

Article

AraFast: Developing and Evaluating a Comprehensive Modern Standard Arabic Corpus for Enhanced Natural Language Processing

Asmaa Alrayzah ^{1,2,*} , Fawaz Alsolami ¹  and Mostafa Saleh ¹

¹ Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; falsolami1@kau.edu.sa (F.A.); msherbini@kau.edu.sa (M.S.)

² Department of Computer Science, College of Computer Science and Information Systems, Najran University, Najran 55461, Saudi Arabia

* Correspondence: asalrayzah@nu.edu.sa

Abstract: The research presented in the following paper focuses on the effectiveness of a modern standard Arabic corpus, AraFast, in training transformer models for natural language processing tasks, particularly in Arabic. In the study described herein, four experiments were conducted to evaluate the use of AraFast across different configurations: segmented, unsegmented, and mini versions. The main outcomes of the present study are as follows: Transformer models trained with larger and cleaner versions of AraFast, especially in question-answering, indicate the impact of corpus quality and size on model efficacy. Secondly, a dramatic reduction in training loss was observed with the mini version of AraFast, underscoring the importance of optimizing corpus size for effective training. Moreover, the segmented text format led to a decrease in training loss, highlighting segmentation as a beneficial strategy in Arabic NLP. In addition, using the study findings, challenges in managing noisy data derived from web sources are identified, which were found to significantly hinder model performance. These findings collectively demonstrate the critical role of well-prepared, segmented, and clean corpora in advancing Arabic NLP capabilities. The insights from AraFast's application can guide the development of more efficient NLP models and suggest directions for future research in enhancing Arabic language processing tools.



Citation: Alrayzah, A.; Alsolami, F.; Saleh, M. AraFast: Developing and Evaluating a Comprehensive Modern Standard Arabic Corpus for Enhanced Natural Language Processing. *Appl. Sci.* **2024**, *14*, 5294. <https://doi.org/10.3390/app14125294>

Academic Editors: José Ramón Méndez Reboledo and David Ruano-Ordás

Received: 29 April 2024

Revised: 9 June 2024

Accepted: 11 June 2024

Published: 19 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: Arabic corpora; Arabic corpus; Arabic NLP; Arabic dataset; transformer model; question answering

1. Introduction

Arabic is one of the most widely spoken languages, with over 300 million native speakers worldwide [1]. It is also the official language of 22 countries across the Middle East and North Africa [2]. Despite its widespread use, there is a lack of extensive research in Arabic, mainly due to the absence of accessible and large Arabic corpora [3]. These corpora are integral for studying various linguistic aspects, such as morphology, syntax, semantics, discourse, and pragmatics, and for developing natural language processing (NLP) systems [4].

Collecting an Arabic corpus for linguistic research serves several critical purposes. First, corpora enable the study of the Arabic language in its authentic form [5]. This differs from traditional linguistic research methods that rely on small artificial datasets. Second, corpora are essential for the development of NLP systems, which are crucial for machine translation, information retrieval, and text analysis applications [6]. Using corpora to train NLP systems on the patterns of human language enables these systems to effectively recognize and understand different linguistic phenomena. Third, corpora allow for comparisons between Arabic and other languages [7], facilitating the identification of similarities and differences to further enhance our understanding of language dynamics.

Numerous studies have shown that large-scale corpora improve the performance of transformers, particularly for NLP tasks [5,8]. Transformer-based language models have

demonstrated remarkable independent learning abilities in discovering relevant language representations without supervision. Consistent access to large English corpora gives English models an advantage over models for other languages [8].

A comprehensive Arabic corpus that consists of authentic Arabic texts is essential for in-depth linguistic analysis. The findings presented in the following article emphasize the significance of collecting an Arabic corpus and its implications for linguistic research. To this end, roughly 48 corpora were gathered from various sources, including dataset repositories mentioned in scientific papers (32 corpora) and repositories without accompanying publications (16 corpora). The majority of the collected corpora focus on modern standard Arabic (MSA) and classical Arabic (CA), excluding Arabic dialects.

The following research questions were addressed in this present study:

RQ1: What freely available MSA corpora can be found online?

RQ2: Does pretraining a transformer model on a large Arabic corpus that includes web-scraped data lead to worse performance than training without web-scraped data and noise?

RQ3: Does pretraining a transformer model on a segmented Arabic corpus lead to better performance than training the model on an unsegmented Arabic corpus?

RQ4: How does pretraining a transformer model on a large Arabic corpus impact its accuracy in NLP tasks, specifically Arabic QA tasks?

The contributions of the present study are as follows:

1. A detailed exploration and critical evaluation of freely accessible modern standard Arabic (MSA) corpora from various online sources is performed.
2. AraFast, a comprehensive and large-scale Arabic corpus, is introduced to support advanced NLP tasks.
3. An extensive assessment of the performance impacts of using different versions of the Arabic corpus, including segmented, unsegmented, mini, and noisy variants, in training NLP models is conducted.

The remainder of this paper is organized as follows: Section 2 highlights the motivations and main objectives of the present study. Section 3 presents a comprehensive review of the literature on the collection of Arabic corpora. Section 4 describes the research methodology employed in this study in detail. Section 5 details four experiments focused on training an Arabic transformer model using the AraFast corpus. Section 6 presents the use of the corpus in developing two transformer models and a discussion of the results. The challenges and limitations of the following study are elaborated in Section 7. Finally, Section 8 concludes the paper, offering insights and suggestions for the application of this Arabic corpus in future studies.

2. Motivation and Objectives

In the fast-evolving field of NLP, the availability of high-quality, accessible data are critical. The present study was initiated with the primary goal of developing a freely accessible Arabic corpus, AraFast, tailored to enhance NLP applications such as language modeling. The necessity for such a corpus stems from the considerable time and resources currently required for data collection, which poses a significant barrier to research and development in Arabic NLP.

Arabic, being a rich and complex language, presents unique challenges in text processing and generation. Existing corpora are either lacking for comprehensive NLP tasks or are not freely available, or there is a need for extensive preprocessing, limiting their utility for widespread research and application development. By creating AraFast, we aim to fill this gap, offering a robust Arabic corpus that supports not only basic NLP tasks, but also more advanced applications such as chatbots, question-answering (QA) systems, summarization tools, and sophisticated Arabic text generation.

Establishing an Arabic corpus, particularly one drawn from broad and diverse content, holds significant importance for several reasons. Firstly, the collected corpora were assembled to create a ready-to-use corpus for future research, which helps in enriching NLP tasks such as the QA and text generation domains.

Moreover, new language models, especially currently trending models such as large language models (LLMs), require rich information sources to become robust language models [9]. LLMs are built based on deep learning approaches. However, deep learning approaches are data-hungry, meaning that a large amount of data are required to improve the performance of such models [10].

Furthermore, by combining corpora from various sources, biases present in individual corpora can be mitigated, leading to more balanced and fair models. Compiling multiple corpora can significantly expand the linguistic diversity covered. A larger and more diverse corpus can help in training models that generalized better to unseen data, thus improving their usability in real-world applications. A well-constructed corpus serves as a fundamental resource for linguistic studies and the development of NLP tasks.

Another significant advantage of a ready-to-use corpus is the considerable time it saves for researchers. Rather than spending valuable time searching for, cleaning, and preprocessing data, researchers can use the corpus in its present condition and immediately begin the model training process. This process accelerates the research and development cycle, allowing researchers to focus more of their time on algorithms for building pretrained models and developing Arabic NLP applications.

By using such a corpus, the findings of the present study are expected to contribute valuable resources that will aid in the advancement of Arabic NLP applications in the field of computational linguistics. In addition, the development of a large corpus will provide essential training and validation data that will help in building and training deep learning models, especially for tasks such as translation, sentiment analysis, and chatbot development.

The objectives of the present study are threefold:

1. To conduct a comprehensive literature review of existing freely available online MSA corpora.
2. To develop a large MSA corpus named AraFast.
3. To investigate the impact of using a large, segmented, clean Arabic corpus in NLP tasks, with a specific focus on QA.

Through this work, AraFast is expected to become a key resource that will greatly improve the capabilities and efficiency of technologies working with the Arabic NLP.

3. Literature Review

Arabic, as one of the most widely spoken languages worldwide, plays a crucial role in areas such as NLP, computational linguistics, and language technology development. For the advancement of research in these fields, the availability of high-quality Arabic corpora is paramount. However, despite the growing interest in Arabic language resources and Arabic NLP, there is a noticeable lack of studies on publishing Arabic corpora [4].

In 2022, Alyafeai et al. [11] introduced “Masader”, a public catalogue of Arabic NLP datasets that includes 200 datasets annotated with 25 attributes. The authors also proposed a metadata annotation strategy that can be extended to other languages. However, Masader only contains Arabic datasets for testing and fine-tuning machine learning and deep learning models; it does not include corpora for training transformer models.

Ahmed et al. [3] conducted a scoping review to identify freely available Arabic corpora and subsequently presented a collection of 48 freely accessible sources. In their review, they provided key information about each corpus, including its name, type or purpose, corresponding website URL, size, and the primary sources from which it was compiled. In line with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses, only published, peer-reviewed studies were included [12]. However, not all available corpora were included, as the focus was on freely accessible resources; corpora that required a subscription or contacting the owners were excluded [3].

Ali [8] discussed, in their study, the design and creation of an extensive Arabic corpus comprising over 500 GB of data. The primary objective of their study was to enhance the cross-domain knowledge of large-scale language models. The above corpus was developed

by collecting text from 22 diverse sources. However, despite being the largest, most diverse, clean Arabic corpus ever compiled, it is not publicly accessible to other researchers aiming to improve models and systems for NLP tasks.

There is a shortage of Arabic corpora and collections of related published papers, and access is limited. Compared to other languages, there are few freely available and comprehensive Arabic corpora. This limitation hinders the development and evaluation of NLP models and algorithms specifically tailored to Arabic. Additionally, diverse data sources are often required to ensure the robustness and generalizability of models used for Arabic NLP tasks. While efforts have been made to create Arabic corpora, there is still a need for diverse, representative resources that cover a wide range of genres, registers, and writing styles. Such diversity is crucial for developing training models that can effectively handle the intricacies of various types of Arabic text.

In Table 1, a comparison of our study and related works is presented, taking into account criteria such as the year of the study, the type of data, and the type of Arabic text analyzed. The data presented in the table also indicate whether the study authors proposed a text preprocessing algorithm and, if so, whether they specified the text size before and after the preprocessing stage; initial preprocessing generally involves crucial steps for cleaning the text and preparing it for training transformer models. Furthermore, in the Table below, the availability of public data is also highlighted.

Table 1. Comparison of related works (Note: MSA: (modern standard Arabic), N/A: (not available) and AD: (Arabic dialects)).

Reference	[11] Alyafeai et al.	[3] Ahmed et al.	[8] Ali	Ours
Year of study	2021	2022	2022	2023
Type of data	Dataset: text/speech	Corpora: text/speech	Corpora: text	Corpora: text
Type of text	MSA/DA	MSA/DA	MSA/DA	MSA
Size before preprocessing	N/A	N/A	1 TB	833 GB
Size after preprocessing	N/A	N/A	500 GB	112 GB
Preprocessing algorithm	✗	✗	✓	✓
Research methodology	✓	✓	✗	✓
Availability of data	✓	✗	✗	✓

4. Methodology

Equations: Rowley and Slack's [13] keyword-based approach to literature reviews was adopted as the research methodology for this study, consisting of five main steps: (1) conducting resource searches, (2) applying selection criteria to filter the results, (3) retrieving metadata information, (4) performing initial processing, and (5) analyzing the results. The main steps with brief descriptions of each step are illustrated in Figure 1.

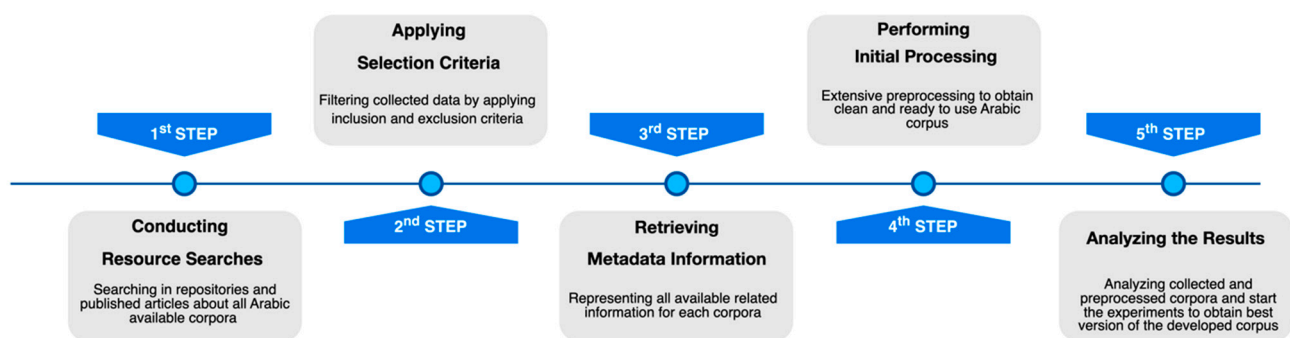


Figure 1. Methodology used in the present study.

A predefined set of keywords was used to identify relevant Arabic language resources, and an initial list of corpus sources was compiled. Subsequently, a filtration process guided by specific inclusion criteria was implemented to refine the collected resources. The

corpora that met the criteria were examined and added to the final list, along with their associated metadata. The gathered data underwent preliminary processing, and the final set of resources was then analyzed, taking into account the accompanying metadata. The relevant findings are presented in the following paper.

The initial search and filtering steps took place between August and September 2022, while the process of downloading the corpus list occurred after the information on all of the corpora was gathered and synthesized, with this process continuing until March 2023. In the following subsections, detailed descriptions of each step are provided.

4.1. Conducting Resource Searches

The Google search engine was used to conduct a targeted search to directly identify Arabic NLP corpora. In addition, a set of keywords was used to search for specific corpora in well-known data repositories and on well-known indexing websites: GitHub (<https://github.com> accessed on 13 August 2022), Kaggle (<https://www.kaggle.com> accessed on 10 August 2022), Metatext (<https://metatext.io> accessed on 27 August 2022), Huggingface (<https://huggingface.co> accessed on 2 September 2022), Google Scholar (<https://scholar.google.com> accessed on 3 August 2022), and SourceForge (<https://sourceforge.net> accessed on 16 September 2022). The search included terms related to NLP and the Arabic language, including various variations as follows:

- “Arabic” AND “NLP” OR “Natural Language Processing” AND “Resource”.
- “Arabic” AND “NLP” OR “Corpus”.
- “Arabic” AND “NLP” OR “Corpora”.
- “Arabic” OR “NLP” OR “Lexical database”.
- “Arabic” OR “Arab” OR “Arabian” AND “Corpora”.
- “Arabic” OR “Arab” OR “Arabian” AND “Corpus”.

Using the above step, we were able to generate our initial list of approximately 80 data sources.

4.2. Applying Selection Criteria for Filtering

In addition to the above search process, the retrieved corpora and their associated articles, when available, were manually reviewed to improve the data collection process. This manual screening process involved a predefined set of inclusion and exclusion criteria, which are detailed below in Table 2. From the preliminary list, roughly 55 corpora were retained, and those used for NLP tools, such as named-entity recognition (NER) and/or part of speech, and those that did not contain ordinary text, such as word lists, were excluded.

Table 2. Inclusion and exclusion criteria.

Inclusion Criteria
Freely available/accessible Arabic corpora/corpus
Modern standard Arabic corpora/corpus
Classical Arabic corpora/corpus
Exclusion criteria
Paid Arabic corpora/corpus
Arabic dialect corpora/corpus
Arabic corpus/corpora serving natural language processing tools such as spell-checking, named-entity recognition, or part of speech

4.3. Gathering Metadata

Metadata, including names, related articles, URLs, sizes, owners or authors, and references, were gathered and compiled from various sources, resulting in the accumulation of metadata for 48 corpora (see Table 3). Among them, 32 corpora were obtained by following the links provided in scientific papers to dataset repositories. The remaining 16 corpora were sourced from repositories for which no associated published papers

were found. The majority of the collected data comprised MSA and CA text, without the inclusion of dialectal variations. As illustrated in Table 3, this allowed us to answer the first research question (RQ1) noted in this section: What freely available MSA corpora can be found online? The answer is summarized in more detail in Table 3.

Table 3. Collected corpora information.

Ref.	Corpus Name (Categories)	Repository/Link to Access	Size
[14]	OSCAR corpus (web crawling data)	OSCAR's web page https://oscar-project.org accessed on 3 August 2022	~82 GB
[15]	Arabic Wikipedia articles (encyclopedia articles)	Wikimedia dump: https://dumps.wikimedia.org/arwiki/ accessed on 3 August 2022	~2.27 GB
[16]	Aljazeera News (Aljazeera articles through web scraping)	Kaggle: https://www.kaggle.com/datasets/arhouati/arabic-news-articles-from-aljazeeranet accessed on 10 August 2022	~1.67 GB
[17]	SANAD (culture, health, politics, religion, and tech)	Kaggle: https://www.kaggle.com/datasets/haithemhermessi/sanad-dataset accessed on 10 August 2022	~180 MB
[18]	KSUCCA (classical Islamic Arabic texts)	SourceForge: https://sourceforge.net/projects/ksucca-corpus/ accessed on 16 September 2022	~462 MB
[19]	Arabic BERT Corpus (historical and Arabic Wikipedia)	Kaggle: https://www.kaggle.com/datasets/abedkhooli/arabic-bert-corpus accessed on 10 August 2022	~6 GB
[20]	Arabic News Dataset (entrepreneurship, science, and tech)	Kaggle: https://www.kaggle.com/datasets/asmaaabelwahab/arabic-news-dataset accessed 10 August 2022	~33 MB
[21]	Watan/Khaleej news (culture, religion, economy, sport)	Soft112.com: https://arabic-corpus.soft112.com accessed on 26 September 2022	~152 MB
[22]	Tashkeela (Islamic classical books)	SourceForge: https://sourceforge.net/projects/tashkeela/ accessed on 16 September 2022	~610 MB
[23]	ANT (culture, economy, politics, society, and sport)	GitHub: https://antcorpus.github.io accessed on 13 August 2022	~18 MB
[24]	MNAD (business, policy, national, health, sport, and tech)	Kaggle: https://www.kaggle.com/datasets/jmourad100/mnad-moroccan-news-articles-dataset accessed on 10 August 2022	~1.06 GB
[25]	CC100-Aarbic Dataset (web crawling data)	Metatext: https://metatext.io/datasets/cc100-arabic accessed on 27 August 2022	~28 GB
[26]	EASC (Arabic articles with human-generated summaries)	Metatext: https://metatext.io/datasets/essex-arabic-summaries-corpus-(easc) accessed on 29 August 2022	~3.2 MB
[27]	Arabic in Business and Management Corpora (economic)	Metatext: https://metatext.io/datasets/arabic-in-business-and-management-corpora-(abmc) accessed on 29 August 2022	~4.5 MB
[28]	Abu El-Khair corpus (articles from eight Arabic countries)	Abu El-Khair's web page: http://www.abuelkhair.net/index.php/en/arabic/abu-el-khair-corpus accessed on 20 September 2022	~18 GB

Table 3. Cont.

Ref.	Corpus Name (Categories)	Repository/Link to Access	Size
[29]	Saudi-NewsNet (Saudi newspapers articles)	Metatext: https://metatext.io/datasets/saudi-newspapers-corpus accessed on 27 August 2022	~103 MB
[30]	News Commentary (political and economic)	OPUS: https://opus.nlpl.eu/results/ar&ar/corpus-result-table accessed on 26 September 2022	~119 MB
[31]	OSIAN (international Arabic news)	CLARIN https://vlo.clarin.eu/search?0 accessed on 23 September 2022	~3 GB
[32]	KACST (Islamic topics and Arabic poems)	SourceForge: https://sourceforge.net/projects/kacst-acptool/ accessed on 16 September 2022	~13.6 MB
[33]	Arabic Big Corpus (Arabic book reviews)	GitHub: https://github.com/mohataher/arabic_big_corpus accessed on 13 August 2022	~746 KB
[34]	AKEC (Arabic articles and their key phrases)	GitHub: https://github.com/ailab-uniud/akec accessed on 13 August 2022	~1.3 MB
[31]	Osac (economic, history, education, religious, and health)	GitHub: https://github.com/motazsaad/osac-corpus accessed on 13 August 2022	~567 KB
[35]	BBC Crawler (business and sport)	GitHub: https://github.com/motazsaad/bbc-crawler accessed on 13 August 2022	~38 MB
[36]	Arabic Stories Corpus (short stories)	GitHub: https://github.com/motazsaad/Arabic-Stories-Corpus accessed on 13 August 2022	~606 KB
[37]	Arabic-News (different categories)	GitHub: https://github.com/motazsaad/Arabic-News accessed on 13 August 2022	~1.78 GB
[38]	Tashkeela2 (economy, health, politics, sport, sociology)	GitHub: https://github.com/motazsaad/tashkeela2/tree/master/data accessed on 13 August 2022	~9 MB
[39]	Ajdir Corpus (BBC, CNN, and Aljazeera news)	AraCorpus's web page: http://aracorpus.e3rab.com/argistestsrv.nmsu.edu/AraCorpus/ accessed on 16 September 2022	~918 MB
[40]	Common Crawl (CC) (web crawling data)	STATM: https://data.statmt.org/ngrams/raw/ accessed on 1 September 2022	~457 GB
[41]	Shamela Corpus (Islamic topics)	GitHub: https://github.com/OpenITI/RAWrabica045000 accessed on 14 August 2022	~12.01 GB
[42]	Commonsense Validation (commonsense sentences)	Metatext: https://metatext.io/datasets/arabic-dataset-for-commonsense-validation- accessed on 27 August 2022	~1.5 MB
[43]	QED Corpus (education lectures)	Qatar Computing Research Institute: https://alt.qcri.org/resources/qedcorpus/ accessed on 16 September 2022	~11 MB
[44]	ArSAS Corpus (Arabic tweets)	SMASH: https://homepages.inf.ed.ac.uk/wmagdy/resources.htm accessed on 17 September 2022	~1.9 MB

Table 3. Cont.

Ref.	Corpus Name (Categories)	Repository/Link to Access	Size
[45]	BRAD-Arabic-Dataset (book reviews)	GitHub: https://github.com/elnapara/BRAD-Arabic-Dataset/tree/master/data accessed on 16 August 2022	~11.61 GB
[46]	Osman Arabic Text Readability (narratives)	SourceForge: https://sourceforge.net/projects/osmanreadability/ accessed on 16 September 2022	~49 MB
[47]	CLCL (Arabic, French, and English Wikipedia articles)	SourceForge: https://sourceforge.net/projects/crlcl/ accessed on 16 September 2022	~4.4 MB
[48]	Arabic Corpus (Arabic books)	SourceForge: https://sourceforge.net/projects/newarabiccorpus/ accessed on 16 September 2022	~233 MB
[49]	Arabic word corpus (different categories)	SourceForge: https://sourceforge.net/projects/arabicwordcorpu/files/ accessed on 17 September 2022	~5.4 MB
[50]	A Corpus of Arabic Literature (history)	Zenodo: https://zenodo.org/record/5772261#.Y2eeoi8RqR accessed on 23 September 2022	~190 MB
[51]	OpenITI Corpus (Islamic topics)	GitHub: https://github.com/OpenITI/RELEASE/tree/v2019.1.1 accessed on 16 August 2022	~2 GB
[52]	Arabic learner corpus (education)	Arabic Learner Corpus website: https://www.arabiclearnercorpus.com accessed on 23 September 2022	~2.3 MB
[53]	Kunuz corpus (Islamic books)	JARIR at Manouba University: http://www.jarir.tn/kunuzcorpus accessed on 23 September 2022	~42 MB
[54]	Habibi corpus (Arabic song lyrics)	Habibi's web page: http://ucrel-web.lancaster.ac.uk/habibi/ accessed on 23 September 2022	~135 MB
[55]	Hindawi Books (science, literature, sport, and health)	Hindawi's web page: https://www.hindawi.org/books/ accessed on 28 September 2022	~27 GB
[56]	OpenArabic/1300AH (Islamic topic)	GitHub repository: https://github.com/OpenArabic/1300AH accessed on 16 August 2022	~731 MB
[57]	Corpora-List (classical text)	Corpora-List's web page: http://korpus.uib.no/icame/corpora/2004-2/0064.html accessed on 16 September 2022	~894 MB
[58]	AraNPCC (medical)	Internet Archive library: https://archive.org/details/AraNPCC accessed on 20 September 2022	21 GB
[59]	OpenITI-proc corpus (Islamic sentences)	Zenodo: https://zenodo.org/record/2535593#.Y4NKBi8RpD2 accessed on 23 September 2022	13 GB
[60]	Corpora Arabic (news crawl)	Leipzig Corpora collection: https://wortschatz.uni-leipzig.de/en/download/Arabic accessed on 23 September 2022	1.8 GB
Total before/after filtering and preprocessing of the corpus		Roughly 833 GB of material was collected/112 GB; the corpus was called "AraFast".	

4.4. Preliminary Processing

After the data collection and scraping processes were complete, the data for processing were then ready for preprocessing. This process involved filtering and tagging the data, which were primarily HTML pages with various Arabic encodings. For example, a Python version 3.8.3 script was used to convert articles from the HTML format to plain text for the Watan and Khaleej news corpus [6,21]. Meanwhile, certain files that contained unreadable symbols required decoding to UTF-8, which made the text accessible on web pages and to the Arabic model. It was important to decode all collected text to a unified Unicode standard format. Subsequently, the data were converted to follow a standardized encoding suitable for further analysis using Python scripts.

In the case of Wikipedia text, the tags and titles were removed, leaving only the contents of the articles, which were then saved in plain text format. The WikiExtractor version 3.3 tool was used for Arabic Wikipedia articles. Numerous corpora, especially Arabic news articles, were downloaded in CSV format. A Python script was then used to extract the articles and save them in plain text format.

For large corpora such as common crawl (CC) [40], the BigTextFileSplitter version 3.8.2 (<https://www.withdata.com/big-text-file-splitter/> accessed on 2 April 2023) tool was used to split such large corpora. Furthermore, some texts, such as the Tashkeela corpora [22], contained emojis and diacritics, which were removed using Python scripts. Several corpora, such as the Saudi-NewsNeCorpus, were converted from JSON format to plain text via a Python script.

The size of the collected corpora was 833 GB before scanning, filtering, and selecting the relevant data. After removing repeated articles and files, the size was reduced to 756 GB. Then, the process of extracting text and articles from various file types, such as converting EPUB to TXT format, was applied, resulting in about 720 GB of data. After applying preprocessing steps to clean the data, noisy data were identified and removed, reducing the corpus size to roughly 249 GB. Following this stage, the preprocessing step was conducted.

For preprocessing in general, Figure 2 illustrates the main preprocessing steps for cleaning and organizing Arabic corpora. The process begins with the collection of corpora, followed by the inspection of files and articles to identify duplicates. If duplicate articles or files are found, they are removed. The next step involves checking for duplicated text within the files. If duplicated text is detected, this text is also removed. Once the data are free of duplicates, unwanted characters, symbols, and links are removed to ensure that the text is clean. Finally, the cleaned text is saved as a separate file in a directory named AraFast corpus, marking the end of the preprocessing stage.

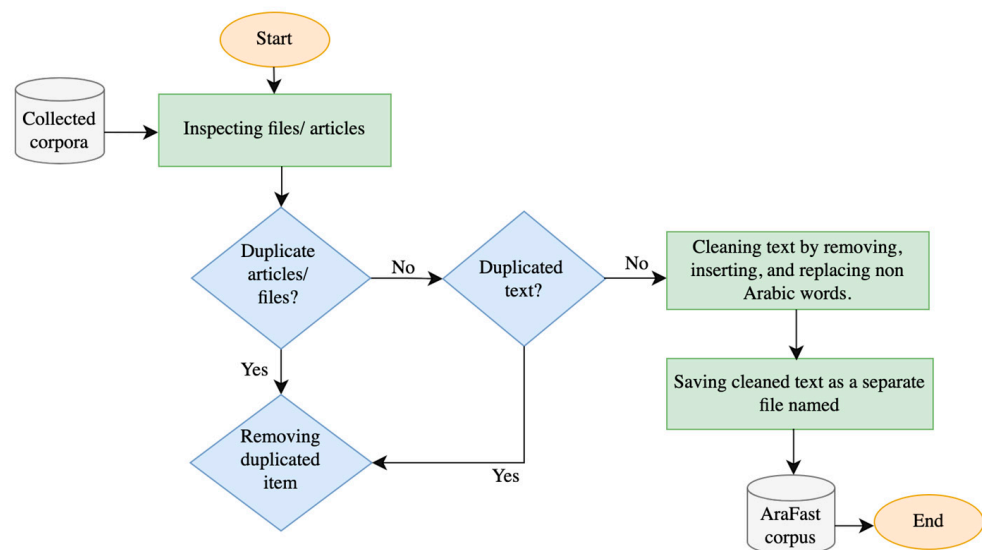


Figure 2. Flowchart of the preprocessing stage.

In the preprocessing stage, the steps detailed in Algorithms 1 and 2 are followed to prepare and clean the overall corpus for training language models. Algorithm 1 enhances the quality and consistency of Arabic text data by removing redundancies and various forms of noise. It starts by identifying and excluding duplicate entries across different files to maintain dataset integrity. It then performs text cleaning operations, including removing URLs, links, emails, HTML line breaks, markup, repeated characters, extra spaces, and other unwanted elements such as Tatweel, punctuation, noisy characters, and diacritics. It also replaces slashes with dashes and ensures proper spacing around non-Arabic digits, English digits, brackets, and between words and numbers. This thorough cleaning and standardization process improve dataset quality, making it more suitable for training Arabic language models and other NLP tasks. Well-processed and clean text is essential for achieving accurate and reliable results and for the effective training of pretrained language models (PLMs).

Additionally, Algorithm 2 merges preprocessed text files into a single directory for easier organization and management. It starts by ensuring the existence of an output folder, creating one if necessary. It then iterates through a list of preprocessed text files, reading each file's content and saving it with a unique name in the output folder. This process centralizes the preprocessed text files, supporting enhanced data management and facilitating subsequent analysis or training of language models. Having all preprocessed text in one location is crucial for systematic data handling and the effective training of PLMs. The above preprocessing algorithms resulted in a cleaned and preprocessed corpus of 112 GB in size, ready to be used for pretraining a language model.

4.5. Results of the Analysis

In the following section, the findings of the data analysis are presented in Table 3, with a specific focus on freely accessible Arabic corpora. As shown in Figure 3, most of the corpora were obtained from reliable repositories, with a considerable number retrieved from the GitHub repository. However, it is worth noting that of the 48 corpora analyzed, 17 were obtained from alternative sources, namely, websites owned by the respective researcher or publisher.

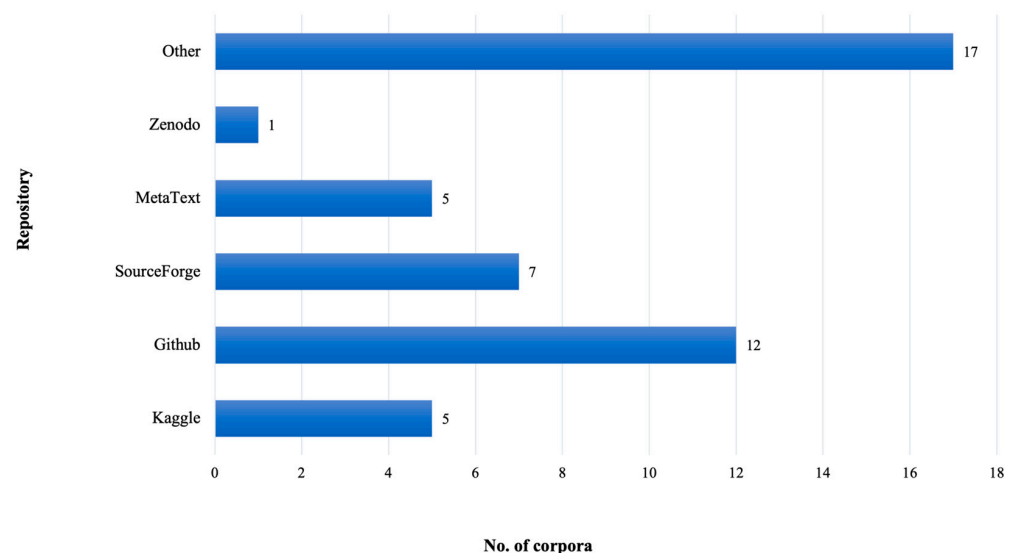


Figure 3. Most widely used repositories to host corpora.

In terms of corpus size, the analysis revealed that 20 corpora were measured in gigabytes (GB), 25 were measured in kilobytes (KB), and the remaining 3 were measured in megabytes (MB), as shown in Figure 4. The variations in corpus size may be due to differences in content type and format. Corpora such as open super-large crawled aggregated corpus (OSCAR) involve the continuous generation of content [14]; in contrast,

other corpora, such as common crawl (CC) corpora, consist of data scraped from the web over several years [25], and some corpora, such as Hindawi, are derived from books. Such corpora tend to be of larger sizes measured in GB; in contrast, corpora that primarily contain articles extracted from CSV files and converted to plain text are in measured KB.

Algorithm 1: Preprocessing algorithm for Arabic raw text

```

Input: Raw text (text)
Output: Preprocessed text
BEGIN
1  Function preprocess (text):
2      Initialize unique-texts as an empty set//Initialize a set to store unique texts
3      For each file-1 in folder-1://Loop through the files in folder-1
4          For each file-n in folder-n://Loop through the files in folder-n
5              If file-1.strip() is equal to file-n.strip()//Check for duplicate texts
6                  Continue to the next iteration//If duplicate, continue to the next iteration
7              End If
8              Add file-1.strip() to unique-texts//Add the text to unique-texts if not a duplicate
9          End For
10     End For
11     Remove URLs from text//Text cleaning operations
12     Remove links from text
13     Remove emails from text
14     Remove HTML line breaks from text
15     Remove HTML/XML markup from text
16     Remove repeated characters from text
17     Remove extra spaces from text
18     Remove Tatweel from text
19     Remove punctuation from text
20     Remove noisy characters from text
21     Remove diacritics from text
22     Remove non-digit repetition from text
23     Replace slash with dash in text
24     Insert whitespace before and after non-Arabic digits, English Digits, and brackets in text
25     Insert whitespace between words and numbers or numbers and words in text
26     Return the preprocessed text//Return the preprocessed text
27 END Function
28 END
  
```

Algorithm 2: Combining preprocessed text into one folder

```

Input: Preprocessed text (text)
Output: Combined text in one folder (combined-text)
1 BEGIN
2  Function un-preprocess (output-text1, output-folder):
3      Create the output folder if it doesn't exist//Create the output folder if it doesn't exist
4      For each i and f-name in enumerate (output-text1)://Loop through the list of file names
5          Read the content of f-name into text//Read the content of the file into text
6          Save text as a separate file named "output-i.txt" in the output-folder
7      End For
8  END Function
9 END
  
```

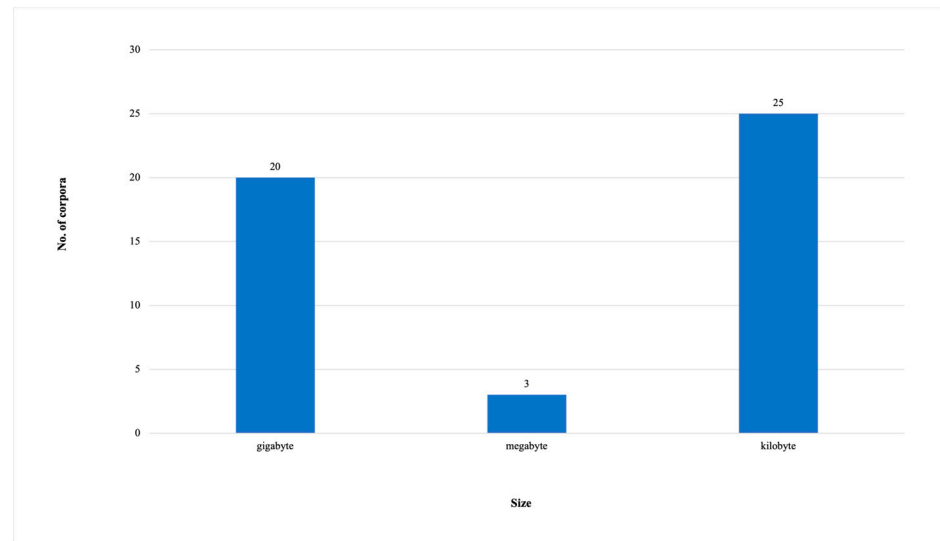


Figure 4. Distribution of corpora sizes.

Our analysis of the collected corpora led to several key findings. First, the majority of corpora were obtained from well-established repositories, with GitHub emerging as the most prominent platform. This finding highlights the significance of these repositories as valuable sources of corpus content and their crucial role in facilitating knowledge sharing and collaboration. The analysis further revealed variations in corpus size, which ranged from GB to KB and MB. These differences are indicative of diverse content types and formats, emphasizing the importance of considering storage capacity and bandwidth limitations when working with corpora of different sizes.

5. Training an Arabic Transformer Model Using the AraFast Corpus

In the present study, a large MSA corpus (AraFast) was developed to train an Arabic transformer model, and then we conducted a study to understand how well this corpus works. Four experiments were set up to identify the best version of the corpus and to determine how the size of the corpus would affect model training. Various forms of the AraFast corpus were tested, including segmented, unsegmented, web-scraped, and smaller versions.

To construct the experiments and obtain the best version of the AraFast corpus, we proposed two Arabic transformer models: AraFastQA-mini, and AraFastQA-base, which were both examined. The AraFastQA-mini model was pretrained on a 10 GB segmented corpus (which is mentioned in Section 5.3); AraFastQA-base, in contrast, was pretrained on the 112 GB segmented AraFast corpus (which is mentioned in Section 5.2). Both models were developed with enhancements to the masking language modeling objective function, building on AraBERT [61] and other transformers based on BERT versions [19]. Departing from predictions of a single token in a sentence, the proposed models predict spans (continuous tokens) within a sentence. Additionally, the models employ a dynamic masking strategy inspired by RoBERTa [62], as opposed to the static masking strategy used in BERT. The use of this novel approach aims to improve the models' ability to capture context and dependencies within continuous segments of text.

In BERT, token masking occurs before pretraining begins, and it remains constant throughout the training process. This static approach implies that the same set of tokens is consistently masked whenever a particular sentence appears in training instances. In contrast, RoBERTa employs a dynamic masking strategy, where tokens to be masked are randomly chosen anew in each training epoch. This means that each time a sentence is processed during training, a different set of tokens may be masked. This dynamic approach allows the model to learn from a diverse range of masked patterns, potentially leading

to robust learning and generalization. All of the experiments were conducted using the proposed base version of AraFastQA model.

Table 4 outlines the experimental setup with a maximum sequence length of 512, a training batch size of 256, 2.4 million training steps, and the AdamW optimizer. The optimizer was utilized for training, ensuring efficient and effective optimization of the model parameters. The experimental setup of the two models (AraFastQA mini and base) is identical. In addition, in Table 4, the AraFast mini and base configurations are provided, featuring 110 million parameters, a 100,000-token vocabulary size, 14.2 billion tokens, 534 million sentences, a 12-layer encoder architecture with 12 attention heads and 768 hidden sizes, a 1×10^{-4} learning rate, span and dynamic masked language modeling tasks, and a WordPiece tokenizer.

Table 4. AraFast mini and base configuration with their experimental setup.

Argument	Value/Method
Maximum sequence length	512
Training batch size	256
Training steps	2.4 M
Optimizer	AdamW
Parameters' number	110 M
Vocabulary size	100 K
Number of tokens/sentences	14.2 B/534 M
Encoder architecture	12 layers, 12 attention heads, and 768 hidden sizes
Learning Rate	1×10^{-4}
Pre-training task	Span and dynamic masked language modeling
Tokenizer	WordPiece

The findings of our experiments were used to address both RQ2, “Does pretraining a transformer model on a large Arabic corpus that includes web-scraped data lead to worse performance than training without web-scraped data and noise?”, and RQ3, “Does pretraining a transformer model on a segmented Arabic corpus lead to better performance than training the model on an unsegmented Arabic corpus?”. Further details about the experiments are provided below.

5.1. Web-Scraped AraFast Corpus

In the NLP domain, training language models on Arabic text, particularly data acquired through web scraping, involves significant challenges related to the nature and cleanliness of data. Arabic, characterized by its rich morphology and complex script, presents unique challenges in NLP. Factors such as the use of dialects, the absence of diacritics, and variations in encoding can lead to inconsistencies in the text data, impacting training stability.

The following subsection is a response to RQ2, “Does pretraining a transformer model on a large Arabic corpus that includes web-scraped data lead to worse performance than training without web-scraped data and noise?”. Therefore, to investigate the efficiency of the Arabic web-scraping corpus, a proposed transformer model was trained on a dataset of Arabic text obtained through web scraping, specifically common crawl (CC) data [40]. The size of the Arabic corpus containing CC data was roughly 700 GB. Notably, when training the proposed model on a larger corpus of CC data, an exploding loss curve was observed, indicating that the model was unable to learn. Figure 5 shows the exploding gradients for the model, particularly the epochs where the training data included high levels of noise. This was evident in the sudden spikes in the loss curve. The loss value of training the model on Arabic text with noise was roughly 131.3 at step 700 K. This value indicates that the model could not correctly predict the masked tokens.

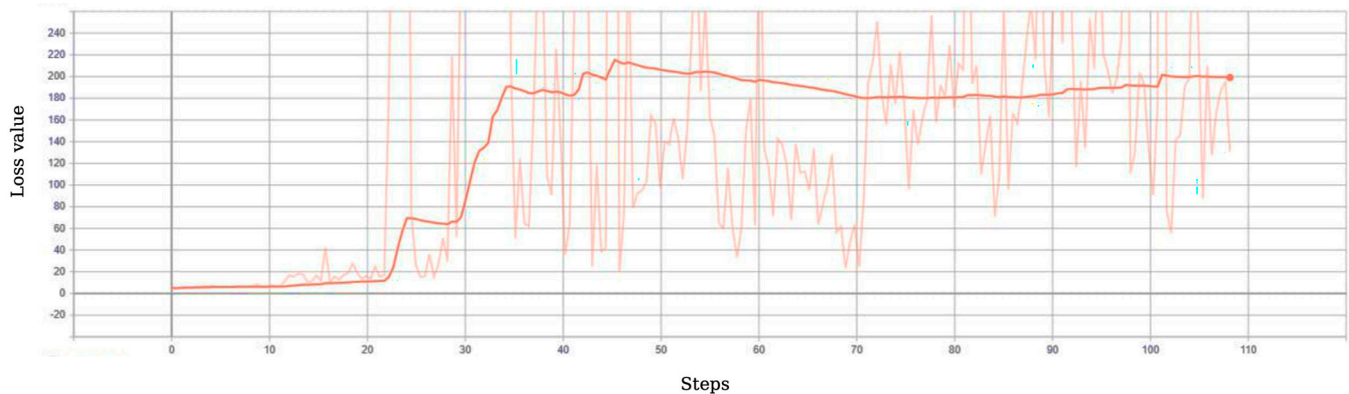


Figure 5. Training loss of the model when training on web scraping data.

To investigate the reasons for the noise, the Arabic corpus was examined and cleaned several times. It was found that the main problem was in the method of scraping data from the web. The size of the CC data was about 457 GB, and many issues affecting this data could not be fixed. For example, many words were not meaningful due to repeated letters and concatenated words, for example, “هع، ااااا، ككككك، الربعا، الخميس، الجمعة السبت”، which means “ha, aaaaa, kkkkkk, wednesdaythursdayfridaysaturday” in English abbreviations such as ﷺ, which means “peace be upon him” in English and short letters. The high frequency of such words led to the created vocabulary list of 700 GB in size being unclean, featuring a lot of meaningless tokens (Figure 6). All of these tokens were found in the original files of text, even though they were preprocessed many times, indicating that there was a problem in the process of extracting the data from the web. In tokenization, the ## symbol indicates that a token is a continuation of a word that started with a previous token. An example of concatenating words is “المجتمعمساعدةالعيدانتبهطباعة” which equivalent to “communityhelpEidyouobeyprint” in English. To overcome this and obtain a clean and meaningful corpus, the CC files were removed, as they were disqualified for the Arabic language models.

جارية أحدث
#الأنوار
#التيمنية الأحدث
#المشاركة
#بنابو ابة
#جديبو اب
#إبحار
#و النجمال
#شخصية ادخل
#المجتمع
#تجرب عظماء
#مما عة اتمل
#التعاقلة
#عشو النية المشاركة
#تجربو الأحدث
#جديبو ابالتعاقلة
#المو اضعاً بجديبو ابالتعاقلة
#المجتمعمساعدة العيد ان
#الأساسية إبحار
#الأساسية إبحار و المو اضعاً بجديبو ابالتعاقلة
#أعمالالمعاينةلنا ربيع
#شخصية ادخلأش
#المجتمعمساعدة العيد التجرب عظماء
#رحلةتعليقات
#ومشجوم
#و نواس
#بنابو
#عيد الن

جارية أحدث
#الأنوار
#التيمنية الأحدث
#المشاركة
#بنابو ابة
#جديبو اب
#إبحار
#و النجمال
#شخصية ادخل
#المجتمع
#تجرب عظماء
#مما عة اتمل
#التعاقلة
#عشو النية المشاركة
#تجربو الأحدث
#جديبو ابالتعاقلة
#المو اضعاً بجديبو ابالتعاقلة
#المجتمعمساعدة العيد ان
#الأساسية إبحار
#الأساسية إبحار و المو اضعاً بجديبو ابالتعاقلة
#أعمالالمعاينةلنا ربيع
#شخصية ادخلأش
#المجتمعمساعدة العيد التجرب عظماء
#رحلةتعليقات
#ومشجوم
#و نواس
#بنابو
#عيد الن

جارية أحدث
#الأنوار
#التيمنية الأحدث
#المشاركة
#بنابو ابة
#جديبو اب
#إبحار
#و النجمال
#شخصية ادخل
#المجتمع
#تجرب عظماء
#مما عة اتمل
#التعاقلة
#عشو النية المشاركة
#تجربو الأحدث
#جديبو ابالتعاقلة
#المو اضعاً بجديبو ابالتعاقلة
#المجتمعمساعدة العيد ان
#الأساسية إبحار
#الأساسية إبحار و المو اضعاً بجديبو ابالتعاقلة
#أعمالالمعاينةلنا ربيع
#شخصية ادخلأش
#المجتمعمساعدة العيد التجرب عظماء
#رحلةتعليقات
#ومشجوم
#و نواس
#بنابو
#عيد الن

Figure 6. Example of concatenation and repeated character problems in the vocabulary list.

The exploding gradient phenomenon during the training phase of the language model was a prominent issue encountered due to the noisy nature of web-scraped data [63]. The process of training language models on Arabic text sourced from the web entailed handling various forms of noise, such as inconsistent spellings, colloquial language variations, and a variety of dialects. Noise can worsen the problem of exploding gradients, a situation where the gradients of the loss value of a model become excessively large, impeding the training and learning process [64]. Exploding gradients occur when large error gradients accumulate, causing vast updates to the neural network weights during training. This process often leads to an unstable network, and the model fails to converge or produces erratic predictions.

Training deep learning (DL) models on clean Arabic text is crucial for achieving high accuracy and efficiency in NLP tasks. Arabic poses unique challenges in text processing due to its complex morphology and script. Clean and well-structured data ensure that nuances such as diacritics, word segmentation, and context are accurately captured [65]. This is particularly important for tasks such as text classification, machine translation, QA, text generation, and sentiment analysis, where the quality of input data directly affects the model's performance. Moreover, clean data aid in avoiding the common pitfalls of training on noisy data, such as overfitting to irrelevant features or misinterpreting the syntax and semantics of the language [66]. Therefore, investing time and resources in preprocessing and cleaning Arabic datasets is essential for developing robust and effective NLP models tailored to the Arabic language.

5.2. Segmented and Unsegmented Versions of the AraFast Corpus

The findings presented in the following subsection respond to RQ3, “Does pretraining a transformer model on a segmented Arabic corpus lead to better performance than training the model on an unsegmented Arabic corpus?”

In order to answer RQ3, in the following study, the web-scraped data were removed from the AraFast corpus to make improvements for training the model. The size of the updated corpus was roughly 112 GB. This version of the corpus was used to determine how segmenting the text would affect training performance.

The Arabic language is characterized by lexical sparsity due to its complicated concatenative system [67]. This characteristic results in words taking various forms while retaining the same meaning. For example, the word “فسيكفيهم”, which translates to “will suffice you against them” in English, is composed of multiple parts within a single word. In Arabic morphology, the definite article “ال—Al”, equivalent to “the” in English, is always prefixed to other words, but is not an inherent part of those words [61]. Consequently, direct tokenization of the corpus results in tokens appearing twice: once with “Al-”, and once without. For instance, both “مدينة—madinah” and “المدينة—Almadina”, “city” in English, must be included in the vocabulary, leading to significant redundancy. To mitigate this issue, we first segmented the words using Farasa [68] into stems, prefixes, and suffixes. For example, “المدرسة—Almadrasah”, “school” in English, is segmented into “مدرسة—Al + madrasa + h”. Subsequently, we employed a WordPiece tokenizer to generate the vocabulary list.

Initially, the proposed transformer model was trained on Arabic text without a segmenter such as Farasa, and the results were notably poor (Figure 7). This was evidenced by a high loss value of approximately 1.3 at 829 K steps out of 2.4 M. Furthermore, the loss value increased as the number of steps increased. The inefficacy of this approach was primarily attributed to the inherent complexities of Arabic, a language characterized by rich morphological structures and a plethora of syntactic intricacies [67]. The absence of vowels in written text, the prevalence of morphology, the use of diacritics, and dots on letters such as “ي” Hamzah “significantly complicate Arabic language processing tasks [69]. Figure 6 shows the loss curve of training the proposed model with the segmented Arabic corpus. Without segmentation, the model struggled to differentiate between root words and their bound morphemes, leading to superficial understanding and processing of the language, as reflected by the high loss value [61].

In contrast, the incorporation of the Farasa segmenter, which is known for its proficiency in Arabic morphological analysis, drastically improved the model's performance [67]. Farasa efficiently decomposes words into their base forms and morphemes, enabling models to capture linguistic subtleties with greater precision [68]. This strategic enhancement led to a more stabilized and effective training process, as indicated by the reduced loss value of around 0.1 when training the proposed model on the segmented Arabic corpus. Figure 8 depicts the loss curve when training the model on segmented text. As shown, the value reached around 0.16 at step 1.65 M, a small value compared to the value obtained when training the model on unsegmented Arabic text. The significant decrease in the

loss value demonstrated the improved capability of the model to grasp and internalize the linguistic nuances of Arabic, thus facilitating a more accurate and efficient learning curve. This decrease indicates that the model was able to learn more as the number of steps increased. Figure 9 represents a sample of segmented text. It is a part of an Arabic text discussing ignorance and how to overcome it. The red dotted lines, as a caution, are because the words are segmented using the Farasa tool, which segments text into parts using the (+) symbol to indicate if the part follows or is followed by another part of the same word.

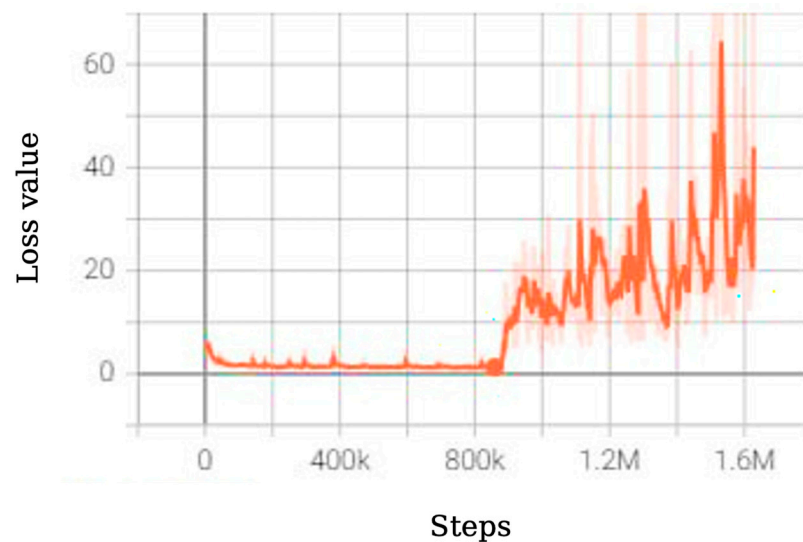


Figure 7. Training loss of the model without segmentation.

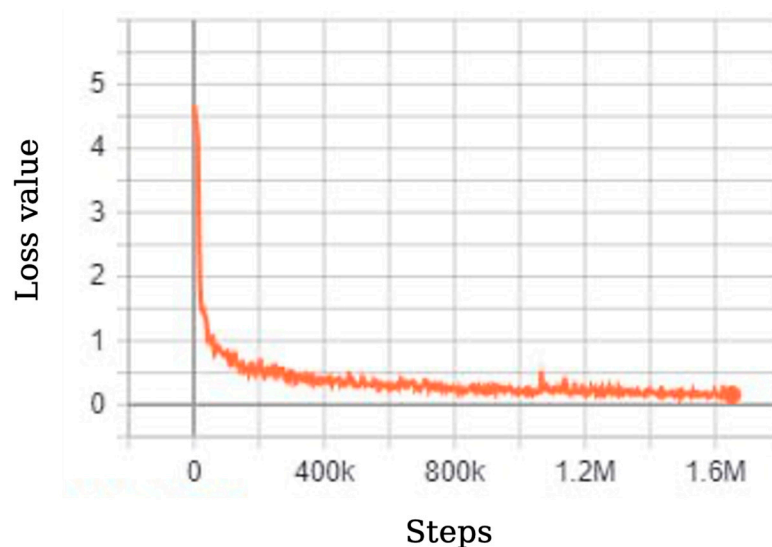


Figure 8. Training loss of the base model with segmentation.

A loss value that decreases during training is preferable. In machine learning (ML) and DL, especially when training language models such as transformers, the loss value is a measure of how well the model is performing with respect to its training objective. A low loss value indicates that the predictions of the model are close to the actual outcomes, highlighting the accuracy and effectiveness of the model [70]. Therefore, a decreasing loss value during training is a positive sign, as it means that the model is learning and that its ability to make predictions or generate text accurately is improving over time.

[illegible]

Figure 9. Farasa sample for the Arabic corpus used.

The findings of this comparative study underscore the critical role of segmentation in Arabic language modeling and the effectiveness of Farasa in optimizing model performance. The contrast between the performance levels of the model with and without Farasa emphasizes the need to incorporate advanced linguistic tools for languages with complex morphological structures such as Arabic.

5.3. Mini AraFast Corpus

To investigate the effect of corpus size on the language model’s performance, an experiment was conducted to train the transformer model on roughly 10 GB of data or less. This mini corpus was extracted from the large AraFast corpus of 112 GB in size. In Figure 10, the training loss curve for the transformer model is shown. The loss value reached roughly 0.24 at step 2.4 M; in contrast, it reached 0.37 at step 2.3 M. This result can be attributed to the model being trained only on text from Arabic Wikipedia and Abu-Elkhair news [28], which constituted the mini version of the AraFast corpus. Moreover, there was no diversity in the domains of those texts. Due to the small data size and the limited domains, the vocabulary size was only about 10 K tokens, potentially affecting the performance of the training model.

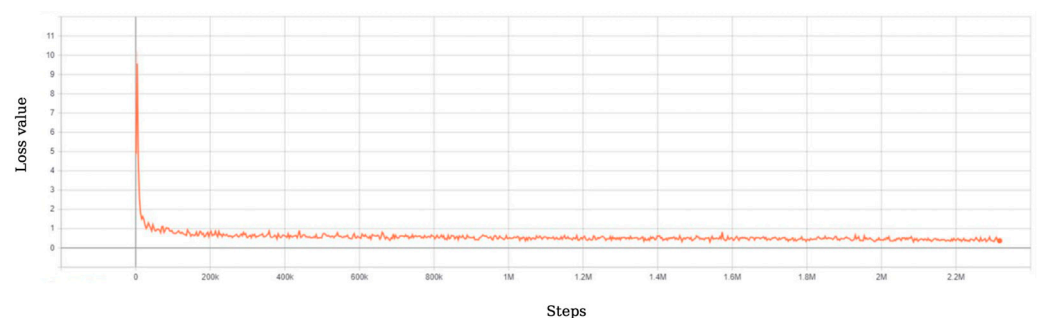


Figure 10. Training loss of the mini model with segmentation.

5.4. Results and Discussion: Training an Arabic Transformer Model

In the following section, the key findings of the four experiments conducted using the AraFast corpus are discussed. To illustrate the general behavior of the transformer model, seven representative steps of the training process with extraction of their loss values were selected for all four versions of the AraFast corpus. Choosing these specific steps allowed for a focused analysis of the model’s performance and the impact of different corpus configurations on the training outcomes.

In Table 5 and Figure 11, the training loss values for the different versions of the proposed model are presented, namely, the mini model, the base model with segmentation (seg.), the base model without segmentation, and the base model trained with noise, which included data obtained following web scraping. The mini version exhibited a consistent and steady decrease in loss as the number of steps increased, which suggests that effective learning and generalization took place. The lowest loss value was seen at 2 M steps, indicating that longer training leads to better performance with this model. The optimal value of loss should be between 0 and 1, and should be less than the value at the start of training. As the loss value decreases, the model's learning improves. For instance, the loss values of the mini and base models with segmentation, as shown in Table 5, dramatically decreased as the model was trained on a high-quality corpus excluding noisy data.

Table 5. Training loss values for the AraFast QA experimental models.

Step Number	Mini	Base with Seg.	Base without Seg.	Base with Noise
10 K	2.7	4.6	4.68	300
50 K	0.91	1.22	2.3	231
100 K	0.83	0.55	1.46	6.47
500 K	0.77	0.28	0.9	1.17
1 M	0.7	0.2	17.3	890
1.5 M	0.43	0.15	23.07	760
2 M	0.24	0.1	20.9	870

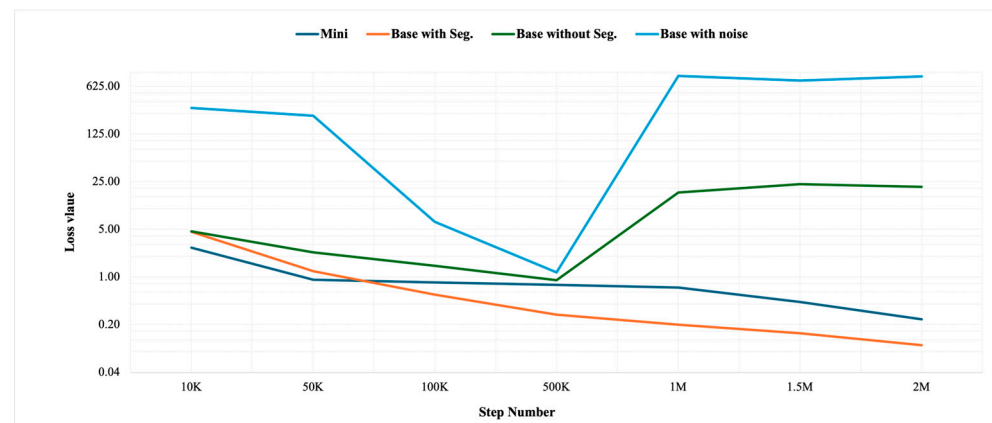


Figure 11. Loss curves when training the model using the different versions of AraFast.

The base model with segmentation also showed a decreasing trend in loss, with significant improvements as training progressed. Segmentation contributed positively to the learning process, as evidenced by the low loss values compared to the base model without segmentation. In contrast, the base model without segmentation initially paralleled the base model with segmentation in terms of loss reduction. However, there was a dramatic increase in loss after 500 K steps, peaking at 1.5 M steps. This finding indicates that the problem of exploding loss was due to the representation of noisy data.

Finally, the loss values for the base model with noise were significantly higher than those of the other models, particularly in the initial stages. A decrease was observed after 100 K steps, followed by irregular increases and decreases. The high loss values may have been due to the model struggling to generalize the noisy data, although performance seemed to improve slightly over time before rising again after 1 M steps. The above findings indicate that the type of data used was not suitable for training, as it contained a considerable amount of noise and meaningless data.

In sum, the training loss values of the various transformer models with Arabic text confirmed the significance of several NLP and DL factors. First, the results suggest that training a model on a large text corpus leads to better performance, as the mini and base

models demonstrated continuous improvements in loss reduction with large corpus sizes. This finding further implies that a large corpus can provide diverse examples from which a model can learn, enhancing its ability to generalize and understand the nuances of the Arabic language. Second, the application of segmentation in the base model led to improved outcomes compared to the base model without segmentation. This finding indicates that segmentation, which involves breaking text into meaningful subunits or tokens, is a beneficial preprocessing step for Arabic text. It enables models to better capture the structure of the language and understand the context of words and phrases, which is particularly important given the morphological complexity of Arabic. Third, the high loss values of the base model trained with noise highlight the importance of using clean text for training. The presence of noise (in the form of irrelevant information, errors, or inconsistencies in the data) can significantly obstruct the learning process, making it difficult for a model to identify patterns and learn effectively. Thus, ensuring the quality of data via cleaning and preprocessing is essential for the successful training of language models on Arabic text.

The above results indicate the necessity of employing the following features: a large, well-segmented, clean corpus for the development of robust and efficient models for Arabic NLP tasks. These features are crucial for advancing the NLP field and achieving high levels of accuracy in various applications, such as question-answering, machine translation, and text summarization systems.

6. Comparison of the AraFastQA Model Versions and the ARBERT Model

6.1. AraFastQA-Mini and AraFastQA-Base

In the following section, the fourth research question (RQ4), “How does pretraining a transformer model on a large Arabic corpus impact its accuracy in NLP tasks, specifically Arabic QA tasks?”, is answered. To explore the above question, a case study using the AraFast corpus and examining the effects of employing large and small Arabic corpora is presented.

The significance of utilizing extensive Arabic corpora for training transformer models has consistently been emphasized in prior research. The authors of several papers have highlighted the effectiveness of large-scale Arabic language models and their applications across various NLP tasks. For instance, Alrowili and Vijay-Shanker [71] discussed, in their study, the development of a large Arabic language model using Funnel Transformer architecture combined with the ELECTRA objective. Their work underscores the importance of using a substantial corpus to train models to efficiently handle the intricacies of the Arabic language. In another study, Antoun et al. [72] introduced AraGPT2, a pre-trained transformer model for Arabic language generation. The researchers emphasized the necessity of a large diverse Arabic corpus for pretraining to enhance the model's performance in generation tasks. Moreover, Abdul-Mageed et al. [73] introduced ARBERT and MARBERT in their study, deep bidirectional transformers for Arabic, and highlighted how large-scale Arabic corpora enable these models to better capture the nuances of the language, leading to improvements in various downstream tasks. Furthermore, Chouikhi and Alsuhaibani [74] focused on using deep transformer language models for Arabic text summarization. Their study underscores the importance of using extensive corpora in training models to effectively summarize Arabic texts, highlighting the vital role of data volume in model efficacy. In another study, Abdelali et al. [75] explored the interpretation ability of Arabic transformer models. They discussed the impacts of large-scale Arabic data on the interpretability and effectiveness of transformer models when processing data in the Arabic language.

To demonstrate the impact of using large and small Arabic corpora, we examined the two proposed Arabic transformer models: AraFastQA-mini, and AraFastQA-base. Moreover, a few-shot learning (FSL) approach was adopted to assess the proposed models' performance in an Arabic QA task. FSL focuses on enabling models to perform language tasks with minimal training data, a crucial consideration in NLP, since the process of

obtaining large, annotated datasets can be expensive and time consuming. For instance, by exploring low-shot learning in NLP, Xia et al. [76] addressed, in their study, the challenges of training models with a limited amount of data. Similarly, Ding and Ye [77] discussed, using bidirectional prompt learning in NLP, a few-shot tasks in their study, highlighting the models' adaptability to effective learning from a limited number of examples. Pasunuru et al. [78] investigated continual few-shot learning for text classification, and highlighted FSL's potential in dynamically evolving NLP applications. The findings of the above studies underscore the growing interest in techniques that reduce dependency on large datasets, while facilitating effective performance in various NLP tasks.

In Table 6, the results of tuning the two proposed models using the FSL approach are presented. The F1 and exact match (EM) scores represent the models' evaluation results in the Arabic Typologically Diverse Question Answering (TyDi QA) dataset, a benchmark for information-seeking question answering in typologically diverse languages, including Arabic [79]. In Table 6, Figures 12 and 13, the performance comparison results of the AraFastQA-mini and AraFastQA-base models in the Arabic QA task with the Arabic TyDi QA dataset are presented. The evaluation metrics F1 score and EM are standard metrics for assessing QA systems.

Table 6. Evaluation results of the proposed models and ARBERT with the TyDi QA dataset.

Number of Shots	Metrics	AraFastQA-Mini	AraFastQA-Base	ARBERT	ArabicTransformer-Base
16 shots	F1	18.79	26.06	8.80	7.40
	EM	3.47	12.24	0.10	0.02
32 shots	F1	22.69	30.66	10.02	7.59
	EM	8.46	15.41	0.32	0.32
64 shots	F1	23.03	34.25	16.46	8.36
	EM	10.2	19.65	3.04	0.21
128 shots	F1	29.89	48.36	30.24	10.69
	EM	13.87	26.38	15.74	0.86
256 shots	F1	36.01	59.51	40.45	26.55
	EM	12.06	38.01	22.14	15.20
512 shots	F1	44.5	67.23	57.88	43.85
	EM	22.18	46.14	36.59	24.97
1024 shots	F1	58.9	73.23	63.42	61.72
	EM	33.62	54.07	40.71	37.35
Full shot	F1	61.68	81.47	71.82	74.88
	EM	45.05	66.12	49.83	51.68

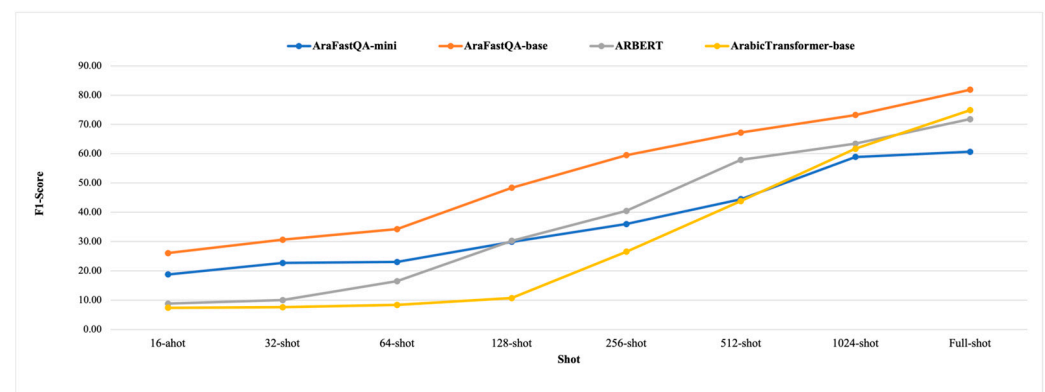


Figure 12. F1 scores of the models.

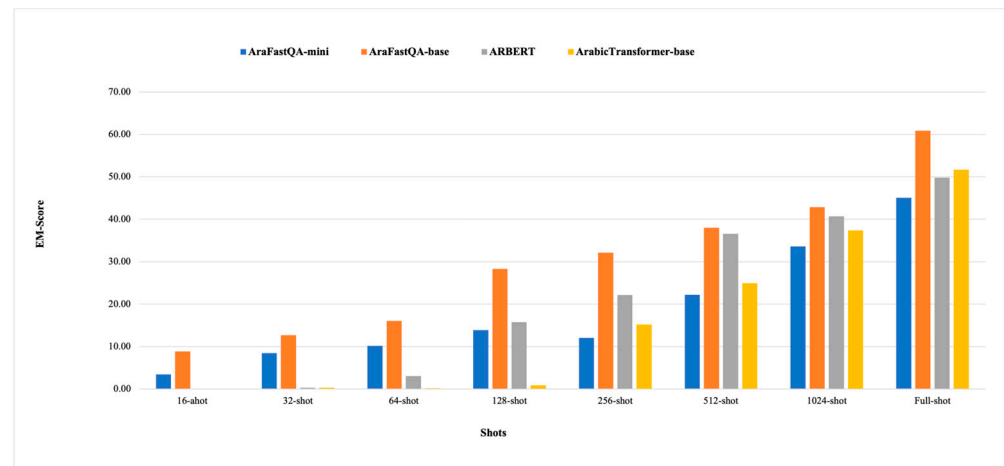


Figure 13. EM scores of the models.

In Table 6, a comprehensive overview of the F1 and EM scores for both models across different shot settings is shown, ranging from 16 shots to a full shot. The term “shot” refers to the number of training examples the model has been exposed to, with “full shot” indicating that training was conducted with the entire dataset.

As shown in Figures 12 and 13, AraFastQA-base consistently outperformed AraFastQA-mini across all shot settings for both F1 and EM metrics. This trend suggests that the base model demonstrated stronger Arabic language understanding and processing capabilities in the context of QA tasks, which can be attributed to the model’s pretraining on a large Arabic corpus. The incremental improvements observed with increasing shot sizes for both models indicate that the additional training examples contributed to each model’s ability to generalize and respond accurately to queries. This trend was particularly evident in the full-shot setting, as both models exhibited their highest performance. Notably, the base model consistently maintained a larger margin of improvement over the mini model, which was pretrained on only 10 GB of data.

The impact of corpus size on model performance was most noticeable in the full-shot setting, with the AraFastQA-base model significantly outperforming AraFastQA-mini. AraFastQA-base achieved an F1 score of 81.46 and an EM score of 66.12; in contrast, AraFastQA-mini’s F1 score was 61.68, and its EM score was 45.05. The substantial differences highlight the significant advantage of using a large Arabic corpus for comprehensive tasks such as QA.

The AraFastQA-base model also demonstrated better learning and adaptation capabilities, as evidenced by its consistent performance across different shot sizes. In the 64-shot setting, for example, the F1 and EM scores of the base model surpassed those of the mini model by 11.22 points and 9.45 percentages, respectively. The base model not only began with a higher F1 score, but also showed a steeper improvement curve as more shots were added.

The performance differences between the AraFastQA-mini and AraFastQA-base models can be attributed to the significant difference in the pretraining data volume. The base model’s superior accuracy was likely a result of pretraining on a large corpus of 112 GB, as compared to the mini model’s 10 GB size. In NLP, the breadth and diversity of the pretraining corpus are crucial for a model to capture a wide range of linguistic patterns and contexts. The base model’s exposure to a large Arabic corpus reflected a rich linguistic experience that equipped the model with a broad knowledge base for extracting answers and understanding complex language structures.

Overall, AraFastQA-base’s superior performance indicates its potential for practical applications requiring a high accuracy in understanding Arabic. However, this high accuracy may come at the cost of increased computational resources and inference times, which are typical trade-offs when deploying large, capable models in real-world scenarios.

The collected Arabic corpora are significant for NLP tasks, such as NER and machine translation. These corpora provide valuable training data for NER models, enabling the accurate extraction and classification of named entities in Arabic texts. The availability of a diverse Arabic corpus can also improve the accuracy and fluency of machine translation systems, contributing to cross-lingual communication. Furthermore, the collected Arabic corpora can support cross-linguistic studies and comparative linguistic analyses. By comparing the Arabic corpus with corpora from other languages, researchers can gain insights into language structure, evolution, and linguistic phenomena, thereby advancing linguistic research and understanding of the Arabic language.

6.2. Evaluating AraFastQA-Base against Other Models

The proposed model AraFastQA-base was additionally compared with the ARBERT model [73]. AraFastQA-base is a large-scale PML model designed specifically for modern standard Arabic (MSA), and it was trained on 61 GB of SMA text with vocabulary comprising 100 K words. ARBERT was built using the same architecture as BERT-base. In Table 6, Figures 12 and 13, the F1 scores of the AraFastQA-mini, AraFastQA-base, and ARBERT models on the TyDi QA dataset across various shot counts are illustrated. AraFastQA-base consistently outperforms ARBERT, particularly as the shot size increases, indicating its superior ability to handle the TyDi QA dataset. This is because the AraFastQA-base is trained to predict spans of tokens rather than just single tokens. This span-based prediction approach allows AraFastQA to capture more context and generate more coherent and accurate answers. Additionally, AraFastQA employs a dynamic strategy in predicting span tokens, which further enhances its ability to adapt to various questions and contexts, improving its overall performance. In contrast, ARBERT's single-token prediction approach limits its contextual understanding and response accuracy.

Moreover, in Table 6, Figures 12 and 13, the performance of ArabicTransformer [71] on the TyDi QA dataset is displayed, measured using F1 and EM score across various shot counts. The ArabicTransformer model, an Arabic language model employing the Funnel Transformer architecture and the ELECTRA training objective, was pre-trained on a 44 GB Arabic corpus with a vocabulary size of 50,000 tokens. This model is designed to operate with lower computational and resource demands compared to models such as AraBERT and AraELECTRA.

AraFastQA-base consistently outperforms ArabicTransformer-base, demonstrating its superior ability to handle the dataset, especially as the shot count increases. AraFastQA-base emerged as the best-performing model, consistently achieving the highest scores across all shot counts, highlighting its superior performance with larger datasets. Following AraFastQA-base, the ArabicTransformer-base model shows better results, particularly as the shot count increases, though it still trails behind AraFastQA-base. ARBERT ranks third, displaying steady improvement with increased shot counts, but ultimately being outperformed by both the AraFastQA-base and ArabicTransformer-base. Lastly, AraFastQA-mini is particularly effective in few-shot scenarios, outperforming ARBERT and ArabicTransformer-base at lower shot counts (16 to 128) due to its tailored design for limited training data. However, it lags behind the other models as the shot count increases.

One of the key reasons AraFastQA-base outperforms ARBERT and ArabicTransformer is that it is trained on a larger corpus, has a better understanding of the morphology of the Arabic language, and can generate more accurate responses. Therefore, while AraFastQA-base demonstrates significant improvements compared to ARBERT, AraFastQA-mini does not show the same level of performance, likely due to it being trained on a smaller corpus.

7. Challenges and Limitations

In the following section, the challenges encountered when collecting and preprocessing the Arabic corpora are outlined, and potential solutions are proposed. The primary challenge was consolidating all available corpora into a single location on the Google Drive storage platform, a task that spanned approximately six months. The duration of

this process was influenced by various factors, especially network disconnections, which impacted the download and upload processes for large text files. This problem was exacerbated by the need to restart downloads in the case of network disruptions. To mitigate this issue and effectively overcome the challenges associated with network interruptions, the BigTextFileSplitter tool was employed to break down large files into 100 MB segments during the upload and corpus creation processes.

Another significant challenge revolved around the availability of data. Despite our comprehensive efforts to include all freely accessible Arabic corpora on the Internet, the accessibility and diversity of the Arabic corpora remained limited. Notably, there is a scarcity of publicly available large-scale Arabic text collections, especially when compared to languages such as English [80], and many publicly available Arabic corpora are either unclear or exist in formats such as JSON, CSV, and HTML rather than plain text. The rich dialectal variations in Arabic further complicated the task of gathering representative and balanced MSA corpora.

The limited availability of Arabic books in text format posed another challenge. The Hindawi organization's books were a notable exception, as these were available in PDF and EBOOK formats. Moreover, Arabic corpora lack coverage in specialized domains and specific subject areas, hindering the development of domain-specific NLP applications. Many corpora focus on general language usage, but do not adequately cover the medical, legal, and scientific domains or the technical literature, which poses challenges for research in specialized areas.

Addressing the above challenges requires collaborative efforts to collect and share diverse, cleaned, domain-specific Arabic corpora. Moreover, investing in linguistic resources and tools specific to Arabic is crucial to overcoming these limitations and driving advancements in the area of Arabic language processing. This collaborative and resource-centric approach is pivotal for the development of comprehensive and representative corpora, and for advancements in various domains of Arabic research and applications.

8. Conclusions

The present study provides a comprehensive understanding of freely available MSA corpora. A key contribution of this work is the introduction of AraFast, an Arabic corpus designed to facilitate language modeling and NLP tasks. This readily available, preprocessed corpus can expedite the development of language models and enhance the generation of contextually relevant Arabic text. The potential applications of AraFast span various domains, including chatbots, QA systems, summarization tools, and Arabic text generation. Furthermore, the impact of segmentation and data quality on the AraFast corpus was evaluated. The study findings also highlight the challenges posed by web-scraped data, which often introduce noise and inconsistencies, affecting the model's efficiency. These evaluations not only validate the robustness of AraFast, but can also guide future corpus development for Arabic NLP tasks.

For future research, it is suggested to further develop the AraFast corpus by collecting and preprocessing data on Arabic dialects (AD), including them in the corpus. When the AraFast corpus contains both MSA and AD, the Arabic PLMs trained on the developed corpus will be able to handle other NLP tasks for AD as well. Moreover, the proposed AraFastQA-base model will be compared with a greater number of existing Arabic PLMs and evaluated on additional Arabic QA datasets.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: A.A., F.A. and M.S.; data collection: A.A.; analysis and interpretation of results: A.A., F.A. and M.S.; draft manuscript preparation: A.A., F.A. and M.S. All authors reviewed the results. 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: The data that support the findings of this study are available from the corresponding author, Asmaa Alrayzah, upon reasonable request.

Acknowledgments: The authors express their gratitude towards the anonymous reviewers, whose comments greatly contributed to improving this paper.

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

References

- Alqurashi, S.; Alhindi, A.; Alanazi, E. Large Arabic Twitter Dataset on COVID-19. *arXiv* **2020**, arXiv:2004.04315.
- Black, W.; Elkateb, S.; Rodriguez, H.; Alkhalifa, M.; Vossen, P.; Pease, A.; Fellbaum, C. Introducing the Arabic WordNet project. In Proceedings of the GWC 2006: 3rd International Global WordNet Conference, Proceedings, Seogwipo, Republic of Korea, 22–26 January 2006; pp. 295–299.
- Ahmed, A.; Ali, N.; Alzubaidi, M.; Zaghouni, W.; Abd-alrazaq, A.A.; Househ, M. Freely Available Arabic Corpora: A Scoping Review. *Comput. Methods Programs Biomed. Updat.* **2022**, *2*, 100049. [\[CrossRef\]](#)
- Alrayzah, A.; Alsolami, F.; Saleh, M. Challenges and opportunities for Arabic question-answering systems: Current techniques and future directions. *PeerJ Comput. Sci.* **2023**, *9*, e1633. [\[CrossRef\]](#)
- Alexopoulou, T.; Michel, M.; Murakami, A.; Meurers, D. Task Effects on Linguistic Complexity and Accuracy: A Large-Scale Learner Corpus Analysis Employing Natural Language Processing Techniques. *Lang. Learn.* **2017**, *67*, 180–208. [\[CrossRef\]](#)
- Abbas, M.; Smaili, K.; Berkani, D. Evaluation of topic identification methods on arabic corpora. *J. Digit. Inf. Manag.* **2011**, *9*, 185–192.
- Rushdi-Saleh, M.; Martín-Valdivia, M.T.; Ureña-López, L.A.; Perea-Ortega, J.M. OCA: Opinion Corpus for Arabic. *J. Am. Soc. Inf. Sci. Technol.* **2011**, *64*, 1852–1863.
- Ali, A.R. A Large and Diverse Arabic Corpus for Language Modeling. *Procedia Comput. Sci.* **2023**, *225*, 12–21. [\[CrossRef\]](#)
- Abdelali, A.; Mubarak, H.; Chowdhury, S.A.; Hasanain, M.; Mousi, B.; Boughorbel, S.; El Kheir, Y.; Izham, D.; Dalvi, F.; Hawasly, M.; et al. Benchmarking Arabic AI with Large Language Models. *arXiv* **2023**, arXiv:2305.14982.
- Pearce, K.; Zhan, T.; Komanduri, A.; Zhan, J. A Comparative Study of Transformer-Based Language Models on Extractive Question Answering. *arXiv* **2021**, arXiv:2110.03142.
- Alyafeai, Z.; Masoud, M.; Ghaleb, M.; Al-shaibani, M.S. Masader: Metadata Sourcing for Arabic Text and Speech Data Resources. In Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), Marseille, France, 20–25 June 2022; European Language Resources Association (ELRA): Paris, France, 2022; pp. 6340–6351.
- Keele, S. *Guidelines for Performing Systematic Literature Reviews in Software Engineering*; EBSE Technical Report; University of Durham: Durham, UK, 2007.
- Rowley, J.; Slack, F. Conducting a literature review. *Manag. Res. News* **2004**, *27*, 31–39. [\[CrossRef\]](#)
- Suárez, P.J.O.; Romary, L.; Sagot, B. A Monolingual Approach to Contextualized Word Embeddings for Mid-Resource Languages. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Seattle, WA, USA, 5–10 July 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 1703–1714. [\[CrossRef\]](#)
- Eldesouki, M.I.; Arafa, W.; Darwish, K.; Gheith, M. Using Wikipedia for Retrieving Arabic Documents. In Proceedings of the Arabic Language Technology International Conference (ALTIC) 2011, Alexandria, Egypt, 9–10 October 2021.
- Abdul-Mageed, M.M.; Herring, S.C. Arabic and English news coverage on Al-Jazeera.net. *Proc. Cult. Attitudes Towards Technol. Commun.* **2008**, *2008*, 271–285.
- Einea, O.; Elnagar, A.; Al Debsi, R. SANAD: Single-label Arabic News Articles Dataset for automatic text categorization. *Data Brief* **2019**, *25*, 104076. [\[CrossRef\]](#)
- Alrabia, M.; Atwell, E.; Al-Salman, A.; Alhelewh, N. KSUCCA: A Key To Exploring Arabic Historical Linguistics. *Int. J. Comput. Linguist.* **2014**, *5*, 27–36.
- Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the NAACL HLT 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MI, USA, 2–7 June 2019; pp. 4171–4186.
- Al-Dulaimi, A.H. Ultimate Arabic News Dataset. *Mendeley Data*. 21 September 2022. Available online: <https://www.kaggle.com/datasets/asmaaabdelwahab/arabic-news-dataset> (accessed on 25 July 2023).
- Abbas, M.; Smaili, K. Comparison of topic identification methods for Arabic language. In Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP, Borovets, Bulgaria, 24 September 2005; pp. 14–17.
- Zerrouki, T.; Balla, A. Tashkeela: Novel corpus of Arabic vocalized texts, data for auto-diacritization systems. *Data Brief* **2017**, *11*, 147–151. [\[CrossRef\]](#)
- Chouigui, A.; Ben Khiroun, O.; Elayeb, B. ANT corpus: An Arabic news text collection for textual classification. In Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications, AICCSA, Aqaba, Jordan, 28 October–1 November 2018; pp. 135–142. [\[CrossRef\]](#)

24. Jbene, M.; Tigani, S.; Saadane, R.; Chehri, A. A Moroccan News Articles Dataset (MNAD) for Arabic Text Categorization. In Proceedings of the 2021 International Conference on Decision Aid Sciences and Application, DASA, Sakheer, Bahrain, 7–8 December 2021; pp. 350–353. [\[CrossRef\]](#)
25. Ruder, S.; Sogaard, A.; Vulic, I. Unsupervised cross-lingual representation learning. In Proceedings of the ACL 2019 57th Annual Meeting of the Association for Computational Linguistics, Tutorial Abstracts, Florence, Italy, 28 July–2 August 2019; pp. 31–38. [\[CrossRef\]](#)
26. Al-Abdallah, R.Z.; Al-Taani, A.T. Arabic Single-Document Text Summarization Using Particle Swarm Optimization Algorithm. *Procedia Comput. Sci.* **2017**, *117*, 30–37. [\[CrossRef\]](#)
27. Mahmoud, E.-H. Arabic in Business and Management Corpora (ABMC) Dataset-NLP Hub. Metatext. Available online: [https://metatext.io/datasets/arabic-in-business-and-management-corpora-\(abmc\)](https://metatext.io/datasets/arabic-in-business-and-management-corpora-(abmc)) (accessed on 25 July 2023).
28. El-Khair, I.A. Abu el-khair corpus: A modern standard arabic corpus. *Int. J. Recent Trends Eng. Res.* **2017**, *2*, 5–13.
29. Alhagri. Saudi Newspapers Corpus Dataset-NLP Hub. Metatext. Available online: <https://metatext.io/datasets/saudi-newspapers-corpus> (accessed on 25 July 2023).
30. Tiedemann, J. Parallel data, tools and interfaces in OPUS. In Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC, Istanbul, Turkey, 21–27 May 2012; pp. 2214–2218.
31. Zeroual, I.; Goldhahn, D.; Eckart, T.; Lakhouaja, A. OSIAN: Open source international arabic news corpus—Preparation and integration into the clarin-infrastructure. In Proceedings of the ACL 2019 4th Arabic Natural Language Processing Workshop, WANLP 2019, Florence, Italy, 1 August 2019; pp. 175–182. [\[CrossRef\]](#)
32. Al-Thubaity, A.; Khan, M.; Al-Mazrua, M.; Al-Mousa, M. New language resources for arabic: Corpus containing more than two million words and a corpus processing tool. In Proceedings of the 2013 International Conference on Asian Language Processing, IALP, Urumqi, China, 17–19 August 2013; pp. 67–70. [\[CrossRef\]](#)
33. Mohammad, T. GitHub—Mohataher/arabic_big_corpus: Text File Containing Big ARABIC Corpus. GitHub. Available online: https://github.com/mohataher/arabic_big_corpus (accessed on 26 July 2023).
34. Helmy, M.; Basaldella, M.; Maddalena, E.; Mizzaro, S.; Demartini, G. Towards building a standard dataset for Arabic keyphrase extraction evaluation. In Proceedings of the 2016 International Conference on Asian Language Processing (IALP), Tainan, Taiwan, 21–23 November 2016; IEEE; pp. 26–29. [\[CrossRef\]](#)
35. Motaz, S. GitHub—Motazsaad/bbc-Crawler: Crawl News Documents from BBC Arabic. GitHub. Available online: <https://github.com/motazsaad/bbc-crawler> (accessed on 26 July 2023).
36. Motaz, S. GitHub—Motazsaad/Arabic-Stories-Corpus: Arabic Stories Corpus. GitHub. Available online: <https://github.com/motazsaad/Arabic-Stories-Corpus> (accessed on 26 July 2023).
37. Motaz, S. GitHub—Motazsaad/Arabic-News: Arabic News. GitHub. Available online: <https://github.com/motazsaad/Arabic-News> (accessed on 26 July 2023).
38. Motaz, S. GitHub—Motazsaad/Tashkeela2: Arabic Vocalized Text Corpus. GitHub. Available online: <https://github.com/motazsaad/tashkeela2/tree/master> (accessed on 26 July 2023).
39. Ahmed, A. [Corpora-List] Arabic Corpora Resource Now Available. Available online: <https://mailman.uib.no/public/corpora/2011-January/012055.html> (accessed on 26 July 2023).
40. Buck, C.; Heafield, K.; Van Ooyen, B. N-gram Counts and Language Models from the Common Crawl. In *European Language Resources Association (ELRA)*; United Nations Educational, Scientific and Cultural Organization: Granada, Spain, 2014; pp. 3579–3584.
41. Belinkov, Y.; Magidow, A.; Romanov, M.; Shmidman, A.; Koppel, M. Shamela: A Large-Scale Historical Arabic Corpus. In Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH), Osaka, Japan, 11–16 December 2016.
42. Tawalbeh, S.; AL-Smadi, M. Is this sentence valid? An Arabic Dataset for Commonsense Validation. *arXiv* **2020**, arXiv:2008.10873.
43. Jansen, D.; Alcalá, A.; Guzman, F. Amara: A Sustainable, Global Solution for Accessibility, Features of the Amara Platform. In *Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2014; Volume 8516, pp. 401–411. [\[CrossRef\]](#)
44. Elmadany, A.; Mubarak, H.; Magdy, W. Arsas: An arabic speech-act and sentiment corpus of tweets. *Osact* **2018**, *3*, 20.
45. Elnagar, A.; Einea, O. BRAD 1.0: Book reviews in Arabic dataset. In Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications, AICCSA, Agadir, Morocco, 29 November–2 December 2016. [\[CrossRef\]](#)
46. El-Haj, M.; Rayson, P. OSMAN—A novel Arabic readability metric. In Proceedings of the 10th International Conference on Language Resources and Evaluation, LREC, Portorož, Slovenia, 23–28 May 2016; pp. 250–255.
47. Saad, M.; Langlois, D.; Smaili, K. Extracting Comparable Articles from Wikipedia and Measuring their Comparabilities. *Procedia Soc. Behav. Sci.* **2013**, *95*, 40–47. [\[CrossRef\]](#)
48. Nagoudi El Moatez Bellah. Arabic Corpus Download | SourceForge.net. SourceForge. Available online: <https://sourceforge.net/projects/newarabiccorpus/> (accessed on 26 July 2023).
49. Taha Zerrouki. Arabicwordcorpus—Browse Files at SourceForge.net. SourceForge. Available online: <https://sourceforge.net/projects/arabicwordcorpu/files/> (accessed on 26 July 2023).
50. Maxim Romanov. A Corpus of Arabic Literature (19–20th Centuries) for Stylometric Tests | Zenodo. Zenodo. Available online: <https://zenodo.org/record/5772261#.Y2eeoi8RrQr> (accessed on 26 July 2023).

51. Maxim, R. GitHub—OpenITI/RELEASE at v2019.1.1. GitHub. Available online: <https://github.com/OpenITI/RELEASE/tree/v2019.1.1> (accessed on 26 July 2023).
52. Abdullah, A.; Eric, A. Arabic Learner Corpus. Available online: <https://www.arabiclearnercorpus.com/> (accessed on 26 July 2023).
53. Bounhas, I.; Ben Guirat, S. KUNUZ: A multi-purpose reusable test collection for classical arabic document engineering. In Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications, AICCSA, Abu Dhabi, United Arab Emirates, 3–7 November 2019; pp. 1–8. [CrossRef]
54. El-Haj, M. Habibi—A multi dialect multi national Arabic song lyrics corpus. In Proceedings of the LREC 2020 12th International Conference on Language Resources and Evaluation, Marseille, France, 11–16 May 2020; pp. 1318–1326.
55. Hindawi Foundation. Hindawi Foundation. Available online: <https://www.hindawi.org/> (accessed on 26 July 2023).
56. Christoph, G. GitHub—OpenArabic/1300AH: Texts from the 13th Hijri Century. GitHub. Available online: <https://github.com/OpenArabic/1300AH> (accessed on 26 July 2023).
57. Abdelali, A.; Cowie, J.; Soliman, H.S. Building A Modern Standard Arabic Corpus. In Proceedings of the Workshop on Computational Modeling of Lexical Acquisition, Split, Croatia, 25–28 July 2005.
58. Al-thubaity, A.; Alkhereyf, S.; Bahanshal, A. AraNPCC: The Arabic Newspaper COVID-19 Corpus. In Proceedings of the 5th Workshop on Open-Source Arabic Corpora and Processing Tools with Shared Tasks on Qur'an QA and Fine-Grained Hate Speech Detection, Marseille, France, 20 June 2022; European Language Resources Association: Paris, France, 2022; pp. 32–40.
59. Belinkov, Y.; Magidow, A.; Barrón-Cedeño, A.; Shmidman, A.; Romanov, M. Studying the history of the Arabic language: Language technology and a large-scale historical corpus. *Lang. Resour. Eval.* **2019**, *53*, 771–805. [CrossRef]
60. Goldhahn, D.; Eckart, T.; Quasthoff, U. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In Proceedings of the 8th International Conference on Language Resources and Evaluation, LREC, Istanbul, Turkey, 21–27 May 2012; pp. 759–765.
61. Antoun, W.; Baly, F.; Hajj, H. AraBERT: Transformer-based Model for Arabic Language Understanding. In Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools with a Shared Task on Offensive Language Detection, Marseille, France, 11–16 May 2020; pp. 9–15.
62. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V.; et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
63. Sidhoum, A.H.; Mataoui, M.; Sebbak, F.; Smaïli, K. ACQAD: A Dataset for Arabic Complex Question Answering. In Proceedings of the International Conference on Cyber Security, Artificial Intelligence and Theoretical Computer Science, Nanjing, China, 3–5 March 2023; HAL Open Science: Boumerdès, Algeria, 2023; pp. 1–12.
64. Zong, C.; Xia, R.; Zhang, J. *Text Data Mining*; Springer: Singapore, 2021. [CrossRef]
65. Husain, F.; Uzuner, O. Investigating the Effect of Preprocessing Arabic Text on Offensive Language and Hate Speech Detection. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2022**, *21*, 1–20. [CrossRef]
66. Fadel, A.; Tuffaha, I.; Al-Jawarneh, B.; Al-Ayyoub, M. Arabic Text Diacritization Using Deep Neural Networks. In Proceedings of the 2019 2nd international conference on computer applications & information security (ICCAIS), Riyadh, Saudi Arabia, 1–3 May 2019; pp. 1–7. [CrossRef]
67. Khondaker, M.T.I.; Waheed, A.; Nagoudi, E.M.B.; Abdul-Mageed, M. GPTAraEval: A Comprehensive Evaluation of ChatGPT on Arabic NLP. *arXiv* **2023**, arXiv:2305.14976.
68. Abdelali, A.; Darwish, K.; Durrani, N.; Mubarak, H. Farasa: A fast and furious segmenter for arabic. In Proceedings of the NAACL-HLT 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 11–16. [CrossRef]
69. Lahbari, I.; Alaoui, S.O.E. Exploring Sentence Embedding Representation for Arabic Question Answering. *Int. J. Comput. Digit. Syst.* **2023**, *14*, 189–198. [CrossRef]
70. Zhu, Y.; Pang, L.; Wu, K.; Lan, Y.; Shen, H.; Cheng, X. Cross-Model Comparative Loss for Enhancing Neuronal Utility in Language Understanding. *ACM Trans. Inf. Syst.* **2023**, *37*, 1–27. [CrossRef]
71. Alrowili, S.; Shanker, V. ArabicTransformer: Efficient Large Arabic Language Model with Funnel Transformer and ELECTRA Objective. *Find. Assoc. Comput. Linguist. EMNLP* **2021**, *2021*, 1255–1261.
72. Antoun, W.; Baly, F.; Hajj, H. AraGPT2: Pre-Trained Transformer for Arabic Language Generation. *arXiv* **2020**, arXiv:2012.15520.
73. Abdul-Mageed, M.; Elmadany, A.R.; Nagoudi, E.M.B. ARBERT & MARBERT: Deep bidirectional transformers for Arabic. In Proceedings of the ACL-IJCNLP 2021 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Online, 1–6 August 2021; pp. 7088–7105. [CrossRef]
74. Chouikhi, H.; Alsuhaibani, M. Deep Transformer Language Models for Arabic Text Summarization: A Comparison Study. *Appl. Sci.* **2022**, *12*, 11944. [CrossRef]
75. Abdelali, A.; Durrani, N.; Dalvi, F.; Sajjad, H. Interpreting Arabic Transformer Models. *arXiv* **2022**, arXiv:2201.07434.
76. Xia, C.; Zhang, C.; Zhang, J.; Liang, T.; Peng, H.; Yu, P.S. Low-shot learning in natural language processing. In Proceedings of the 2020 IEEE 2nd International Conference on Cognitive Machine Intelligence, CogMI, Atlanta, GA, USA, 8–31 October 2020; pp. 185–189. [CrossRef]
77. Ding, L.; Ye, S. Using Bidirectional Prompt Learning in NLP Few Shot Tasks. *Front. Comput. Intell. Syst.* **2023**, *3*, 167–172. [CrossRef]

78. Pasunuru, R.; Stoyanov, V.; Bansal, M. Continual Few-Shot Learning for Text Classification. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online and Punta Cana, Dominican Republic, 7–11 November 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 5688–5702.
79. Clark, J.H.; Choi, E.; Collins, M.; Garrette, D.; Kwiatkowski, T.; Nikolaev, V.; Palomaki, J. Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 454–470. [\[CrossRef\]](#)
80. Eid, A.M.; El-Makky, N.; Nagi, K. Towards machine comprehension of Arabic text. In Proceedings of the IC3K 2019 Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, Vienna, Austria, 17–19 September 2019; pp. 282–288. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.