than average. Some of the compared algorithms work faster under certain conditions, and incorporating some of their ideas into the fitness function might lead to quicker computation. Most importantly, this research shows that further advancements in single-sequence grammatical inference for D0L-systems are possible, and new solutions can provide better results, solving even more complex systems.

Author Contributions: Conceptualization, M.Ł. and O.U.; methodology, M.Ł.; software, M.Ł.; validation, M.Ł.; formal analysis, M.Ł.; investigation, M.Ł.; resources, M.Ł. and O.U.; data curation, M.Ł.; writing—original draft preparation, M.Ł. and O.U.; writing—review and editing, M.Ł. and O.U.; visualization, M.Ł.; supervision, O.U.; and project administration, O.U. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Komosinski, M.; Adamatzky, A. *Artificial Life Models in Software*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2009.
2. Langton, C. *Artificial Life: Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems*; Routledge: Oxfordshire, UK, 2019.
3. Rozenberg, G.; Salomaa, A. *The Mathematical Theory of L Systems*; Academic Press: Cambridge, MA, USA, 1980.
4. Parish, Y.I.; Müller, P. Procedural modeling of cities. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, Los Angeles, CA, USA, 12–17 August 2001; pp. 301–308.
5. Manousakis, S. Non-standard Sound Synthesis with L-systems. *Leonardo Music J.* **2009**, *19*, 85–94. [CrossRef]
6. Prusinkiewicz, P. Graphical applications of L-systems. In Proceedings of the Graphics Interface, Vancouver, BC, Canada, 26–30 May 1986; Volume 86, pp. 247–253.
7. De la Higuera, C. *Grammatical Inference: Learning Automata and Grammars*; Cambridge University Press: Cambridge, UK, 2010.
8. Whitley, D. An overview of evolutionary algorithms: Practical issues and common pitfalls. *Inf. Softw. Technol.* **2001**, *43*, 817–831. [CrossRef]

9. Abdel-Basset, M.; Abdel-Fatah, L.; Sangaiah, A.K. Metaheuristic algorithms: A comprehensive review. In *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*; Academic Press: Cambridge, MA, USA, 2018; pp. 185–231.
10. Santos, E.; Coelho, R.C. Obtaining l-systems rules from strings. In Proceedings of the 2009 3rd Southern Conference on Computational Modeling, Rio Grande, Brazil, 23–25 November 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 143–149.
11. Nakano, R.; Yamada, N. Number theory-based induction of deterministic context-free L-system grammar. In Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, Valencia, Spain, 25–28 October 2010; SCITEPRESS: Setúbal, Portugal, 2010; Volume 2, pp. 194–199.
12. Koza, J.R. Discovery of rewrite rules in Lindenmayer systems and state transition rules in cellular automata via genetic programming. In Proceedings of the Symposium on Pattern Formation (SPF-93), Claremont, CA, USA, 13 February 1993; Citeseer: Princeton, NJ, USA, 1993; pp. 1–19.
13. Eszes, T.; Botzheim, J. Applying Genetic Programming for the Inverse Lindenmayer Problem. In Proceedings of the 2021 IEEE 21st International Symposium on Computational Intelligence and Informatics (CINTI), Budapest, Hungary, 18–20 November 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 000043–000048.
14. Runqiang, B.; Chen, P.; Burrage, K.; Hanan, J.; Room, P.; Belward, J. Derivation of L-system models from measurements of biological branching structures using genetic algorithms. In *Developments in Applied Artificial Intelligence: 15th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems IEA/AIE 2002, Cairns, Australia, 17–20 June 2002*; Springer: Cham, Switzerland, 2002; pp. 514–524.
15. Beaumont, D.; Stepney, S. Grammatical Evolution of L-systems. In Proceedings of the 2009 IEEE Congress on Evolutionary Computation, Trondheim, Norway, 18–21 May 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 2446–2453.
16. Bernard, J.; McQuillan, I. New techniques for inferring L-systems using genetic algorithm. In Proceedings of the Bioinspired Optimization Methods and Their Applications: 8th International Conference, BIOMA 2018, Paris, France, 16–18 May 2018; Springer: Cham, Switzerland, 2018; pp. 13–25.
17. Bernard, J.; McQuillan, I. A fast and reliable hybrid approach for inferring L-systems. In Proceedings of the ALIFE 2018: The 2018 Conference on Artificial Life, Tokyo, Japan, 23–27 July 2018; MIT Press: Cambridge, MA, USA, 2018; pp. 444–451.
18. Bernard, J.; McQuillan, I. Techniques for inferring context-free Lindenmayer systems with genetic algorithm. *Swarm Evol. Comput.* **2021**, *64*, 100893. [CrossRef]
19. McQuillan, I.; Bernard, J.; Prusinkiewicz, P. Algorithms for inferring context-sensitive L-systems. In Proceedings of the Unconventional Computation and Natural Computation: 17th International Conference, UCNC 2018, Fontainebleau, France, 25–29 June 2018; Springer: Cham, Switzerland, 2018; pp. 117–130.
20. Bernard, J.; McQuillan, I. Inferring stochastic L-systems using a hybrid greedy algorithm. In Proceedings of the 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI), Volos, Greece, 5–7 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 600–607.
21. Bernard, J.; McQuillan, I. Stochastic L-system inference from multiple string sequence inputs. *Soft Comput.* **2022**, *27*, 6783–6798. [CrossRef]
22. Bernard, J.; McQuillan, I. Inferring Temporal Parametric L-systems Using Cartesian Genetic Programming. In Proceedings of the 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI), Baltimore, MD, USA, 9–11 November 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 580–588.
23. Lee, S.; Hyeon, D.; Park, G.; Baek, I.-j.; Kim, S.W.; Seo, S.W. Directional-DBSCAN: Parking-slot detection using a clustering method in around-view monitoring system. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 19–22 June 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 349–354.
24. Prusinkiewicz, P.; Hanan, J. *Lindenmayer Systems, Fractals, and Plants*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013; Volume 79.
25. Purnomo, K.D.; Sari, N.P.W.; Ubaidillah, F.; Agustin, I.H. The construction of the Koch curve (n, c) using L-system. In *AIP Conference Proceedings*; AIP Publishing LLC: Woodbury, NY, USA, 2019; Volume 2202, p. 020108.
26. Guo, J.; Jiang, H.; Benes, B.; Deussen, O.; Zhang, X.; Lischinski, D.; Huang, H. Inverse procedural modeling of branching structures by inferring L-systems. *ACM Trans. Graph. (TOG)* **2020**, *39*, 1–13. [CrossRef]
27. Syswerda, G. A study of reproduction in generational and steady-state genetic algorithms. In *Foundations of Genetic Algorithms*; Elsevier: Amsterdam, The Netherlands, 1991; Volume 1, pp. 94–101.

information

MDPI

Article

Minoan Cryptanalysis: Computational Approaches to Deciphering Linear A and Assessing Its Connections with Language Families from the Mediterranean and the Black Sea Areas

Aaradh Nepal [1,*] and Francesco Perono Cacciafoco [2,*]

1 School of Computer Science and Engineering (SCSE), Nanyang Technological University (NTU), Singapore 639798, Singapore
2 Department of Applied Linguistics (LNG), School of Humanities and Social Sciences (HSS), Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou 215123, China
* Correspondence: aaradh001@e.ntu.edu.sg (A.N.); francesco.perono@xjtlu.edu.cn (F.P.C.)

Abstract: During the Bronze Age, the inhabitants of regions of Crete, mainland Greece, and Cyprus inscribed their languages using, among other scripts, a writing system called Linear A. These symbols, mainly characterized by combinations of lines, have, since their discovery, remained a mystery. Not only is the corpus very small, but it is challenging to link Minoan, the language behind Linear A, to any known language. Most decipherment attempts involve using the phonetic values of Linear B, a grammatological offspring of Linear A, to 'read' Linear A. However, this yields meaningless words. Recently, novel approaches to deciphering the script have emerged which involve a computational component. In this paper, two such approaches are combined to account for the biases involved in provisionally assigning Linear B phonetic values to Linear A and to shed more light on the possible connections of Linear A with other scripts and languages from the region. Additionally, the limitations inherent in such approaches are discussed. Firstly, a feature-based similarity measure is used to compare Linear A with the Carian Alphabet and the Cypriot Syllabary. A few Linear A symbols are matched with symbols from the Carian Alphabet and the Cypriot Syllabary. Finally, using the derived phonetic values, Linear A is compared with Ancient Egyptian, Luwian, Hittite, Proto-Celtic, and Uralic using a consonantal approach. Some possible word matches are identified from each language.

Keywords: Linear A; Minoan; cryptanalysis; computational linguistics; language decipherment

Citation: Nepal, A.; Perono Cacciafoco, F. Minoan Cryptanalysis: Computational Approaches to Deciphering Linear A and Assessing Its Connections with Language Families from the Mediterranean and the Black Sea Areas. *Information* **2024**, *15*, 73. https://doi.org/10.3390/info15020073

Academic Editor: Peter Revesz

Received: 27 December 2023
Revised: 21 January 2024
Accepted: 23 January 2024
Published: 25 January 2024

1. Introduction

In 1900, Sir Arthur Evans, while excavating the Knossos Palace in Crete, unearthed clay tablets with unknown scripts on them. The writings belong to a family of scripts used in Crete, Mainland Greece, and Cyprus [1]. Among the two, which showed many similarities, the older one, Linear A, was used between 1700–1450 BCE and is yet to be deciphered [2]. The other script, Linear B, which seems to be the grammatological offspring of Linear A, was deciphered in 1952 by Michael Ventris [3]. Linear A also served as a model for another script near the end of the Bronze Age: Cypro-Minoan, which was used by the pre-Greek inhabitants of Cyprus. Cypro-Minoan, in turn, served as a model for the Cypriot syllabary, a script used by the locals to write their own dialect of Greek [1].

Sir Evans' choice of the name 'Linear' stems from the fact that both Linear A and B consist of only lines inscribed in clay [4]. Since their discovery, however, Linear A inscriptions have also been found on artefacts such as vases, jewelry, and other objects in different locations including Cyprus, mainland Greece, Turkey [5], and other Aegean islands (Kea, Kythera, Melos, and Thera) [6]. The corpus, altogether, currently consists of about 7150 signs inscribed on 1427 artefacts [1].

Most decipherment attempts begin by attributing Minoan, the language behind Linear A, to a known language family. Scholars have hypothesized links between Minoan and the Indo-European languages, the Semitic languages, the Tyrsenian languages, and the Uralic languages, among others. However, most such arguments are met with skepticism, as these attributions only yield a limited number of meaningful results [7]. Furthermore, a major ('fatal') challenge in the decipherment process is our inability to 'read' Linear A. Although there exist reasonable justifications to assign Linear B phonetic values onto Linear A for this, such an approach still produces meaningless words and has not been proven to be reliable.

Recently, novel attempts to decipher the script have emerged, which usually involve an algorithmic component. In this paper, we propose the combination of two such approaches, by Loh and Perono Cacciafoco [7] and Revesz [8], and the reasons for this are two-fold. Firstly, we aim to overcome the limitations involved in provisionally assigning Linear B phonetic values to Linear A and, secondly, we aim to shed more light on the possible connections between Linear A and other writing systems and languages from the Mediterranean and the Black Sea areas.

This paper is organized as follows. Section 2 outlines the main challenges with deciphering Linear A, along with some past attempts. Section 3 introduces and gives an overview of our proposed approach. Section 4 consists of the methodological details of this approach. Section 5 presents the results obtained when comparatively assessing the writing systems (Carian Alphabet and Cypriot Syllabary) and the languages (Ancient Egyptian, Luwian, Hittite, Proto-Celtic, and Uralic), and Section 6 discusses their implications. Section 7 concludes the entire paper and invites further work.

2. Some of the Past Decipherment Attempts

The main challenge with deciphering Linear A begins with our inability to 'read' it. Since its phonetic values are unknown, analytical attempts are likely to be unproductive. To address this, one approach, as aforementioned, has been to assign the phonetic values of Linear B to Linear A. Not only is Linear B largely derived from Linear A, but there exist visual similarities among signs in these two systems. Hence, it is reasonable to approach the decipherment of Linear A by assigning to it the phonetic values of Linear B. However, as discussed, although this allows for the 'reading' of Linear A, it has not proven to be very reliable, as it has, so far, only produced meaningless words [9].

The other challenge lies in the fact that the language behind the Linear A signs is unknown. Attempts to link so-called 'Minoan' to other known languages have remained unsatisfactory. Among the numerous hypotheses, there appear to be four main language families that scholars argue have a connection with Minoan: Indo-European, Semitic, Tyrsenian, and the Uralic language families. Vladimir Ivanov Georgiev, one of the scholars who suggests an Indo-European connection, posits that Linear A tablets, specifically the ones from Hagia Triada, encode Ancient Greek. He also believes that the other Linear A documents were transcribing Hittite–Luwian [10]. Other scholars similarly suggest an Indo-European connection. Gregory Nagy, for instance, conducted a comparative analysis of Linear A and Linear B by looking into the visual compatibilities between them, demonstrating Minoan's Indo-European-like features [11]. Similarly, Gareth A. Owens, by using phonetic values from Linear B and the Cypriot Syllabary, postulated that Minoan could be related to Sanskrit or Latin [12]. Leonard R. Palmer, another prominent scholar, suggested the possibility of Minoan being an Anatolian language linked to Luwian [13]. Palmer's theory stemmed from the two historically reconstructed invasions of Crete and Greece by the Luwians during the time when Linear A was adopted. Furthermore, statistical techniques applied to the frequency analysis of symbols and grammatological comparisons have also been considered for studying Indo-European links. Most notably, Hubert La Marle used such techniques to derive conclusions that suggest an Indo-Iranian connection for Minoan [14–17]. These theories, however, have remained controversial and unproven. Palmer's work, specifically, was criticized for relying on his subjective interpretation of the tablets, which led to varied interpretations [7]. La Marle's work, similarly, has been contested by John Grimes Younger due to questionable comparisons with various writing systems from different origins [18].

Other scholars argue for possible connections between Minoan and the Semitic language family. Cyrus H. Gordon, one of the first to propose this link, also assigned Linear B phonetic values to Linear A signs and discovered words in Linear A that appeared to be similar to words from the Semitic language family [19]. However, Gordon's approach was also met with skepticism. Critics have argued that because the matches identified were mainly vocabulary items, the reliability of the language family connection is compromised, as they could be Semitic lexical borrowings rather than examples of Linear A. Additionally, Gordon's methodology involved associating elements to several Semitic languages, such as Akkadian and Canaanite. The fact that the comparison was not carried out with one specific language led scholars to consider the Semitic hypothesis unsuccessful [20]. Jan Best's attempts at postulating Phoenician as the ancestor of Linear A were similarly countered by scholars who highlighted the lack of linguistic evidence supporting the Semitic link [21]. Eu Min et al. also investigated the plausibility of a Semitic link with their study of Linear A libation tables [20]. Although their research pointed towards a possible connection, the result was not significant enough—indeed, they produced negative results, which, in their conclusions, led to their exclusion of a Semitic option.

The third language family that has received consideration for its possible connections with Minoan is the Tyrsenian one. Helmut Rix was the scholar who theorized this language family's existence, which would include, according to him, Etruscan (spoken in central Italy between around 700 BC and 50 AD), Lemnian (spoken on the island of Lemnos around the VI Century BC), and Rhaetic (spoken in the Eastern Alps between the I millennium BC and the III century AD) [22]. Giulio Mauro Facchetti, one of the first to propose the connection, hypothesized relationships between Etruscan, Lemnian, and Minoan [23]. Facchetti suggested that Minoan could be the ancestor of the proto-Tyrsenian branch of languages from which Etruscan, Lemnian, and Rhaetic were derived. This also meant, then, according to Facchetti, that Minoan would be the ancestor of the Eteocretan branch, which he assumes is different from the other Cretan branch [24]. James Mellaart extended this work by positing a connection between Etruscan, Lemnian, and Rhaetic and pre-Indo-European Anatolian languages by studying Anatolian place names [25]. However, due to a lack of proper verification of the plausibility of connections among Etruscan, Lemnian, and Rhaetic [21], the Tyrsenian argument remains disputed.

Another approach to using Linear B phonetic values to attempt to decipher Linear A has involved an analytical interpretation of the symbols shared between the two writing systems by John Grimes Younger. Younger attempted to discover words and names by assigning the Linear B phonetic values to Linear A. He was able to recognize a few possible Linear A toponyms with a comparison with Mycenaean Greek along with a positional and frequency study of place names in Linear B (and Linear A) tablets [26]. However, this comparative examination between Linear A and Linear B, although logical and well-grounded, did not yield decisive results, unfortunately.

A recent decipherment approach [7] proposes comparing Linear A with other language families according to their grammatological elements through a 'brute force attack'. The method, originating from cryptanalysis, involves assigning Linear B phonetic values to Linear A and then comparing the consonant clusters of Linear A with the consonant clusters of other languages from the region. A set of dictionaries of various languages stored in spreadsheet files is used as the input for a Python program which performs this comparison. The 'consonantal' approach is declared to be effective, because consonant clusters are, presumably, more stable and consequently allow for the easier analysis of the morphological parts of a language.

Peter Z. Revesz [8] proposed another approach, which involves comparing Linear A to other writing systems visually, by using a feature-based similarity measure. This novel algorithm was employed to develop a new phonetic grid for Linear A, which was then used to generate a Minoan–Uralic dictionary. According to Revesz, he was able to translate twenty-eight Linear A inscriptions. More recently, Revesz also pointed to archaeogenetic evidence that suggests Minoans may have originated in the Danube Basin and the Western Black Sea coastal area [27], which could further suggest a Minoan–Uralic connection, given earlier and newer publications that argue that the Proto-Uralic people once lived in the Black Sea area [28–31].

Another important challenge in deciphering Linear A, however, is simply that the corpus is very small. There are currently about 7150 Linear A signs inscribed on 1427 artefacts [1]. This is in contrast with the larger corpus of Linear B, comprising about 30,000 signs at the time it was deciphered [26].

3. Proposed Approach

First, Linear A is compared with other writing systems in the region using the feature-based similarity measure proposed by Revesz [8]. Second, the phonetic values of those writing systems that are visually similar are assigned to Linear A. These two steps are to ensure that the potential limitations involved with provisionally assigning Linear B phonetic values to Linear A are accounted for. Finally, Linear A is compared with other languages using the consonantal approach/'brute force attack' proposed by Loh and Perono Cacciafoco [7].

3.1. Writing Systems Compared with Linear A

The writing systems that will be compared visually to Linear A include the Cypriot Syllabary and the Carian Alphabet. Some scholars have previously assigned phonetic values from the Cypriot Syllabary onto Linear A for its analysis. Most notably, Owens [12] used phonetic values from the Cypriot Syllabary and Linear B to hypothesize possible links between Minoan and the Indo-European language family. Since Linear A was used as a model for Cypro-Minoan which, in turn, was used to model the Cypriot Syllabary, this paper aims to further explore the relationships between the two.

The Carian Alphabet, similarly, is argued to be linked to the Cretan Scripts' family which, among other writing systems, includes Linear A and Linear B [32]. Revesz discusses the possible connections between Old Hungarian and the Carian Alphabet using a feature-based similarity measure and postulates that the Carian Alphabet is an ancestor of Old Hungarian. Therefore, a possible link between the Carian Alphabet and Linear A is considered.

3.2. Languages Compared with Linear A

Adopting the 'brute-force attack' proposed by [7], the languages/language clusters compared using the consonantal approach include Ancient Egyptian, Luwian, Hittite, Proto-Celtic, and Uralic, which belong, largely, to three language families: Indo-European, Afro-Asiatic, and Uralic. Since a considerable number of decipherment attempts suggest the possibility of Minoan belonging to the Indo-European language family, this paper aims to explore this further, with Luwian and Hittite. With Ancient Egyptian, it aims to investigate possible connections between the Minoans and the Egyptians. Sir Arthur Evans posited that the interaction between Crete and Egypt began during the third millennium BC [33]. Archeological evidence also strongly suggests a link between the two. Thus, we propose a further analysis of the possible connections between their languages. We also include a comparison with Proto-Celtic, which, although it does not have an apparent relation to Linear A, allows us to leverage the unbiased and universally applicable 'brute-force' nature of the consonantal approach. Finally, we also aim to further explore the Minoan–Uralic connection mentioned above.

4. Methods

4.1. Deriving the Phonetic Values

We use the feature-based similarity measure proposed by Revesz [8] to derive a new phonetic grid for Linear A. It has the following components:

1. Similarity Function

To compute the similarity between any two symbols, we let S $= \{s_1, s_2, s_3, \ldots, s_n\}$ be a set of n symbols, F $= \{f_1, f_2, f_3, \ldots, f_m\}$ be a set of m elementary features, and T $:(S, F) \rightarrow \{0,1\}$ be a function that maps a symbol–feature pair with either 0 or 1, depending on whether that symbol has that feature. Then,

$$\mathrm{sim}\left(s_i, s_j\right) = \sum_{k=1,\mathrm{T}(s_i,f_k)=\mathrm{T}(s_j,f_k)}^{m} w_k$$

where w_i is a weight function that maps any feature i to a real number (the weight assigned to that feature).

2. Elementary Feature Set

The elementary feature set describes the feature of each symbol using a set of descriptors. Each feature corresponds to elements found in the symbols. Table 1 shows the elementary feature set used for this paper, which is based on the one developed by Revesz [8].

Table 1. Elementary features with their corresponding weights.

Elementary Feature	Weight
The symbol contains some curved lines	0.01
The symbol encloses some regions	0.01
The symbol has a slanted straight line	0.01
The symbol contains parallel lines	0.02
The symbol contains crossing lines	0.02
The symbol's top is a wedge	0.12
The symbol's bottom is a wedge	1.00
The symbol's right side is a wedge	0.33
The symbol contains a stem, that is, a straight vertical line that runs down the middle	0.03
The symbol's bottom has two legs	0.06
The symbol's bottom has three legs	0.09
The symbol contains a hair, a small line extending from an enclosed space	0.04
The symbol contains two triangles	0.33

3. Weight of Each Feature

In Revesz [8], the weight of all features is 1. However, in this study, a different set of weights for each feature is used. The weight of each feature is the inverse of its frequency of occurrence across all symbols in Linear A. In other words, a feature that exists in most symbols will have a lower weight compared to a feature that only exists in some. This means that sharing a rarely occurring feature is given more importance than sharing a commonly occurring one.

Table 1 illustrates the weight of each feature in the elementary set based on a frequency analysis performed for all features across all standard simple signs in Linear A (A001–A371) from GORILA (the Linear A corpus by Louis Godart and Jean-Pierre Olivier).

With the elementary feature set, a feature map is first computed for each symbol in Linear A, the Carian Alphabet, and the Cypriot Syllabary. The feature map demonstrates the existence of specific elementary features in the symbol. Each symbol in Linear A is then compared with each symbol in the Carian Alphabet and the Cypriot Syllabary to derive their similarity scores. This expectedly results in a large output. Hence, after the comparison, some criteria are necessary to keep only those symbol matches that are strongly correlated. In this paper, the following criteria are employed:

- The threshold for the similarity value given by the similarity function is set to 2.05. This means only two symbols whose similarity values are above or at 2.05 are considered potential matches;
- If there are multiple matches with the same similarity value, the tie is broken by manually analyzing the symbols;
- Matches that meet the threshold, but are visually dissimilar upon manual analysis are also not considered. Such a case could arise due to the limited number of features considered or because of the interdependence among features in the elementary feature set. For instance, for a symbol to contain a hair it must also enclose some 'region'.

Additionally, since the phonetic grid is derived via visual comparison, allographs in these writing systems are important for consideration. The Carian Alphabet, specifically, is composed of a few of them. For instance, B, Ʋ, Ɲ, and Ч all have the same phonetic value. In this paper, all variants of symbols in the Carian Alphabet specified in the Unicode Standard are examined independently for comparison. In the case of both the Cypriot Syllabary and Linear A, only the standard signs are used.

4.2. Consonant Cluster Comparison

After assigning the new phonetic values, the comparison between the languages is performed using a Python program developed by a research team led by Francesco Perono Cacciafoco and Colin Loh at Nanyang Technological University (NTU), Singapore [34]. The program works by using two CSV files as the input. The first CSV file is a Linear A master list with three columns: 'Source' (the artefact that contains the Linear A word), 'New Format' (the Linear A word with phonetic values derived from the feature-based similarity measure), and 'Linear A' (the Linear A word with the vowels removed). The second CSV file contains a single column with all the dictionary words of the language being compared. The program then removes vowels from the words of all the dictionary words of the language being compared and compares each of them with words from the Linear A master list. It finally produces a list of exact matches found between Linear A consonant clusters and the consonant clusters of the language which is compared. These matches are finally compared manually, in turn, to dictionary entries in the selected language, to see whether a meaning can be assigned to them. If a meaningful entry is found, this is cross-referenced with the original Linear A tablet and a judgment is made as to whether it allows us to 'read' the tablet itself, or part of it.

5. Results

5.1. Phonetic Values for Linear A

Using the feature-based similarity measure, each symbol in Linear A is compared with every symbol in the Cypriot Syllabary and the Carian Alphabet, to derive the possible phonetic values of Linear A. Table 2 shows a sub-set of these comparisons, filtered using the criteria outlined in the Methodology. The last column indicates the writing system that the matched sign is assumed to belong to ('CS' denotes the Cypriot Syllabary, 'CA' denotes the Carian Alphabet). The phonetic values are transcribed using Latin/Roman letters. For the Carian Alphabet, they are based on the transcription system posited by Ignacio J. Adiego [35].

It is important to note that Linear A, being—plausibly—a syllabary, is likely not composed of pure consonants, unlike the Carian Alphabet. This poses a challenge with using the Carian Alphabet to derive the phonetic values of Linear A. Revesz [8] proposes that if a Linear A symbol corresponds to a Carian Alphabet symbol with the phonetic value /C/ (a consonant), then the Linear A symbol for some vowels will have a phonetic value of /CV/ (consonant/vowel). For instance, the Linear A sign Ɨ, which could match with the Carian Alphabet sign I, would have a phonetic value of /L/+/V/. Revesz then derives the value for this /V/ by searching for the "appropriate word to describe the meaning of the Linear A symbol" [8] in Uralic, Finno-Ugric, and Ugric vocabulary lists.

Table 2. Feature-based similarity scores for a sub-set of symbol pairs.

Linear A Sign		Matched CS/CA Sign		Phonetic Value	Similarity Score	Assumed Origin
AB01		TA		TA	2.07	CS
AB02		LO		LO	2.07	CS
AB03		PA		PA	2.07	CS
AB07		UUU2		Y	2.07	CA
AB08		UUU3		Y	2.07	CA
AB09		SE		SE	2.07	CS
AB11		B		B	2.07	CA
AB13		NE		NE	2.06	CS
AB17		RA		RA	2.07	CS
AB20		TI		TI	2.05	CS
AB22		U		U	2.07	CS
AB24		LD		L	2.05	CA
AB31		U		U	2.07	CA
AB34		D		D	2.07	CA
AB37		A		A	2.06	CA
AB39		E		E	2.06	CS
AB44		A		A	2.07	CS
AB46		X		C	2.06	CA
AB48		NG		NG	2.05	CA
AB50		S		S	2.06	CA
AB51		SU		SU	2.07	CS
AB54		UU		W	2.06	CA
AB55		E2		E	2.06	CA
AB59		R		R	2.06	CA
AB65		D2		D	2.07	CA
AB70		JA		JA	2.07	CS
AB77		Q		QU	2.07	CA
A302		RI		RI	2.07	CS
A304		TI		TI	2.07	CS
A306		XE		XE	2.06	CS
A309A		O		O	2.07	CA
A311		TT2		CH	2.07	CA
A312		L		L	2.05	CA
A314		MB		MB	2.07	CA
A318		G2		G	2.07	CA
A319		LD		L	2.07	CA
A325		T		T	2.07	CA
A326		NN		N	2.07	CA
A330		KU		KU	2.06	CS
A339		LE		LE	2.07	CS
A349		ST2		Z	2.07	CA
A351		PE		PE	2.07	CS
A355		KI		KI	2.07	CS

In this paper, no particular vowel was concatenated with the pure consonants, as it was assumed that they could have any value. Since this comparison of Linear A with other languages involves a consonantal approach, the lack of specific vowels does not entirely render the phonetic values obsolete.

5.2. Comparing Linear A with Other Writing Systems and Related Languages

With the phonetic values derived in Table 2, Linear A was compared with Ancient Egyptian, Luwian, Hittite, Proto-Celtic, and Uralic by using the consonantal approach proposed by Loh and Perono Cacciafoco [7]. The results derived from the operation of the Python program developed by the two scholars for this task, highlighted in the Methods section, are presented in Tables 3–7. Since the Python program yields a lot of matches for each of the languages, the results presented have been filtered manually, to ensure that only matches with a high likelihood of plausibility are kept for consideration.

Table 3. Python program results for Ancient Egyptian.

Matched Consonants	Linear A Cluster	Egyptian Word	Linear A Source	Meaning ' | ' Separates Different Meanings '?' Indicates that the Meaning Is Uncertain
nr	NE-RA	iner	HT10A	shell of an egg | gravel, stone
p	PA-[],]-PA	ipA	KN32b, KH 91	to make to fly, to fly | house, dwelling, harem
pr	PA-RA-[ ],]-PA-R	aper	ZA006b, KH 79 + 89	to be equipped, to be provided with, furnished (of a house) | a boat equipped with everything necessary and a crew
r	RA	ArA	ZA009	to go up, to embark in a boat, to bring, to be high
rp	R-PA	irp	HT104	wine | wine plant, vine | to rot, to decay, to ferment
rr	RI-R	irr	HT30	deaf (?) | grapes, grape seeds | a wine jar
ry	RI-Y	ary	HT28a, HT28b	he who goes up | light, fiery one | the name of a Dekan | a kind of fish | breeze, wind
yS	Y-SE	AyS	HT132, HT81, HT93a, HT85a	truce

Table 4. Python program results for Luwian.

Matched Consonants	Linear A Cluster	Luwian Word	Linear A Source	Meaning ' | ' Separates Different Meanings '?' Indicates that the Meaning Is Uncertain
ll	LO-LO	lalai	KE Wc 2b	take
p	PA-[],]-PA	pa	KN32b, KH91	protect (?)
r	RA	ura	ZA009	great
ry	RI-Y	ariya	HT28a, HT28b	raise | check, restrain (?)
t	TA	ta	HT86a, Wa 1031	step | arrive
tn	TA-NE	taini	HT95a, HT95b	of oil, oily
w	W	wi	HT98a, Wc 3019, HT97a	see | appear (?)

Table 5. Python program results for Hittite.

Matched Consonants	Linear A Cluster	Hittite Word	Linear A Source	Meaning ' | ' Separates Different Meanings '?' Indicates that the Meaning Is Uncertain
ll	LO-LO	lulu	KE Wc 2b	evenness, steadiness, stability, security
lr	L-R	luri	HT10B	loss, shortfall, decimations | loss of standing, comedown, disgrace, degradation
p	PA-[],]-PA	apa	KN32b, KH 91	that (one) | he, she, it | the one in question | thy, thine, your(s)
pr	PA-RA-[ ],]-PA-R	puri	ZA006b, KH 79 +89	lip | rim, edge, border
prl	PA-R-L	parala	HT122a, HT94b	something of wood used on sacrificed cattle, nom
prt	PA-R-TA	parta	PH31a	side, siding, partition
ps	PA-SE	pus	HT18, HT27b	diminish, fade, be eclipsed | be small, act petty, be pusillanimous
r	RA	ara	ZA009	belonging (or: proper) to one's own social group, communally accepted or acceptable, congruent with social order
rp	R-PA	arp	HT104	bad luck, setback, misfortune

Table 6. Python program results for Proto-Celtic.

Matched Consonants	Linear A Cluster	Proto-Celtic Form	Linear A Source	Meaning ' | ' Separates Different Meanings '?' Indicates that the Meaning Is Uncertain
lr	L-R	*liro	HT10B	sea (?)
n	[]-NE	*ne	ZA020	not
nr	NE-RA	*nero	HT10A	hero (?)
rr	RI-R, []-RI-R	*eriro	HT30	eagle
ry	RI-Y	*aryo	HT28a, HT28b	free man
sny	SU-NE-Y	*sniyo	HT19	spin, weave
t	TI, TI-[, TI-[]	*eti	HT28a, KH90, Wc 3015b	yet, still, but | beyond | also
tn	TA-NE	*tini	HT95a, HT95b	melt
wy	W-Y	*way	HT94b	woe, oh, alas
y	Y	*yo	HT85b, We 1023/ Wd 1024	which
yr	Y-R-[]	*yaro	ZA009	chicken, hen

Table 7. Python program results for Uralic.

Matched Consonants	Linear A Cluster	Uralic Form	Linear A Source	Meaning ' | ' Separates Different Meanings '?' Indicates that the Meaning Is Uncertain
n	[ ]-NE	une	ZA020	sleep, dream
nr	NE-RA	nure	HT10A	to press

Table 7. *Cont.*

Matched Consonants	Linear A Cluster	Uralic Form	Linear A Source	Meaning '∣' Separates Different Meanings '?' Indicates that the Meaning Is Uncertain
pr	PA-RA-[ ]	para	ZA006b, KH 79 + 89	good
ps	PA-SE	pese	HT18, HT27b	to wash (head?)
r	RA	ora	ZA009	awl∣squirrel
rp	R-PA	orpa	HT104	melt
sr	SU-[ ]-RA	sira	ZA018a	a k. of relative
t	TA	ta	HT86a, Wa 1031	this, that
t	TI, TI-[, TI-[]	tE	HT28a, KH 90, Wc 3015b	you
tn	TA-NE	tana	HT95a, HT95b	birch bark
w	W, W-[]	owe	HT98a, Wc 3019	door
wl	W-L	wElE	HT38	to understand

6. Discussion

Although the results could suggest possible links between Linear A and Ancient Egyptian, Luwian, Hittite, Proto-Celtic, and Uralic, the matches found are insufficient to yield conclusive evidence of any connection. For each compared language, the matches appear sparse and spread across multiple tablets. Additionally, the number of matches across the languages are similar, with certain Linear A words matching with words in all the languages, suggesting that the result is coincidental rather than indicative of concrete links.

The limited number of matches could be due to the phonetic values used for the comparison. The feature-based similarity measure, with the parameters utilized in this paper, was only successful in producing 43 matches for comparing Linear A with other languages. In contrast, since Linear A and B potentially share 92 similar signs, naturally the phonetic grid based on Linear B includes more signs. There are several reasons for the derived phonetic grid being small. Firstly, it could simply indicate a lack of concrete links between the scripts. Secondly, while the feature-based similarity measure allows for an analysis of different writing systems, it is not without its limitations. The method depends highly on the elementary feature set, and since we only had a few features, it is plausible to assume that certain important features may have been missed during the analysis. Additionally, a small feature set also increases the probability of finding multiple matches for any symbol with the same similarity scores, and breaking the tie becomes a challenging decision. In Revesz [8], for instance, the tie is broken by choosing the symbol that is earlier in the standard ordering of symbols.

It is important to note, additionally, that the limited number of matches could simply indicate a lack of connections between the languages. Most connection hypotheses, as discussed previously, have shown to be unsuccessful due to reasonable justifications. Considerations such as the temporal and spatial relations of the writing systems and languages are undeniably important factors. For instance, a Minoan and Luwian or Hittite (Indo-European) link could be considered unlikely due to temporal gaps, if the emergence of the Minoan civilization is believed to predate the arrival of Indo-European speakers to Anatolia.

The Combined Approach

Our approach aimed to leverage different characteristics of two computational methods of decipherment, in the effort to interpret Linear A. The feature-based similarity measure, for instance, has been considered effective for visual comparisons of writing systems. In [36], Barla et al. performed a feature analysis of Indus Valley- and Dravidian-connected scripts. They propose a novel elementary feature set consisting of six additional features on top of the one employed in this paper and generate heat maps for the different writing systems. Comparing their approach to our approach in this paper, we chose to use the same feature set as Revesz [8]. However, this choice is arbitrary and evidently influences the results obtained post analysis. Selecting a good elementary feature set is not straightforward and requires experimentation and further analysis. This suggests that although the approach seemingly aims to provide an objective way to compare writing systems, it is still subjective to an extent. For this paper, however, the approach has allowed us to account for biases that arise while assigning Linear B phonetic values to Linear A, which is also inherent in the so-called 'consonantal approach'. It is important to note, however, that visually similar symbols may not necessarily share the same phonetic values [37].

After the derivation of the phonetic values, the consonantal approach has enabled us to attempt a statistical analysis of a new rendition of Linear A with other languages from the region, resulting in a few matches. In [7], Colin Loh and Francesco Perono Cacciafoco outlined preliminary results using this consonantal approach with Linear B phonetic values, and they found matches across Hittite and Luwian. They posited that the Linear A cluster "PA-RE", from the document HT4 of *GORILA*'s volume 1, could be a possible match with "PARI" from the Luwian dictionary, which represents "forth", or "away". Due to the limited number of phonetic values derived in this paper, it is difficult to assess the effectiveness of the consonantal approach for the comparative analysis of languages. A limitation inherent in such an approach is the loss of information resulting from the removal of vowels. The matches may just be loanwords or purely coincidence. Furthermore, filtering the large number of matches generated by the Python program is not arbitrary and requires further consideration and study. The approach's effectiveness in performing a 'brute-force' analysis, however, is evident.

Overall, the combined approach is effective in a cross-language and cross-script analysis, albeit with some limitations inherent in the two approaches that have been combined. It is also worth noting that there are possible limitations with the combination as well. Due to the dependence of the consonantal approach on the feature-based similarity measure, it may be difficult to determine the plausibility of links between languages using this approach. Obtaining a low number of matches, for instance, could indicate a lack of connection between the languages, compared to using the consonantal approach or when the writing systems are compared visually. Hence, the combined approach necessitates a stepwise assessment of the results. If there are low matches when comparing the writing systems visually, the decision of whether these writing systems are appropriate for use with the consonantal approach must be made first.

7. Conclusions

Among the numerous attempts to decipher Linear A, some recent approaches involve a computational component. This paper aimed to combine two such novel methods to firstly account for the biases inherent in provisionally assigning Linear B phonetic values to Linear A and, secondly, to shed more light on the possible connections between Linear A and other writing systems and languages of the Mediterranean and the Black Sea areas. This paper also aimed to highlight some limitations inherent to such approaches. The first step in the combined approach involved a feature-based similarity measure to visually compare writing systems and the second involved using a consonantal approach to compare different languages. Although the writing system still remains largely undeciphered, by employing the combined approach some Linear A signs were found to be similar to signs from both the Cypriot Syllabary and the Carian Alphabet. Applying the phonetic values of those similar signs to Linear A and comparing it with Ancient Egyptian, Luwian, Hittite, Proto-Celtic, and Uralic resulted in a few word matches between the languages. Although these could suggest possible connections, they are not significantly conclusive, due to the limited number of matches. Along with some limitations inherent to the combined approach, the small corpus still poses a major challenge in deciphering Linear A. However, our approach can be applied and used to compare any known writing system and language possibly connected to Linear A, removing our dependence on assigning Linear B phonetic values to Linear A and allowing for an unbiased analysis. Further research could investigate the use of the combined approach with other scripts and languages.

Author Contributions: Conceptualization, A.N. and F.P.C.; methodology, F.P.C. and A.N.; software, F.P.C. and A.N.; validation, F.P.C.; formal analysis, A.N.; investigation, F.P.C. and A.N.; resources, F.P.C.; data curation, F.P.C.; writing—original draft preparation, A.N. and F.P.C.; writing—review and editing, F.P.C. and A.N.; supervision, F.P.C.; project administration, F.P.C.; funding acquisition, F.P.C. All authors have read and agreed to the published version of the manuscript.

Funding: The tools and research developed for this paper were funded by a Singapore Ministry of Education (MOE) AcRF Tier 1 Research Grant (Grant Number 2017-T1-002-193—Principal Investigator: Dr Francesco Perono Cacciafoco), "Giving Voice to the Minoan People: The Decipherment of Linear A", 2018–2021.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data collected and used in this study are safely stored in physical data discs and in an online (private) data repository. They can be made freely and promptly available to any scholar interested in it upon request by email to the authors. The Python program mentioned in our study can be found at the following GitHub page: https://github.com/L-Colin/Linear-A-decipherment-programme (accessed on 21 November 2023).

Acknowledgments: We would like to acknowledge Colin Loh (National University of Singapore—NUS, Singapore) for all his work and help in the development of the software (https://github.com/L-Colin/Linear-A-decipherment-programme, accessed on 21 November 2023) used and implemented in the project which is the origin of this paper. This project was supported by Nanyang Technological University (NTU), Singapore, under the URECA Research Programme.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Davis, B. Introduction to the Aegean Pre-Alphabetic Scripts. *Kubaba* **2010**, *43*, 38–61.
2. Cadogan, G. *Palaces of Minoan Crete*; Barrie and Jenkins: London, UK, 1976; ISBN 978-0-214-20079-3.
3. Chadwick, J. *The Decipherment of Linear B*, 2nd ed.; Cambridge University Press: Cambridge, UK, 2014; ISBN 978-1-107-69176-6.
4. Robinson, A.; Robinson, A. *Writing and Script: A Very Short Introduction*; Very Short Introductions; Oxford University Press: Oxford, NY, USA, 2009; ISBN 978-0-19-956778-2.
5. Mycenean Artifacts Found in Bodrum. Available online: https://www.hurriyetdailynews.com/mycenean-artifacts-found-in-bodrum--74114 (accessed on 21 November 2023).
6. Perono Cacciafoco, F. Linear A and Minoan: The Riddle of Unknown Origins. In Proceedings of the LMS Fieldwork and Language Analysis Group (FLAG) Meeting, School of Humanities (SoH), Nanyang Technological University (NTU), Singapore, 10 June 2014.
7. Loh, J.S.C.; Perono Cacciafoco, F. A New Approach to the Decipherment of Linear A: Coding to Decipher Linear A, Stage 2 (Cryptanalysis and Language Deciphering: A 'Brute Force Attack' on an Undeciphered Writing System). In *Proceedings of the Grapholinguistics in the 21st Century—2020, Part II*; Fluxus: Brest, France, 2020; pp. 927–943. ISBN 978-2-9570549-7-8.
8. Revesz, P. Establishing the West-Ugric Language Family with Minoan, Hattic and Hungarian by a Decipherment of Linear A. *WSEAS Trans. Inf. Sci. Appl.* **2017**, *14*, 306–335.
9. Godart, L. Du Lineaire A Au Lineaire B. In *Aux Origines de L'hellénisme: La Crète et la Grèce. Hommage à Henri van Effenterre*; Histoire Ancienne et Médiévale; Publications de la Sorbonne: Paris, France, 1984; pp. 121–128.
10. Georgiev, V.I. Les Deux Langues des Inscriptions Crétoises En Linéaire A. *Linguist. Balk.* **1963**, *7*, 1–104.
11. Nagy, G. Greek-like Elements in Linear A. *Greek Rom. Byzantine Stud.* **1963**, *4*, 181–211.
12. Owens, G. The Structure of the Minoan Language. *J. Indo Eur. Stud.* **1999**, *27*, 15–56.
13. Palmer, L.R. *Mycenaeans and Minoans: Aegean Prehistory in the Light of the Linear B Tablets*, 1st ed.; Alfred A. Knopf: New York, NY, USA, 1962.

14. La Marle, H. *Linéaire A: La Première Écriture Syllabique de Crète: Essai de Lecture*; Librairie Orientaliste Paul Geuthner: Paris, France, 1996; ISBN 978-2-7053-3641-7.
15. La Marle, H. *Linéaire A, la Première Écriture Syllabique de Crète: Éléments de Grammaire*; Librairie Orientaliste Paul Geuthner: Paris, France, 1997; ISBN 978-2-7053-3642-4.
16. La Marle, H. *Linéaire A, la Première Écriture Syllabique de Crète: L'histoire et la Vie de Crète Minoenne: Textes Commentés*; Librairie Orientaliste Paul Geuthner: Paris, France, 1998; ISBN 978-2-7053-3643-1.
17. La Marle, H. *Linéaire A: La Première Écriture Syllabique de Crète. Signes Rares, Textes Brefs, Substitutions 4*; Librairie Orientaliste Paul Geuthner: Paris, France, 1999; ISBN 978-2-7053-3644-8.
18. Younger, J.G. Linear A: Critique of Decipherments by Hubert La Marle and Kjell Aartun. Available online: https://people.ku.edu/~jyounger/LinearA/LaMarleAartun.html (accessed on 7 December 2023).
19. Gordon, C.H. *Forgotten Scripts: Their Ongoing Discovery and Decipherment*; Revised and Enlarged edition; Dorset Press: Dorchester, UK, 1987; ISBN 978-0-88029-170-5.
20. Eu Min, N.C.; Perono Cacciafoco, F.; Cavallaro, F.P. Linear A Libation Tables: A Semitic Connection Explored. *Ann. Univ. Craiova Ser. Philol. Linguist. Analele Univ. Din Craiova Ser. Ştiințe Filol. Linguist.* **2019**, *41*, 51–63.
21. Perono Cacciafoco, F. Linear A and Minoan: Some New Old Questions. *Ann. Univ. Craiova Ser. Philol. Linguist. Analele Univ. Din Craiova Ser. Ştiințe Filol. Linguist.* **2017**, *39*, 154–170.
22. Rix, H. *Rätisch und Etruskisch*; Institut für Sprachwissenschaft der Universität Innsbruck: Innsbruck, Austria, 1998; ISBN 978-3-85124-670-4.
23. Facchetti, G.M.; Negri, M. *Creta Minoica: Sulle Tracce Delle più Antiche Scritture d'Europa*; Archivum romanicum Biblioteca dell'Archivum romanicum; Ser. 2, Linguistica; Olschki: Firenze, Italy, 2003; ISBN 978-88-222-5291-3.
24. Facchetti, G.M. *Appunti di Morfologia Etrusca: Con Un'appendice Sulla Questione Delle Affinità Genetiche Dell'etrusco*; Biblioteca dell' "Archivum Romanicum"; L.S. Olschki: Firenze, Italy, 2002; ISBN 978-88-222-5138-1.
25. Mellaart, J. *The Neolithic of the Near East*, 1st ed.; Scribner: New York, NY, USA, 1975; ISBN 978-0-684-14483-2.
26. Younger, J.G. Linear A Texts & Inscriptions in Phonetic Transcription & Commentary. Available online: https://people.ku.edu/~jyounger/LinearA/ (accessed on 7 December 2023).
27. Revesz, P.Z. Data Mining Autosomal Archaeogenetic Data to Determine Minoan Origins. In Proceedings of the 25th International Database Engineering & Applications Symposium, New York, NY, USA, 7 September 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 46–55.
28. Krantz, G.S. *Geographical Development of European Languages*; P. Lang: Berlin, Germany, 1988; ISBN 978-0-8204-0800-2.
29. Makkay, J. An Archeologist Speculates on the Origin of the Finno-Ugrians. *Mank. Q.* **2003**, *43*, 235–272. [CrossRef]
30. Wiik, K. The Uralic and Finno-Ugric Phonetic Substratum in Proto-Germanic. *Linguist. Ural.* **1997**, *33*, 258–280. [CrossRef]
31. Revesz, P. Inscription on a Naxian-Style Sphinx Statue From Potaissa Deciphered as a Poem in Dactylic Meter. *Mediterr. Archaeol. Archaeom. MAA* **2023**, *23*, 1–15.
32. Revesz, P.Z. The Cretan Script Family Includes the Carian Alphabet. *MATEC Web Conf.* **2017**, *125*, 5019. [CrossRef]
33. Marinatos, N. The Indebtedness of Minoan Religion to Egyptian Solar Religion: Was Sir Arthur Evans Right? *J. Anc. Egypt. Interconnect.* **2010**, *1*, 22–28. [CrossRef]
34. Loh, J.S.C. Linear-A-Decipherment-Programme. Available online: https://github.com/L-Colin/Linear-A-decipherment-programme (accessed on 21 November 2023).
35. Adiego Lajara, I.-J. *The Carian Language*; Handbook of Oriental studies; Section one, The Near and Middle East; Brill: Leiden, The Netherlands; Boston, MA, USA, 2007; ISBN 978-90-04-15281-6.
36. Barla, S.S.; Alamuru, S.S.S.; Revesz, P.Z. Feature Analysis of Indus Valley and Dravidian Language Scripts with Similarity Matrices. In Proceedings of the International Database Engineered Applications Symposium, Budapest, Hungary, 22 August 2022; ACM: Budapest, Hungary, 2022; pp. 63–69.
37. Yao, Y.; Perono Cacciafoco, F.; Cavallaro, F.P. The Minoan Challenge: An External Analysis Approach to the Linear A Decipherment. *Ann. Univ. Craiova Ser. Philol. Linguist.* **2022**, *44*, 456–475. [CrossRef]

information

Article

A Proposed Translation of an Altai Mountain Inscription Presumed to Be from the 7th Century BC

Peter Z. Revesz [1],* and Géza Varga [2]

1 School of Computing, College of Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA
2 Írástörténeti Kutató Intézet, 1121 Budapest, Hungary; vargageza2@gmail.com
* Correspondence: revesz@cse.unl.edu; Tel.: +1-402-421-6990

Abstract: The purpose of this study is to examine an Old Hungarian inscription that was recently found in the Altai mountain and was claimed to be over 2600 years old, which would make it the oldest extant example of the Old Hungarian script. A careful observation of the Altai script and a comparison with other Old Hungarian inscriptions was made, during which several errors were discovered in the interpretation of the Old Hungarian signs. After correcting for these errors that were apparently introduced by mixing up the inscription with underlying engravings of animal images, a new sequence of Old Hungarian signs was obtained and translated into a new text. The context of the text indicates that the inscription is considerably more recent and is unlikely to be earlier than the 19th century.

Keywords: Altai inscription; decipherment; inscription; Old Hungarian script; Orkhon script; translation

Citation: Revesz, P.Z.; Varga, G. A Proposed Translation of an Altai Mountain Inscription Presumed to Be from the 7th Century BC. *Information* **2022**, *13*, 243. https://doi.org/10.3390/info13050243

Academic Editor: Francesco Fontanella

Received: 14 February 2022
Accepted: 7 May 2022
Published: 10 May 2022

1. Introduction

A puzzling, unique inscription from the Altai Mountain was recently presented by Karžaubaj Sartkožauly, who is a member of Kazakhstan academy of sciences, in a monograph on the Orkhon script [1]. According to Sartkožauly, the inscription was made in the 7th century BC.

Sartkožauly [1] also noticed that the inscription has similarities with the Old Hungarian script (Hungarian: *székely írás* or *rovásírás*), which was used by Hungarians before the adoption of the Latin alphabet in the Middle Ages [2,3]. Sartkožauly's book [1] remained unnoticed in Hungary until Lajos Máthé brought it to the attention of the second author. Subsequently, the second author alerted the first author and asked for his help in the translation of the inscription. The second author already correctly identified a few words, and the first author identified the still-missing words and completed the translation. Both authors were intrigued by the Altai inscription and the possibility that it may be the oldest extant example of the Old Hungarian script.

Although Sartkožauly already presented a translation of the inscription, we show that it has several errors. One of the problems is that the inscription is partly written over the engraved images of several animals. As we show, there are several instances where Sartkožauly mixed up the actual inscription and the engraving of the animals. Correcting these mistakes gives us a different sequence of Old Hungarian signs. Moreover, this enables us to give a better, alternative translation of the Altai inscription.

The rest of this paper is organized as follows. Section 2 describes the materials and methods. Section 3 describes the main results of the paper, including our identification of a new Old Hungarian signs sequence read off from the Altai inscription (Section 3.1) and a transliteration and translation of the Altai inscription (Section 3.2). Section 4 discusses the Altai inscription and finds a new date range for its creation. Section 5 presents an alternative transliteration and translation of the inscription. Finally, Section 6 gives some conclusions and directions for further work.

2. Materials and Methods

The main method of our research was a careful examination of the original photo of the Altai inscription in [1]. It was discovered that the inscription was overlayed on the engraved images of some animals. The engravings are usually fainter than the inscription, but there are cases where the lines are indistinguishable. This causes several problems in the precise identification of the Old Hungarian signs that were intended by the scribe. We could correct several of the earlier mistakes made by Sartkožauly [1] and obtain a new sequence of Old Hungarian signs.

Next, we transcribed the new sequence of Old Hungarian signs. The transcription was complicated by the presence of ligatures, which are combinations of letters. We also looked for various Old Hungarian alphabets from various centuries to identify the one that contained signs that have similar forms to the one in the Altai inscription.

Finally, we translated the inscription first into Hungarian and then into English. The etymology of the Hungarian words was considered in finding an improved date range for this Altai inscription.

3. Results

This section may be divided by subheadings. It should provide a concise and precise description of the experimental results, their interpretation, as well as the experimental conclusions that can be drawn.

3.1. A Reexamination of the Old Hungarian Signs

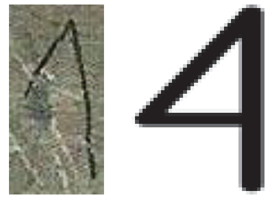
Sartkožauly [1] gave a drawing of the inscription. Figure 1a is a modification of that drawing by enhancing it with different colors for the inscription itself and the underlying animal drawings. This distinction is important because it influences the interpretation of the signs that are thought to belong to the inscription. In fact, we do not completely agree with Sartkožauly's identification of what belongs to the inscription versus the underlying drawings.

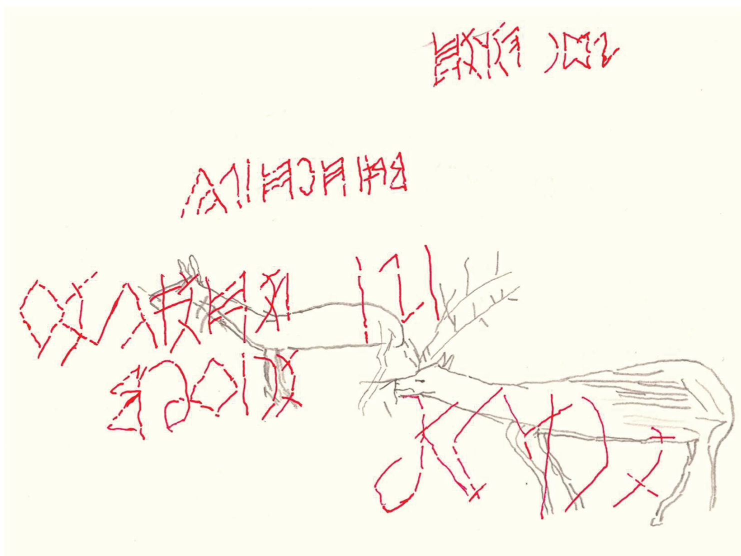
In fact, our examination of the photo of the Altai inscription led us to a different identification of the sequence of Old Hungarian signs as shown in Figure 1b. We believe these differences are due to different interpretations of what little line segments belong to the inscription itself and what line segments belong to the engravings of the animal figures on the rock surface where the inscription was found.

In addition, there are also some cracks on the rock that may cause problems in the correct discernment of the Old Hungarian signs that belong to the inscription. Below we list the most important differences that we identified.

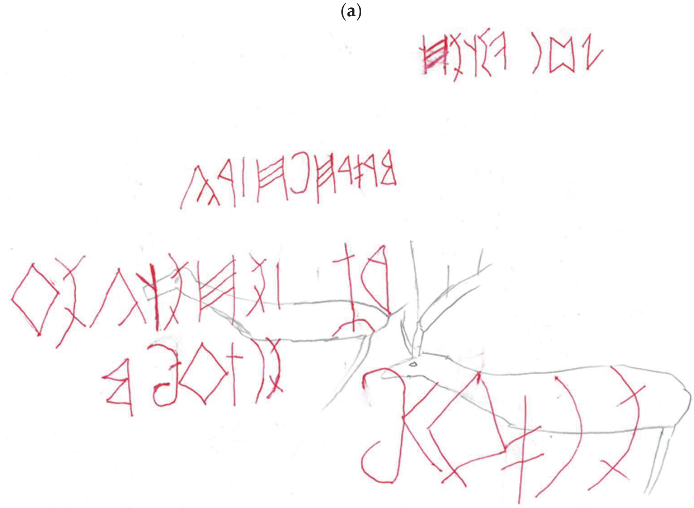
In the second row, we identified the fourth and the ninth signs from right to be the Old Hungarian sign denoting the vowel *a*. Here is an enlargement of the fourth sign from the right in the second row of the photo in [1] next to the Old Hungarian *a* sign:

Similarly, let us consider now the ninth sign from the right in the second row next to the Old Hungarian *a* sign:





(a)



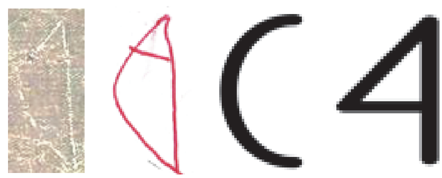
(b)

Figure 1. Two drawings of the Altai inscription based on a photo from [1]. (**a**) the first author's redrawing of Sartkožauly's drawing in [1]. The improved drawing shows the inscription in red color and part of the animal drawings in the background in black color; (**b**) an alternative drawing of the same inscription by the first author. This alternative drawing follows closer the original inscription shown in the photo.

In both cases, there are line segments which look deliberate and belong to the Old Hungarian sign. In the second case, it is not clear why some of the lines have been blackened, but this feature also appears in some other signs of the Altai inscription. As can be seen in Figure 1a [1], left out some of the line segments that form the little triangle in these signs.

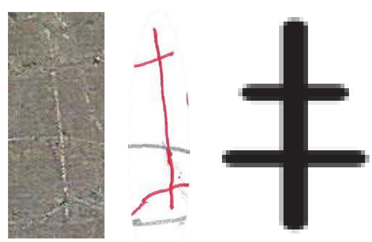
Sartkožauly [1] overlooked that the inscription contains some ligatures, which are combinations of two or more signs. Ligatures often save some space and are common in Old Hungarian inscriptions. In the Altai inscription, we also find a few examples of ligatures. For example, in the third row of the inscription, we believe that the first sign on the right is a ligature of the Old Hungarian *n* and *a* signs.

Below we show an enhanced image of the first sign from the right in the third row, our drawing of it, and the Old Hungarian *n* sign written with a mirror symmetry and an *a* sign:



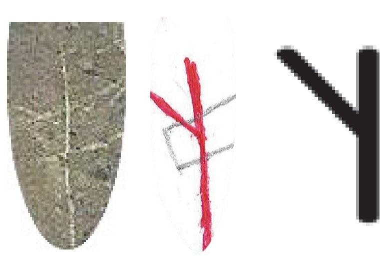
The comparison shows that this sign may be a ligature of the Old Hungarian signs. The Old Hungarian *n* sign  is likely mirrored to make the combination with the  sign easier and to save more space. The ligature is read as *na*.

The next sign in the third row is not a straight vertical line as [1] assumes, because it also has additional overlooked details, as can be seen in the following enhanced photo:



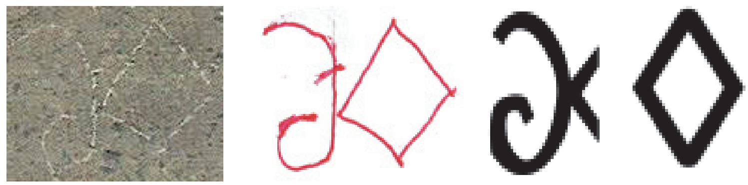
Here we need to be careful to ignore the engravings that depict part of the back and the belly of a deer. The lines to be ignored are shown in black in our drawing.

The seventh sign from the right in the third row is an Old Hungarian *t* sign:



In the enhanced photo, the Old Hungarian  is clearly visible. In addition, there are two parallel lines that belong to the head of one of the engraved deers. These lines do not belong to the Old Hungarian inscription and should be ignored. Unfortunately, Sartkožauly considered these lines part of the inscription and obtained an Old Hungarian  sign in this place.

In the fourth line, there are additional missing details in Sartkožauly's drawing. The first sign from the left is missing its top half, the diamond sign misses on side, and in the second word, which is written with smaller signs, the third sign from the right misses a small horizontal crossing line segment. These can also be verified by a careful observation of the original photo of the Altai inscription. In addition, the following ligature was also overlooked:



This ligature is a combination of the Old Hungarian *k* sign  and *ő* sign . Together they can be read as *kő*.

3.2. Transliteration and Translation of the Altai Inscription

We agree with Sartkožauly that the Altai inscription needs to be read from right-to-left. Most Old Hungarian inscriptions known from Hungary and the Carpathian Basin are also read from right-to-left. On the other hand, a left-to-right presentation would make the translation hard to read. Hence Table 1 presents each row of the Altai inscription in red based on our drawing, its Old Hungarian left-to-right transliteration in black, and below the Old Hungarian signs a Latin alphabet transliteration of the Old Hungarian letters. The Latin alphabet is extended by some accent marks.

Table 1. The Altai inscription and its row-by-row transliterations into Old Hungarian and Latin.

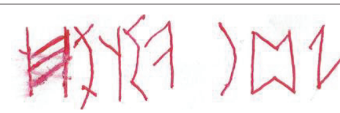
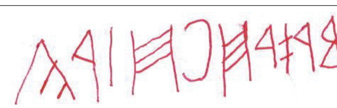
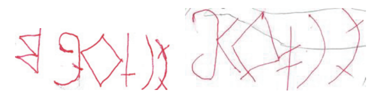
Row	Script	Inscription
1	Altai, right-to-left	
	Old Hungarian, left-to-right	
	Latin	k u n p é t e r
2	Altai, right-to-left	
	Old Hungarian, left-to-right	
	Latin	m a gy a r o r sz á g

Table 1. *Cont.*

Row	Script	Inscription
3	Altai, right-to-left	
	Old Hungarian, left-to-right	
	Latin	n agy sz e r e t l e k
4	Altai, right-to-left	
	Old Hungarian, left-to-right	
	Latin	e n i k ő e n i k ő m

There are certain peculiarities in the Latin transliteration that we made in order to obtain meaningful words. In particular, we believe that the scribe was not using the standard Old Hungarian signs but mixed up some of the similar looking signs. In particular, the scribe mixed up the Old Hungarian letters for *r* and *z*, which are the following, respectively:

r z

Similarly, the scribe also mixed up the Old Hungarian letters for *g* and *l*, which are the following, respectively:

g l

We had to assume these two interchanges to obtain meaningful Hungarian words. We give a row-by-row translation of the Altai inscription in items (1–4) below.

1. The first row of the inscription starts with the name *Kun Peter*. Interestingly, the family name *Kun* is written first, and the given name *Peter* is written second. This order agrees with the Hungarian word order. In addition, *Peter* is a common given name in Hungary, and *Kun,* meaning 'Cuman', is also a common family name. In fact, Hungarian *Kunság* is the name of a region of Hungary that was settled by Cumans in the 13th century. Many people in that region consider themselves to be descendants of the Cumans and took *Kun* as a family name in later centuries.
2. The second row of the inscription contains the Hungarian word *Magyarország,* which means 'Hungary.' The Hungarians' neighbors apparently confused the Hungarians with the Huns and the Onogurs, who occupied present day Hungary before the Magyars and allied peoples arrived in the 9th century. For example, German speakers in Austria and Germany call the country *Ungarn*.
3. The third row of the inscription contains the Hungarian word *nagy*, which means 'big' or 'much', and the Hungarian word *szeretlek,* which means 'I love you.' Hence the two words together express the sentence 'I much love you.'

4. The fourth row of the inscription contains the Hungarian word *Enikő*, which is a common woman's name, and its conjugation *Enikőm*, where the -m suffix is a first-person possessive marker. Hence the meaning of *Enikő, Enikőm* is 'Enikő, my Enikő'. The name *Enikő* is said to derive from Hungarian *enéh* meaning 'young hind (female deer)' [4]. It is perhaps for this reason that we see two deers drawn next to these words in the inscription.

In summary, the inscription can be translated into Hungarian as follows.

Kun Péter, Magyarország: Nagy szeretlek, Enikő, Enikőm.

This means the following in English:

Enikő, my Enikő, I much love you.—Peter Kun, Hungary

Therefore, the Altai inscription is a message of love from a gentleman named *Peter Kun* to *Enikő*, who is his beloved woman.

4. The Inscription's Implications for the Development of the Old Hungarian Script

The Old Hungarian alphabet is thought to be a descendant of the Orkhon Turkic alphabet [3]. An early example of an Old Hungarian inscription from the Altai Mountain would support the theory of an Orkhon Turkic origin of the Old Hungarian alphabet.

On the other hand, the first author argued that the Old Hungarian alphabet may be a descendant of the Carian alphabet, which in turn may be a descendant of the Minoan Linear A script [5]. The second author has also proposed that the Old Hungarian script had a pictogram or hieroglyph script-like origin in the Carpathian Basin even earlier [6]. These two views do not exclude each other because there is growing evidence based on archaeogenetics [7,8] and art motif comparisons [9] that the Minoans came from the Danube Basin to the Aegean islands in the early Bronze Age. Hence both authors were skeptical about an Asian or in particular an Orkhon Turkic origin of the Old Hungarian script. However, we were intrigued by the reported find and undertook the research described in this paper.

During the translation, we noticed that the Old Hungarian signs of Altai inscription reflected not the earliest known forms, as one would expect from a 2600 years old inscription, but from later centuries.

Luckily, the date of the inscription can be narrowed down a pure linguistic reason. The reason is that the name *Enikő* was created by Mihály Vörösmarty (1800–1855), a Hungarian poet [4]. Hence the Altai inscription was carved in the latter half of the 19th century or later. Already in the 19th century, Hungarians had a strong interest in exploring the area because of presumed cultural connections with some people living near the Altai Mountains. In fact, a well-known Hungarian scientific expedition to the Altai Mountain was led by Count Jenő Zichy in 1895 [10].

5. An Alternative Translation of the Inscription

Sartkožauly [1] has given a transliteration of the letters of the Old Hungarian inscription based on an interpretation of the drawing as shown in Figure 1a. His transliteration, which is only the substitution of the Old Hungarian letters by Latin letters, is the following from the topmost line to the bottom-most line:

Line 1: kunpétez
Line 2: magy sz zcz sz
Line 3: sz ksz sz eze gügek
Line 4: enü ? o en sz kom

As can be seen, the transliteration is different from ours because some letters are faintly written over some underlying drawings of animals. Therefore, they have ambiguous interpretations. In fact, Sartkožauly [1] has used a question mark at some point in the last line to indicate that at that point he did not find a clearly readable letter that he could transliterate with confidence. Sartkožauly [1] could not give an actual translation.

Sartkožauly's drawing and transliteration was the starting point of our translation of the inscription. Initially, we tried to make only minimal changes to both his drawing and transliteration as shown in Figure 2. In particular, Figure 2 follows Figure 1a at the right end of the third line. In the third line, the three letters are supposedly the following from right to left: sz ksz.

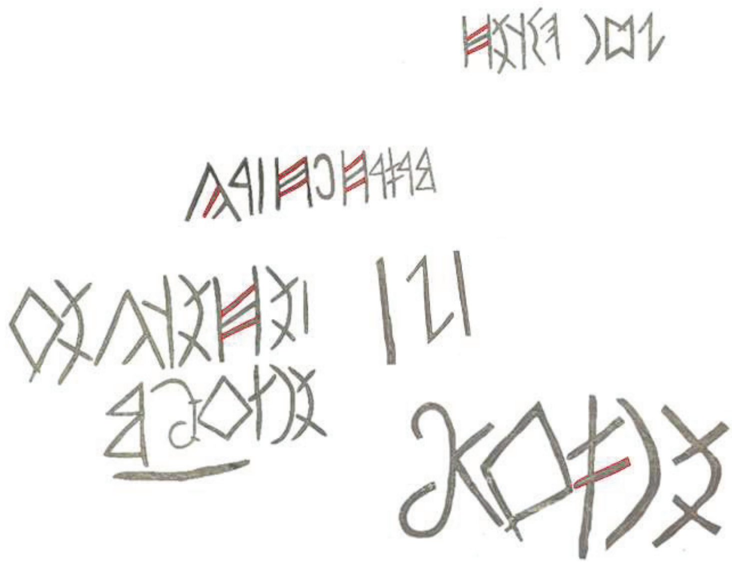


Figure 2. An alternative drawing of inscription. Here the red lines are those that seem extra to the letters that are apparently needed for a meaningful reading of the inscription.

It is possible to translate this as the word *szex* because while the traditional Old Hungarian alphabet does not have an *x* letter, the convention is to render the letter *x* as a combination of Hungarian *k* and *sz*, which is equivalent to English *k* and *s*, respectively. Of course, then the translation would change to the following:

Enikő, my Enikő. I love sex [with you].—Peter Kun, Hungary

This alternative has some problems. First, the Hungarian word *szex* is a borrowed word that was first used only in 1958 according to Zaicz [11]. Hence this would require a late 20th century origin of the inscription. Second, the use of the two Old Hungarian *k* letters would be inconsistent. Usually, the Old Hungarian diamond-shaped letter *k* is used with front vowels, while the Old Hungarian Z-shaped letter *k* is used with back vowels. The latter can also be used to express the frequent syllable *ak* because the vowel *a* can be omitted.

The Altai inscription adheres to this custom because the Z-shaped letter is used in the word *Kun,* which contains the back vowel *u*, while the diamond-shaped letter is used in the words *szeretlek*, *Enikő*, and *Enikőm*, all of which contain front vowels. However, this custom would be broken by the use of the Z-shaped letter in writing the word *szex,* which has a front vowel. For these reasons and also the visual analysis that we presented in Section 3, this alternative reading seems less plausible. We present this here mainly to show the evolution of our thinking.

In Section 3, we mentioned that the scribe mixed up some letters because of misremembering some details. Instead, it is possible to imagine that the scribe remembered correctly the letters and at first wrote them correctly as shown by the black lines in Figure 2. Then he became embarrassed by the inscription and deliberately added the red lines shown in Figure 2. The scribe may have thought that the addition of the red lines makes the original inscription unreadable. The hypothesis of deliberately adding extra lines can be used with either of our translations. Because it is only an explanation for the apparently mixed-up letters, it can be accepted or rejected without changing the meaning of the translation. The reason this hypothesis may be attractive is that whenever the scribe mixed up letters, the intended letter, whether *r* or *g*, always has fewer lines than the actual written letter, whether *z* or *l*.

6. Conclusions

We gave a new, correct transliteration and translation of the Old Hungarian inscription from the Altai Mountain that was reported by Sartkožauly [1]. We also redated the inscription to the 19th century or later based on a linguistic argument. Although the inscription did not prove to be as ancient as originally assumed, it still provides an amazing and valuable cultural connection between the peoples near the Altai Mountain and Hungarians in Central Europe.

Author Contributions: Conceptualization, methodology, and investigation, P.Z.R. and G.V.; draft notes in Hungarian, drawing of Figure 2, G.V.; extension, writing in English, drawing of Figure 1a,b, and dating using linguistic argument, P.Z.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sartkožauly, K. *Complete Atlas of the Orkhon Monuments*; Almaty Samga Press: Almaty, Kazakhstan, 2019; Volume 3.
2. Wikipedia, Old Hungarian Script. Available online: https://en.wikipedia.org/wiki/OldHungarianscript (accessed on 8 December 2021).
3. Benkő, E.; Sándor, K.; Vásáry, I. *A Székely Írás Emlékei*; Bölcsészettudományi Kutatóközpont: Budapest, Hungary, 2021.
4. Wikipedia, Enikő. Available online: https://en.wikipedia.org/wiki/Enikő (accessed on 13 January 2022).
5. Revesz, P.Z. Establishing the West-Ugric language family with Minoan, Hattic and Hungarian by a decipherment of Linear A. *WSEAS Trans. Inf. Sci. Appl.* **2017**, *14*, 306–335.
6. Varga, G. *Magyar Hieroglif Írás*; Írástörténeti Kutatóintézet: Budapest, Hungary, 2017.
7. Revesz, P.Z. Minoan archaeogenetic data mining reveals Danube Basin and western Black Sea littoral origin. *Int. J. Biol. Biomed. Eng.* **2019**, *13*, 108–120.
8. Revesz, P.Z. Data mining autosomal archaeogenetic data to determine Minoan origins. In Proceedings of the 25th International Database Engineering and Applications Symposium, Montreal, QC, Canada, 14–16 July 2021. [CrossRef]
9. Revesz, P.Z. Art motif similarity measure analysis: Fertile Crescent, Old European, Scythian and Hungarian elements in Minoan culture. *WSEAS Trans. Math.* **2019**, *18*, 264–287.
10. Maracskó, A. Hungarian Orientalism and the Zichy Expeditions. Master's Thesis, Central European University, Budapest, Hungary, 2014.
11. Zaicz, G. (Ed.) *Etimológiai Szótár: A Magyar Szavak és Toldalékok Eredete*; Tinta Press: Budapest, Hungary, 2006.

information

Article

Decipherment Challenges Due to Tamga and Letter Mix-Ups in an Old Hungarian Runic Inscription from the Altai Mountains

Peter Z. Revesz

School of Computing, College of Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588, USA; revesz@cse.unl.edu; Tel.: +1-402-421-6990

Abstract: An Old Hungarian Runic inscription from the Altai Mountains with 40 signs has posed some special challenges for decipherment due to several letter mix-ups and the use of a tamga sign, which is the first reported use of a tamga within this type of script. This paper gives a complete and correct translation and draws some lessons that can be learned about decipherment. It introduces sign similarity matrices as a method of detecting accidental misspellings and shows that sign similarity matrices can be efficiently computed. It also explains the importance of simultaneously achieving the three criteria for a valid decipherment: correct signs, syntax, and semantics.

Keywords: decipherment; error correction; inscription; Old Hungarian Runic script; sign; similarity matrix; tamga

Citation: Revesz, P.Z. Decipherment Challenges Due to Tamga and Letter Mix-Ups in an Old Hungarian Runic Inscription from the Altai Mountains. *Information* **2022**, *13*, 422. https://doi.org/10.3390/info13090422

Academic Editors: Francesco Fontanella and Arkaitz Zubiaga

Received: 2 June 2022
Accepted: 6 September 2022
Published: 7 September 2022

1. Introduction

The history of paleography never saw a case when a scribe came alive and told the would-be decipherers that they were wrong. Embarrassingly, something like that happened to us after we published [1] our decipherment of a puzzling Old Hungarian Runic (Hungarian: *székely írás* [2], *székely-magyar rovás* or *rovásírás* [3]) inscription that was previously described by Karžaubaj Sartkožauly, a member of the Kazakhstan Academy of Sciences, in a three-volume monograph on the Orkhon script [4], where he presumed the inscription to be from the seventh century BC.

The Hungarian name is alternatively translated as Székely-Hungarian Rovash [5] or Old Hungarian [3]. The term 'Old Hungarian' may be confusing because it is used by some scholars to refer to the Latin alphabet-based script that was used from the 10th to the 16th century in Hungary. The extended name 'Old Hungarian Runic' inscription is clearer because 'runic' means 'relating to runes (magic marks or letters, especially the letters of an ancient alphabet cut into stone or wood in the past)' according to the Cambridge Dictionary. Hence, English 'runes' and Hungarian 'rovás' both refer to the same means of writing.

Our journal article generated much public interest in Hungary. It was also featured in a popular YouTube video on Hungarian history. Eventually, one viewer left a comment, which can be translated into English as follows: 'I carved this inscription into the rock at the Mongolian Altai Mountains in the Bayan-Ölgii Province, near the upper flow of the Uygariin River in June 2000'.

Finding the scribe allowed a unique opportunity to check our translation and ask some details about the circumstances of the inscription. This was important because the inscription consists of 40 signs, and, out of those 40 signs, a sequence of three signs remained uncertain. The goal of this paper is to describe the problem with that sequence of signs in our earlier paper and to propose a complete and correct translation. As part of the analysis, the paper introduces the use of similarity matrices to check for misspellings and draws some general lessons for decipherers of ancient inscriptions.

This paper is organized as follows: Section 2 gives some background information on the Old Hungarian Runic script; Section 3 describes the data source and data curation;

Section 4 gives a transliteration of the signs. A sign similarity matrix is used to show that the inscription contains some common misspellings; Section 5 reviews earlier decipherment proposals and evaluates them according to the criteria of correct signs, syntax, and semantics; Section 6 gives the correct identification of the disputed sign group as a tamga; Section 7 presents some lessons learned about decipherment; lastly, Section 8 presents some conclusions and future work.

2. Background on the Old Hungarian Script

The Old Hungarian Runic script (Hungarian: *székely írás* or *rovásírás*) has been the subject of many studies [2,3,5]. An early book about the subject by Sebestyén [6] popularized the idea that the Old Hungarian Runic script is a descendant of the Old Turkic Orkhon. This origin theory developed even before the Minoan civilization, and its scripts were discovered on the island of Crete by Sir Arthur Evans. During a cryptographic study of the Minoan Linear A script, the author discovered its relationship with the Old Hungarian Runic script. More precisely, it was shown that the Minoan Linear A script is an ancestor of the Carian script, which is the ancestor of the Old Hungarian Runic script [7].

As the above history suggests, the Old Hungarian Runic script has developed considerably from its earliest form to the present. Table 1 shows its current state that is also part of the Unicode standard. Even the two-letter Hungarian transliterations denote single phonemes [8]. There is only one remarkable exception to the pure alphabetic nature of the script. K^1 and K^2 are used with front and back vowels, respectively. This feature may hark back to an era when these were syllabic signs denoted KE and KA, respectively.

Table 1. The Old Hungarian Runic script with its Hungarian transliteration.

Old Hungarian Runic Sign	Hungarian Transliteration	Old Hungarian Runic Sign	Hungarian Transliteration
𐲀	A	𐲗	LY
𐲁	Á	𐲘	M
𐲂	B	𐲙	N
𐲄	C	𐲚	NY
𐲆	CS	𐲛, 𐲜	O, Ó
𐲇	D	𐲞, 𐲟	Ö, Ő
𐲉	E	𐲠	P
𐲋	É	𐲢	R
𐲌	F	𐲤	S
𐲍	G	𐲥	SZ
𐲎	GY	𐲦	T
𐲏	H	𐲨	TY

Table 1. *Cont.*

𐲐	I	𐲪	U
𐲒	J	𐲬	Ü
𐲓	K[1] (front-vowel)	𐲮	V
𐲔	K[2] (back vowel)	𐲯	Z
𐲖	L	𐲰	ZS

3. Data Sources and Data Curation

Karžaubaj Sartkožauly's drawing had some minor inaccuracies. He included a photograph in his work. A new drawing based on that photo is shown in Figure 1. The drawing shows that some parts of the inscription are unclear because of the drawings of the deer and some cracks in the rock.

Figure 1. The author's redrawing of the inscription based on the photograph in Sartkožauly [4].

Figure 2 shows an enhanced drawing with red highlighting of those elements that clearly belong to the inscription and labeling the various groups of signs.

Those who are familiar with the Old Hungarian Runic script can easily recognize many of the signs. Hence, one can suspect that some more elements also belong to the Old Hungarian signs in sign group (d) in the middle of the drawing, where unfortunately the tail of the female deer on the left and the antler of the stag on the right interfere with the Old Hungarian signs. This interference results in at least two different interpretations as shown in Figure 3.

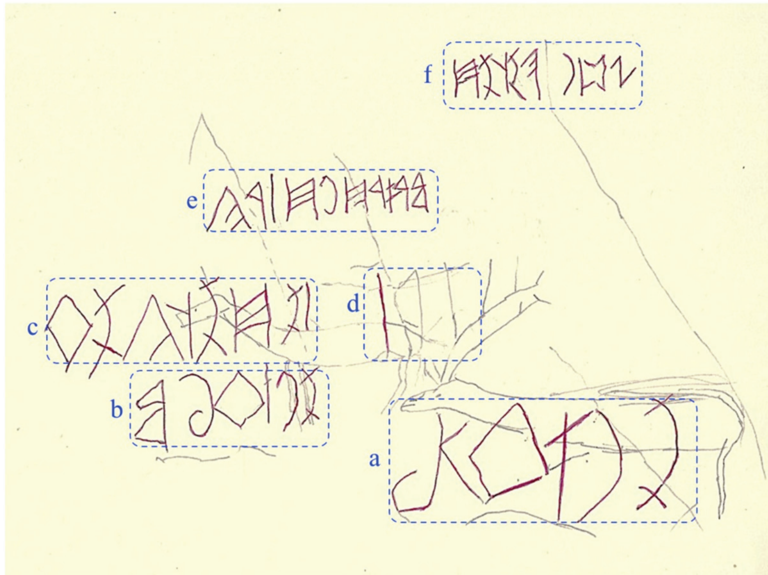


Figure 2. An enhanced drawing of the inscription with red highlighting of those elements that undisputedly belong to the inscription. The six sign groups are also labeled (a–f).

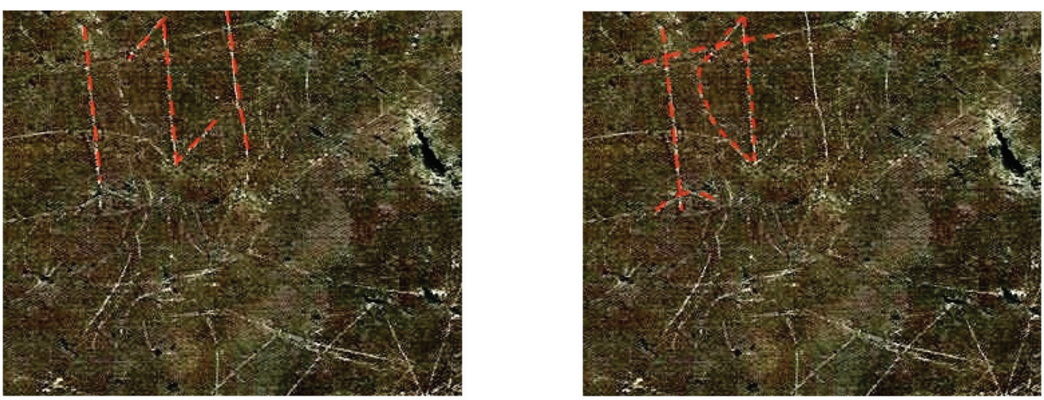


Figure 3. Two interpretations of sign group (d) in the middle of the photograph.

The first interpretation of sign group (d) leads to the following sign sequence:

The second interpretation, which contains an Old Hungarian A and N ligature, leads to the following sign sequence:

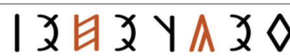
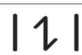
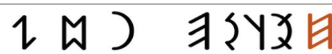
While the N sign normally looks as shown above, a scribe could reverse the direction for the sake of a ligature. The scribe also used an Ő-K[1] ligature in sign group (b). The difference in these two interpretations is a subtle matter of interpreting a few faintly scratched lines. What the first interpretation considers the Old Hungarian S, the second interpretation considers part of the antler of the stag.

The most logical way to handle ambiguities is to proceed further in the decipherment because the context of the other words can help to choose among the choices. Hence, for now, let us simply refer to these two sign group options as (d1) and (d2), respectively.

4. Transliteration and Correction of the Signs

Since Old Hungarian inscriptions are written from right to left, we first convert the sign groups into a left-to-right order as shown in Table 2. Next, we also attempted a transliteration to find the meaning of the words.

Table 2. The Altai Mountain inscription with incorrect signs highlighted in brown.

Row	Inscription	Transliteration	Meaning
a		E N I K^{1} Ő	Enikő
b		E N I K^{1} Ő M	my Enikő
c		SZ E Z E T G E K^{1}	
d1		SZ K^{2} SZ	
d2		N A GY	great
e		M A GY A Z O Z SZ Á L	
f		K^{2} U N P É T E Z	Kun

It is apparent to Hungarian language speakers that some words do not make sense, although they are close to common Hungarian words. For example, in sign group (f), the intended name PÉTER can be easily recognized instead of the nonsense string PÉTEZ. This suggests that the scribe made a spelling mistake. In particular, the scribe wrote the Old Hungarian Z sign instead of the Old Hungarian R sign.


R


Z

These two signs look similar; hence, it is understandable that such a mistake can be made by someone who is not completely familiar with the script. The Altai Mountain inscription uses a form of Z that has two legs. In many texts, including this paper, the following slightly different form of Z is used:


Z

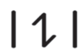
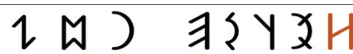
Apparently, the scribe also mixed up the Old Hungarian signs G and L in the words MAGYARORSZÁG and SZERETLEK. These two signs also look similar.


G


L

The incorrect signs and transliterated letters are highlighted in brown in Table 2. Those signs and letters can be corrected to their intended versions as shown in Table 3.

Table 3. The Altai Mountain inscription after replacing incorrect signs with intended ones.

Row	Inscription	Transliteration	Meaning
a		E N I K^1 Ő	Enikő
b		E N I K^1 Ő M	my Enikő
c		SZ E R E T L E K^1	I love you
d1		SZ K^2 SZ	
d2		N A GY	great
e		M A GY A R O R SZ Á G	Hungary
f		K^2 U N P É T E R	Kun

The mix-up of the above pairs of Old Hungarian signs is a natural consequence of their similar look. Nevertheless, it is possible to ask why exactly these signs are mixed up in the inscription. To answer that question, we can apply a mathematically based approach to sign similarities. This approach was developed in an earlier paper that compared the Minoan Linear A, the Carian, and the Old Hungarian script [7]. The approach starts by identifying which sign has which of the following thirteen features:

1. The symbol contains some curved line.
2. The symbol encloses some region.
3. The symbol has a slanted straight line.
4. The symbol contains parallel lines.
5. The symbol contains crossing lines.
6. The symbol's top is a wedge $\wedge$.
7. The symbol's bottom is a wedge $\vee$.
8. The symbol's right side is a wedge $>$.
9. The symbol contains a stem, a straight vertical line that runs across the middle.
10. The symbol's bottom has two legs, two single lines touching the bottom.
11. The symbol's bottom has three legs, three single lines touching the bottom.
12. The symbol contains a hair, a small line extending from an enclosed space.
13. The symbol contains two triangles.

Figure 4 shows a matrix that results from a feature analysis of the Old Hungarian Runic signs in terms of the above 13 features.

Figure 5 shows a similarity matrix of the Old Hungarian signs. Each entry shows the number of features on which the row and the column signs agree. Two signs agree on a feature if they both contain the feature or both lack the feature. This means that they both have a value of 1 or they both have a value of -1 for the same feature in the feature table in Figure 4. We can propose the theorem below.

	1	2	3	4	5	6	7	8	9	10	11	12	13
4	-1	1	1	-1	-1	1	-1	-1	-1	-1	-1	1	-1
ᐊ	-1	1	1	-1	-1	1	-1	-1	-1	-1	-1	1	-1
X	-1	-1	1	-1	1	-1	-1	-1	-1	1	-1	-1	-1
↑	-1	-1	1	-1	-1	1	-1	-1	1	-1	-1	-1	-1
Ϥ	-1	1	1	1	-1	-1	-1	-1	-1	1	-1	1	-1
+	-1	-1	-1	-1	1	-1	-1	-1	1	-1	-1	-1	-1
Ʒ	1	-1	1	-1	1	-1	-1	-1	-1	1	-1	-1	-1
ʢ	1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
⊗	1	1	1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1
ʌ	-1	-1	1	1	-1	1	-1	-1	-1	-1	1	-1	-1
ǂ	-1	-1	-1	1	1	-1	-1	-1	1	-1	-1	-1	-1
ɤ	1	1	-1	-1	1	-1	-1	-1	-1	1	-1	1	-1
†	-1	-1	1	-1	1	-1	-1	-1	1	-1	-1	-1	-1
˥	-1	-1	1	-1	-1	1	-1	-1	-1	-1	-1	-1	-1
◊	-1	1	1	1	-1	1	1	1	-1	-1	-1	-1	-1
ϟ	-1	-1	1	1	-1	1	1	-1	1	-1	-1	-1	-1
Ѧ	-1	-1	1	1	-1	1	-1	-1	-1	-1	-1	-1	-1
ɵ	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
ᗕ	-1	1	1	1	-1	1	1	-1	-1	-1	-1	-1	1
Ↄ	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
D	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
ɔ	1	-1	-1	-1	-1	1	1	-1	-1	-1	-1	-1	-1
ɔ<	1	-1	1	-1	-1	1	1	-1	-1	-1	-1	-1	-1
Ǝ	-1	-1	1	1	-1	1	-1	-1	-1	1	-1	-1	-1
H	-1	-1	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
Λ	-1	-1	1	-1	-1	1	-1	-1	-1	1	-1	-1	-1
|	-1	-1	-1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1
Y	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
Ӿ	-1	-1	1	1	1	-1	-1	-1	-1	-1	1	-1	-1
ᛞ	-1	1	1	1	-1	-1	-1	-1	-1	-1	-1	-1	-1
⋈	-1	1	1	1	1	-1	-1	-1	-1	-1	-1	-1	1
M	-1	-1	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
ᛔ	-1	1	1	1	-1	-1	-1	-1	-1	1	-1	1	-1
ᛘ	-1	-1	1	-1	-1	-1	-1	-1	1	-1	-1	-1	-1

Figure 4. A feature analysis of the Old Hungarian Runic signs: 1 indicates that the sign in the row contains the feature in the column; -1 indicates that it does not contain the feature. This analysis uses the Altai Mountain version of the Z sign.

Theorem 1. *Let A be an $n \times m$ feature matrix with n signs and m features. Furthermore, let A^T be the transpose of A, and let M be the $n \times n$ similarity matrix for the n signs. Then, the following formula holds:*

$$M = 0.5\,((A \times A^T) + C), \tag{1}$$

where C is a matrix in which each entry is m.

Proof. Consider any entry $M[i, j]$ of the similarity matrix. This entry has the value of

$$M[i, j] = 0.5\,((\mathrm{A[i]} \cdot \mathrm{A[j]}) + \mathrm{m}), \tag{2}$$

where the dot indicates the dot product of the two vectors. The inner parenthesis in Equation (2) contains the number of times signs i and j that either both contain or both lack a feature minus the number of times they disagree on a feature as follows:

$$1 \times 1 = 1 \text{ when } i \text{and } j \text{ both contain a feature.} \tag{3}$$

$$(-1) \times (-1) = 1 \text{ when } i \text{and } j \text{ both lack a feature.} \tag{4}$$

$$(-1) \times 1 = -1 \text{ when } i \text{lacks and } j \text{contains a feature.} \tag{5}$$

$$1 \times (-1) = -1 \text{ when } i \text{contains and } j \text{lacks a feature.} \tag{6}$$

Let *agree* be the number of times that cases (3) and (4) occur. Let *disagree* be the number of times that cases (5) and (6) occur. Then, the following must hold for any number of features *m* because the two signs must either agree or disagree on each feature:

$$m = agree + disagree. \quad (7)$$

Hence, according to the above observation and Equation (7), the inner parenthesis has the following value:

$$agree - disagree = agree - (m - agree) = 2agree - m. \quad (8)$$

From Equation (8), it can be also seen that

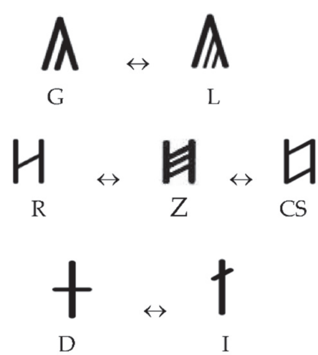
$$M[i, j] = 0.5((2agree - m) + m) = agree. \quad (9)$$

Therefore, the value of $M(i, j)$ is the total number of features on which signs *i* and *j* agree as required for the similarity matrix. QED.

Theorem 1 is useful for the fast calculation of the similarity matrix given any feature matrix. Theorem 1 was used to calculate the similarity matrix shown in Figure 5 from the feature matrix shown in Figure 4. After the similarity matrix was calculated, the entries with a similarity value of 12 or 13 between two different signs were highlighted in pink as shown in Figure 5.

The similarity matrix had 34 × 33 = 1122 nondiagonal entries. Out of those, 52 (4.63%) were marked pink. Intuitively, these pairs were those most likely to be confused with each other according to this mathematical model.

At my request, Klara Friedrich, a prominent researcher and teacher of the Old Hungarian Runic script, verified that, in her decades of experience, it is common to mix up the following letters:



Among the above, the G–L pair has a similarity of 12, the R–Z and the Z–CS pairs have similarities 11 and 13, respectively, and the D–I pair has a similarity of 12. Hence, these frequently mixed up pairs also have high similarity scores according to the similarity matrix in Figure 5. Hence, the strong agreement between the mathematical model and the teacher's experience shows that the G–L and R–Z pair mix-ups in the Old Hungarian Runic inscription in Figure 1 were likely due to an accident. □

	13	13	8	10	10	7	7	9	9	9	6	8	8	11	9	8	10	9	9	8	9	8	9	9	8	10	8	10	7	10	8	8	10	9
	13	13	8	10	10	7	7	9	9	9	6	8	8	11	9	8	10	9	9	8	9	8	9	9	8	10	8	10	7	10	8	8	10	9
	8	8	13	9	9	10	12	10	10	8	9	9	11	10	6	7	9	8	6	9	8	7	8	10	11	11	9	11	10	9	9	11	9	10
	10	10	9	13	7	10	8	10	8	10	9	5	11	12	8	11	11	8	8	9	8	9	10	10	9	11	11	11	8	9	7	9	7	12
	10	10	9	7	13	6	8	8	8	8	7	9	7	8	8	7	9	8	8	7	8	5	6	10	11	9	7	9	8	11	9	11	13	8
	7	7	10	10	6	13	9	9	9	7	12	8	12	9	5	8	8	9	5	10	9	8	7	7	8	8	12	10	9	8	8	8	6	11
	7	7	12	8	8	9	13	11	11	7	8	10	10	9	5	6	8	9	5	10	9	8	9	9	10	10	8	10	9	8	8	10	8	9
	9	9	10	10	8	9	11	13	11	9	8	8	10	11	7	8	10	11	7	12	11	10	11	9	10	10	10	12	9	10	8	10	8	11
	9	9	10	8	8	9	11	11	13	7	8	10	10	9	7	6	8	11	7	10	11	8	9	7	8	8	8	10	9	10	10	8	8	9
	9	9	8	10	8	7	7	9	7	13	8	4	8	11	9	10	12	7	9	8	7	8	9	11	10	10	8	10	11	10	8	10	8	9
	6	6	9	9	7	12	8	8	8	8	13	7	11	8	6	9	9	8	6	9	8	7	6	8	9	7	11	9	10	9	9	9	7	10
	8	8	9	5	9	8	10	8	10	4	7	13	7	6	4	3	5	10	4	9	10	7	6	6	7	7	7	7	6	7	7	7	9	6
	8	8	11	11	7	12	10	10	10	8	11	7	13	10	6	9	9	8	6	9	8	7	8	8	9	9	11	11	10	9	9	9	7	12
	11	11	10	12	8	9	9	11	9	11	8	6	10	13	9	10	12	9	9	10	9	10	11	11	10	12	10	12	9	10	8	10	8	11
	9	9	6	8	8	5	5	7	7	9	6	4	6	9	13	10	10	7	11	6	7	8	9	9	8	8	6	8	7	10	8	8	8	7
	8	8	7	11	7	8	6	8	6	10	9	3	9	10	10	13	11	6	10	7	6	9	10	10	9	9	9	9	8	9	7	9	7	10
	10	10	9	11	9	8	8	10	8	12	9	5	9	12	10	11	13	8	10	9	8	9	10	12	11	11	9	11	10	11	9	11	9	10
	9	9	8	8	8	9	9	11	11	7	8	10	8	9	7	6	8	13	7	12	13	10	9	7	8	8	10	10	7	10	8	8	8	9
	9	9	6	8	8	5	5	7	7	9	6	4	6	9	11	10	10	7	13	6	7	8	9	9	8	8	6	8	7	10	10	8	8	7
	8	8	9	9	7	10	10	12	10	8	9	9	9	10	6	7	9	12	6	13	12	11	10	8	9	9	11	11	8	9	7	9	7	10
	9	9	8	8	8	9	9	11	11	7	8	10	8	9	7	6	8	13	7	12	13	10	9	7	8	8	10	10	7	10	8	8	8	9
	8	8	7	9	5	8	8	10	8	8	7	7	7	10	8	9	9	10	8	11	10	13	12	8	7	9	9	9	6	7	5	7	5	8
	9	9	8	10	6	7	9	11	9	9	6	6	8	11	9	10	10	9	9	10	9	12	13	9	8	10	8	10	7	8	6	8	6	9
	9	9	10	10	10	7	9	9	7	11	8	6	8	11	9	10	12	7	9	8	7	8	9	13	12	12	8	10	9	10	8	12	10	9
	8	8	11	9	11	8	10	10	8	10	9	7	9	10	8	9	11	8	8	9	8	7	8	12	13	11	9	11	10	11	9	13	11	10
	10	10	11	11	9	8	10	10	8	10	7	7	9	12	8	9	11	8	8	9	8	9	10	12	11	13	9	11	8	9	7	11	9	10
	8	8	9	11	7	12	8	10	8	8	11	7	11	10	6	9	9	10	6	11	10	9	8	8	9	9	13	11	8	9	7	9	7	12
	10	10	11	11	9	10	10	12	10	10	9	7	11	12	8	9	11	10	8	11	10	9	10	10	11	11	11	13	10	11	9	11	9	12
	7	7	10	8	8	9	9	9	9	11	10	6	10	9	7	8	10	7	7	8	7	6	7	9	10	8	8	10	13	10	10	10	8	9
	10	10	9	9	11	8	8	10	10	10	9	7	9	10	10	9	11	10	10	9	10	7	8	10	11	9	9	11	10	13	11	11	11	10
	8	8	9	7	9	8	8	8	10	8	9	7	9	8	8	7	9	8	10	7	8	5	6	8	9	7	7	9	10	11	13	9	9	8
	8	8	11	9	11	8	10	10	8	10	9	7	9	10	8	9	11	8	8	9	8	7	8	12	13	11	9	11	10	11	9	13	11	10
	10	10	9	7	13	6	8	8	8	8	7	9	7	8	8	7	9	8	8	7	8	5	6	10	11	9	7	9	8	11	9	11	13	8
	9	9	10	12	8	11	9	11	9	9	10	6	12	11	7	10	10	9	7	10	9	8	9	9	10	10	12	12	9	10	8	10	8	13

Figure 5. A similarity matrix of the Old Hungarian Runic signs. Entries that indicate a similarity of 12 or 13 between two different signs are highlighted in pink.

Not everyone agrees with the accidental nature of the letter mix-ups. G. Varga imagined that the inscription had some sexual message. Moreover, he claimed that a male scribe wrote every sign originally correctly, but he later deliberately changed the inscription by adding extra lines for the sake of a woman called *Enikő*, who was embarrassed and 'obviously did not want to make public what happened'. According to Varga, these deliberately added extra lines explain the mix-up of the letters as shown in his figure (Figure 2 in [1]). However, this theory runs into a major problem in explaining the incorrect G in the word SZ E **Z** E T **G** E K[1].

Since scratches and carvings cannot be erased from a rock surface like from a paper, one cannot destroy a correct L into an incorrect G because it requires the deletion instead of the addition of a line. Hence, it is an untenable hypothesis that all spelling mistakes were deliberately introduced to destroy the meaning of the writing.

5. Decipherment Requires Correct Signs, Syntax, and Semantics

Valid decipherment requires correct signs, syntax, and semantics. These can be defined as described below.

1. Signs: This means a combination of two things.

First, the shapes of the signs are visually recognized correctly. As in the case of the Altai Mountain inscription, shape recognition can be hindered by deficiencies in the visual quality of the object (cracks in the rock, weathering, overwriting the signs by other inscriptions and drawings, etc.) and deficiencies in the photographs available to the investigator. An onsite investigation is almost always preferable to even the best available photograph.

Second, the visually correctly identified sign needs to also be correctly transliterated. It is of no use to correctly discern the shape of a sign, and then incorrectly look up its transliteration. Obviously, that cannot lead to a valid decipherment.

2. Syntax: This means that the words fit together according to the accepted grammatical rules. Moreover, the grammar must match the period of the inscription. For example, one cannot use present day Hungarian language grammar for an inscription from the Middle Ages. Translations that add suffixes purely from the imagination of the decipherer cannot be considered valid, even if the root words look acceptable. Even ancient Sumerian pictographs and cuneiforms reflect a well-formed, complex grammar.
3. Semantics: This means that the sentences and story are meaningful. The meaningfulness of the text needs to be evaluated in terms of the time and other circumstances of writing. For example, there should not be any anachronisms such as talking about dinosaurs in an ancient text because those became extinct long before the first scripts were developed.

In the Altai Mountain inscription, all the sign groups have an unambiguous reading except sign group (d). Now, let us evaluate the proposal (d2), which is equivalent to the word NAGY. If we read the sign groups in order from bottom up as shown in Figure 2, then we obtain the following Hungarian sentence:

E N I K^1 Ő, E N I K^1 Ő M, SZ E R E T L E K^1.

N A GY- M A GY A R O R SZ Á G, K^2 U N P É T E R.

Here, the Hungarian compound word *Nagy-Magyarország* 'Greater Hungary' refers to the historical Hungary, which includes present day Hungary and territories in neighboring countries where Hungarians live as minorities. It is necessary to add as an explanation that a literary reference to *Nagy-Magyarország* does not mean territorial aspirations but is only a reference to the international Hungarian ethnic community to which many minority Hungarians feel they belong. Hence, the inscription can be translated as a grammatically and semantically correct message as follows:

I love you Enikő, my Enikő!

–Peter Kun, Greater Hungary.

Now, let us consider the proposal (d1), which was SZ K^2 SZ. One can immediately see that this proposal has a weakness because this is not a meaningful word. It lacks vowels. In the older, mostly medieval examples of Old Hungarian Runic inscriptions, the vowels were often omitted when they did not affect the readability of the text. However, this is clearly not a medieval text. Some orthographic considerations regarding the form of the Old Hungarian signs support this assertion, but we can skip those considerations because there is a simpler explanation of recentness, i.e., that the name *Enikő* was created by the poet Vörösmarty (1800–1855) [9]. That linguistic consideration alone helps date the text to after the latter half of the 19th century. Hence, we need to consider a period when the omission of vowels was no longer practiced. This period includes a considerable revival of interest in the Old Hungarian Runic script in the past 30 years.

It is unlikely that the scribe wrote down each vowel in every other word except in SZ K^2 SZ. However, let us entertain this idea by trying to find a word. Since K^2 requires a back-vowel, a word that may be found is SZaK^2aSZ or *szakasz* (International Phonetic Alphabet notation:/sakas/) with the meaning 'segment'. However, this lacks correct semantics because the phrase SZeReTLeK^1 SZaK^2aSZ 'I love you segment' makes no sense.

Mr. Varga suggested the Hungarian word *szex* (International Phonetic Alphabet notation:/seks/) with the meaning 'sex'. Since letter X does not occur in the Old Hungarian Runic script, words with X are written down by a K SZ combination. Hence, let us try to write down the word as SZeK^2SZ. That would violate the second condition of sign correctness because one needs to transliterate K^2 as a consonant that occurs with a back-vowel, while *e* is a front-vowel.

The argument can be made that the scribe forgot about the differences between K^1 and K^2. However, it is unlikely because everywhere else the scribe uses these two signs correctly, as can be easily checked.

Front-vowel words: E N I K^1 Ő, E N I K^1 Ő M, SZ E R E T L E K^1.

Back-vowel word: K^2 U N.

Apparently, the scribe is consistent in the use of K^1 and K^2, and there is no real logic of supposing that they made a mistake just here regarding this usage convention, as well as making a mistake just here regarding explicitly writing down the vowel just in this word. Moreover, SZeK^2SZ is grammatically incorrect. A grammatically correct phrase would be the following:

E N I K^1 Ő, E N I K^1 Ő M, SZeK^2SZ-uálisan SZ E R E T L E K^1,

which means

I love you sexually Enikő, my Enikő.

However, the suffix *-uálisan* is completely absent. Hence, the SZeK^2SZ word proposal is semantically correct, but it is incorrect in signs and syntax. Despite the above concerns, this proposal of my coauthor was kept as an alternative together with my NAGY word proposal. Unfortunately, we omitted to mention that sign group (d) may be a personal sign or tamga, although Varga added the following endnote to his blog entry of 16 March 2022. The top shows a screenshot of the original Hungarian text, with an English translation in italics below.

(3) A szó olvasatát alátámasztja, hogy ez a *szeksz* magyarázza meg, miért is próbálta meg titkosítani a már elkészült feliratot Kun Péter. Ha ez egy tamga lenne, ahogyan azt Révész Péter említette, akkor a titkosítást még meg kellene magyarázni.

(3) The word's reading as 'sex' is supported by the fact that it explains why Peter Kun tried to destroy the readability of the inscription. If this were a tamga, as Peter Revesz once mentioned, then this deliberate destruction would be unexplained.

6. Identification of Sign Group (d) as a Tamga

A *tamga* is an emblem of a family, clan, or tribe. Tamgas were widely used by Eurasian nomads as a mark of personal property such as in branding livestock. For example, the early Bulgarian ruling dynasty, the Dulo clan, used the tamga shown in Figure 6a. For example, this tamga was found on the back of a seventh to ninth century bronze rosette at Pliska, Bulgaria [10] and on a ninth century clay pot fragment at Zalavár, Hungary [11]. The Kayi was one of the 21 Oghuz Turkic tribes. The Kayi tamga is shown in Figure 6b.

(a) (b) (c)

Figure 6. The Dulo clan's tamga (**a**), Kayi tamga (**b**), and Peter Kun's tamga (**c**). Picture credits: Wikipedia https://en.wikipedia.org/wiki/Dulo (accessed on 16 May 2022) and https://en.wikipedia.org/wiki/Tamga (accessed on 16 May 2022).

Thanks to the publicity of our publication [1], as a wonderful crowdsourcing effort, many people sent me various tips about who Peter Kun may be. When I got a tip about his phone number, I called him, and he verified that he was the scribe of the Altai Mountain inscription. I followed up our conversation with an email in which I asked some detailed questions. In his reply, which is shown in Figure 7, he explains that sign group (d) is a tamga. The middle K^2 sign stands for his family name, Kun, which has a back vowel. Hence, K^2 is used instead of K^1, which would be appropriate for a name with a front-vowel.

The two parallel signs on the left and right sides of the tamga are symbols of the Cumans, an ancient steppe people, whose domain extended from Hungary to Mongolia ca. 1200. Sometimes, the parallel lines are replaced by two arrows or spears. The three tamgas of Figure 6 all have two vertical parallel lines on the left and right sides. They differ only in the middle letter that is enclosed between those two parallel lines. These letters are Y-shaped for the Dulo clan, V-shaped for the Kayi tribe, and Z-shaped for Peter Kun. These three tamgas can be classified as members of the same subgroup of Turkic tamgas.

Peter Kun created this tamga for his own use in honor of his Cuman ancestors, who settled in a part of Hungary that is named after them to this day. It is called *Kunság* in Hungarian. The Cuman descendants in Hungary have their own organization, and Peter Kun serves as a leader in that organization. Peter Kun is also a cattle rancher and uses the tamga as a branding sign for his cattle.

Peter Kun verified that he did not make any deliberate alterations of the signs. He also explained that he was longing for *Enikő*, his wife, who was left behind in Hungary, while he was traveling in the Altai Mountains and doing research. He has a doctorate in Turkic studies. He even published a book about his research travels in Asia during which he studied the equestrian culture of the Steppe nomads [12].

Hence, the entire inscription can be seen as follows:

E N I K^1 Ő, E N I K^1 Ő M, SZ E R E T L E K^1.

, M A GY A R O R SZ Á G, K^2 U N P É T E R.

The tamga is not transliterated because it is a personal property symbol or emblem that can stand for 'Kun Ranch'. Hence, the correct translation into English is the following:

I love you Enikő, my Enikő!

–Peter Kun, Kun Ranch, Hungary.

Peter Dr. Kun May 15, 2022, 3:27 PM (2 days ago)
to me

Dear Peter, Mr Révész!
In the middle there is the ancient Hungarian runic 'k' which represents my family name.
The two paralel line is the symbol our tribe, the kuns or cumans what is rooted back to the vast steppes of Central-Asia and the same with Central -Asian kipchaks (among the kazaks, kirgiz, tatars, nogays, etc.)
This tribal brand can be found in Hungary and on land of Kazakstan, Tatarstan, among the nogays.
Sometimes it is not just a line but 2 paralel arrows or spears.
The outside circle is for the unit of the universe and also symbolizes the roof ring of the yurta.
This brand is used on my horses and cattle.
The script what You had sent to me is drawn by me in june in 2000 in the Altai mountains in Western-Mongolia as a memory of my beloved wife whom I missed so much, since she was at home in Hungary while I did my resarch work in Asia. These draws and scripts are close to Kstuu and Uighur rivers as I remember.
There are no mistakes or accidents on the scripts, maybe I was not so clever how to draw on rocks.
This was my first and last time I did so.
Hope my answers were clear.
Sincerely,
Peter

Figure 7. Dr. Peter Kun's email that verifies that he wrote the inscription in June 2000. This original email contains some minor misspellings. For example, the names of ethnic groups are written in lowercase letters, which is the common way of writing ethnic names in Hungarian.

7. Lessons Learned about Decipherment

That sign group (d) is a tamga did not seem plausible because there are no other instances of the use of tamga signs within Old Hungarian Runic inscriptions. Hence, this sign triplet can be termed a *hapax legomenon maximus* because it is not only unique within the corpus of Old Hungarian Runic inscriptions, but it is also unique in it being a tamga.

The Kun Ranch tamga is easily confusable with an SZ K^2 SZ sequence of Old Hungarian Runic signs as shown in Figure 8.

Figure 8. Confusability of Peter Kun's tamga (**left**) and Old Hungarian signs (**right**).

The presence of a *hapax legomenon maximus* together with the confusability of its elements with a sequence of Old Hungarian Runic sign made a complete decipherment of the Altai Mountain inscription nearly impossible. It is with luck that the actual scribe could be found and the exact meaning of the tamga was revealed to us.

Decipherers of ancient inscriptions may learn some valuable lessons from this work. As Figure 9 shows, only the tamga is the correct solution in this case. Unfortunately, it was not pursued enough because other proposals were not rejected earlier. In particular, the SZ eK^1 SZ proposal should have been dropped earlier when its problems became clear. My advice is to always look for a solution that satisfies the three S's of correct sign, syntax, and semantics and not to get stuck with any solution that fails any of these three criteria.

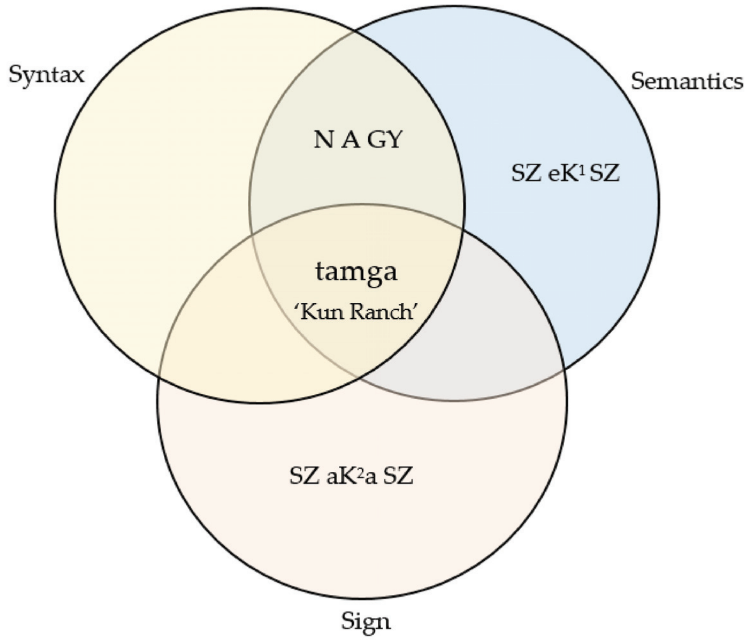


Figure 9. A valid decipherment needs to get three things correct: signs, syntax, and semantics. The above Venn diagram places four proposals for sign group (d) on the basis of correctness according to these three criteria.

8. Conclusions and Further Work

The Old Hungarian Runic inscription from the Altai Mountains now has a complete decipherment. The story of this inscription taught several valuable lessons that may be useful in the decipherment of other inscriptions in any script. Similarity matrices, which can be efficiently calculated using the formula in Theorem 1, may become generally used in future decipherments. It may be considered together with other machine-aided translation methods that use some type of similarity metrics [13,14]. This may aid in the continuing decipherment of the Indus Valley Script [15] and the Minoan scripts [16–18].

The work was also personally satisfying in contacting the scribe, who happened to be a generous and hardworking person, a cattle farmer from the Great Hungarian Plains, an adventurer. He is a great cultural ambassador between the peoples near the Altai Mountains and Hungarians in Central Europe. May this work also help to strengthen the cultural ties between the two regions.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Revesz, P.Z.; Varga, G. A proposed translation of an Altai Mountain inscription presumed to be from the 7th century BC. *Information* **2022**, *13*, 243. [CrossRef]
2. Sartkožauly, K. *Complete Atlas of the Orkhon Monuments*; Almaty Samga Press: Almaty, Kazakhstan, 2019; Volume 3.
3. Benkő, E.; Sándor, K.; Vásáry, I. *A Székely Írás Emlékei*; Bölcsészettudományi Kutatóközpont: Budapest, Hungary, 2021.
4. Wikipedia. Old Hungarian Script. Available online: https://en.wikipedia.org/wiki/OldHungarianscript (accessed on 8 December 2021).
5. Hosszú, G. *Scriptinformatics: Extended Phenetic Approach to Script Evolution*; Nap Kiadó: Budapest, Hungary, 2021.
6. Sebestyén, G. *A Magyar Rovásirás Hiteles Emlékei*; Magyar Tudományos Akadémia: Budapest, Hungary, 1915.
7. Revesz, P.Z. Establishing the West-Ugric language family with Minoan, Hattic and Hungarian by a decipherment of Linear, A. *WSEAS Trans. Inf. Sci. Appl.* **2017**, *14*, 306–335.
8. Wikipedia. Hungarian Phonology. Available online: https://en.wikipedia.org/wiki/Hungarian_phonology (accessed on 14 May 2022).
9. Wikipedia. Enikő. Available online: https://en.wikipedia.org/wiki/Enikő (accessed on 13 January 2022).
10. Wikipedia. Pliska Rosette. Available online: https://en.wikipedia.org/wiki/Pliska_rosette (accessed on 6 July 2022).
11. Szőke, B.M. *A Karoling-kor a Kárpát-Medencében*; Magyar Nemzeti Múzeum: Budapest, Hungary, 2014.
12. Kun, P. *Szelek Szárnyán*; Arcadas Press: Debrecen, Hungary, 2003.
13. Tóth, L.; Hosszú, G.; Kovács, F. Deciphering Historical Inscriptions Using Machine Learning Methods. In *Proceedings of the 10th International Conference on Logistics, Informatics and Service Sciences*; Liu, S., Bohács, G., Shi, X., Shang, X., Huang, A., Eds.; Springer: Singapore, 2020; pp. 419–435. [CrossRef]
14. Daggumati, S.; Revesz, P.Z. Data mining ancient scripts to investigate their relationships and origins. In Proceedings of the 23rd International Database Engineering and Applications Symposium, Athens, Greece, 10–12 June 2019; ACM Press: New York, NY, USA, 2019; pp. 209–218. [CrossRef]
15. Daggumati, S.; Revesz, P.Z. A method of identifying allographs in undeciphered scripts and its application to the Indus Valley Script. *Humanit. Soc. Sci. Commun.* **2021**, *8*, 50. [CrossRef]
16. Revesz, P.Z. Bioinformatics evolutionary tree algorithms reveal the history of the Cretan Script Family. *Int. J. Appl. Math. Inform.* **2016**, *10*, 67–76.
17. Revesz, P.Z. A translation of the Arkalochori Axe and the Malia Altar Stone. *WSEAS Trans. Inf. Sci. Appl.* **2017**, *14*, 124–133.
18. Revesz, P.Z. Experimental evidence for a left-to-right reading direction of the Phaistos Disk. *Mediterr. Archaeol. Archaeom.* **2022**, *22*, 79–96.

MDPI
St. Alban-Anlage 66
4052 Basel
Switzerland
www.mdpi.com

Information Editorial Office
E-mail: information@mdpi.com
www.mdpi.com/journal/information

MDPI

Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-1370-4