entropy

Special Issue Reprint

Entropy in Machine Learning Applications

Edited by
Yanchun Liang

mdpi.com/journal/entropy

MDPI

Entropy in Machine Learning Applications

Entropy in Machine Learning Applications

Guest Editor

Yanchun Liang

MDPI

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

Guest Editor
Yanchun Liang
Zhuhai Sub Laboratory of
Key Laboratory for Symbol
Computation
and Knowledge Engineering
of National Education
Ministry
Zhuhai College of Science
and Technology
Zhuhai
China

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Entropy* (ISSN 1099-4300), freely accessible at: https://www.mdpi.com/journal/entropy/special_issues/1567UCE88V.

ISBN 978-3-7258-3065-7 (Hbk)
ISBN 978-3-7258-3066-4 (PDF)
https://doi.org/10.3390/books978-3-7258-3066-4

Contents

Article

A Semantic-Enhancement-Based Social Network User-Alignment Algorithm

Yuanhao Huang [1], Pengcheng Zhao [1], Qi Zhang [2], Ling Xing [1,*], Honghai Wu [1] and Huahong Ma [1]

1 The College of Information Engineering, Henan University of Science and Technology, Luoyang 471023, China
2 The School of Information Engineering, Southwest University of Science and Technology, Mianyang 621010, China
* Correspondence: xingling_my@haust.edu.cn

Abstract: User alignment can associate multiple social network accounts of the same user. It has important research implications. However, the same user has various behaviors and friends across different social networks. This will affect the accuracy of user alignment. In this paper, we aim to improve the accuracy of user alignment by reducing the semantic gap between the same user in different social networks. Therefore, we propose a semantically enhanced social network user alignment algorithm (SENUA). The algorithm performs user alignment based on user attributes, user-generated contents (UGCs), and user check-ins. The interference of local semantic noise can be reduced by mining the user's semantic features for these three factors. In addition, we improve the algorithm's adaptability to noise by multi-view graph-data augmentation. Too much similarity of non-aligned users can have a large negative impact on the user-alignment effect. Therefore, we optimize the embedding vectors based on multi-headed graph attention networks and multi-view contrastive learning. This can enhance the similar semantic features of the aligned users. Experimental results show that SENUA has an average improvement of 6.27% over the baseline method at hit-precision30. This shows that semantic enhancement can effectively improve user alignment.

Keywords: social networks; user alignment; semantic enhancement; graph contrastive learning

Citation: Huang, Y.; Zhao, P.; Zhang, Q.; Xing, L.; Wu, H.; Ma, H. A Semantic-Enhancement-Based Social Network User-Alignment Algorithm. *Entropy* **2023**, *25*, 172. https://doi.org/10.3390/e25010172

Academic Editor: Yanchun Liang

Received: 13 December 2022
Revised: 9 January 2023
Accepted: 11 January 2023
Published: 15 January 2023

1. Introduction

As different social networks offer their users distinctive functions, people tend to register accounts on several different social networks. In recent years, the number of online users on each social network has grown significantly. A huge amount of user data is generated due to users sharing and communicating on various social networks. Based on these data, researchers are able to analyze users' behavior and the evolution of social networks, which can in turn facilitate research in areas such as community discovery [1], recommender systems [2], link prediction [3], and other related fields. However, this development of multiple social networks also brings some problems. First, cross-domain user recommendation is inaccurate because users' behavior across different social networks is not always consistent. It is also difficult to find abnormal users and trace the abnormal sources, because malicious users tend to spread false remarks on multiple social networks. After user alignment associates a user's multiple accounts across different social networks, comprehensive analysis of users' behaviors on these networks can be used to solve problems such as cross-domain recommendation [4] and abnormal user detection [5]. User alignment is a basic and meaningful research, and the accuracy of alignment needs to be improved.

User alignment is also known as anchor link prediction, user identification, and social network alignment [6–8]. Its purpose is to associate the accounts registered by real users across different social networks. However, the differences in the same user's features and friends across the different social networks will reduce the accuracy of user alignment, which is referred to as the semantic gap problem. Improving the effect of user alignment by reducing the semantic gap can be broken down into three aspects: (1) Accurate and comprehensive representation of user characteristics. Due to the heterogeneity between different social networks, computing user similarity based on user features and network topologies is commonly influenced by noise [9,10]. Existing methods of this kind are too biased to determine whether two users of two different social networks are the same real user by simply analyzing the users' attributes, such as age and gender. Users' writing patterns, personal emotion, and other semantic features can be mined through an analysis of usernames and text posts [11]. Integrated consideration of the username, user-generated content, geographic location, network topology, and other data can help mine users' semantic features, comprehensively characterize users, and reduce the negative impacts of local feature differences on user-alignment effects [12–15]. Notably, however, user feature mining methods discussed above do not consider the reliability of data, computing overhead, and missing data problems. (2) Improving noise adaptation ability. Since the user features and network topology of the same user differ slightly from one social network to another, the noise contained in the semantic features of the user will reduce the user similarity. Feng et al. [16] achieved user alignment based on the user's position and reduced the interference of position noise with user alignment by constructing a position encoder and trajectory encoder to calculate the user similarity. Xiao et al. [9] enhanced the noise adaptation ability of the model by adding perturbations to the data and designing a noise-adapted loss function. Xue et al. [17] proposed three noise-processing strategies: dropping, retaining, and conditional retention. Notably, the above noise-processing measures do not consider the effect of data propagation between users on noise. (3) Optimizing user-alignment effects. After user features are pre-processed, user alignment is often achieved using network representation learning. This method compares the similarity of user embedding vectors to determine whether they are the same real user, after embedding users of two social networks into the same vector space. To improve the accuracy of user alignment, many embedding optimization methods have been proposed [18–20]. Zhang et al. [21] and Chen et al. [22] improved the alignment effect by using a generative adversarial network to optimize the embedding representation of users. Notably, while these embedding optimization methods can improve user alignment, they do not sufficiently consider the impact of highly similar users on user alignment in the same social network and in the social network to be aligned.

To solve the above problems, we propose a semantic enhancement algorithm for social network user alignment, which enhances the semantic features of users from three aspects: semantic representation, noise adaptation, and embedding optimization. It can improve the accuracy of user alignment. (1) There are different characteristics of user attributes, UGCs, and user check-ins. First, user attributes have a low computational overhead and reflect users' behaviors. There are more semantic features included in UGCs, such as users' preferences and writing habits, but the data volume of pictures and videos is too large. User check-ins contain highly reliable data related to the time and place of posting. Therefore, we represent the semantic features of users at multiple levels based on user attributes, text in UGCs, and user check-ins. (2) The embedded view constructed based on semantic representations contains both feature noise and topological noise. Considering the impact of data propagation among users on user alignment, we compute the semantic centrality of users based on their influence and preferences. During graph-data augmentation, the weights of features and topologies are adaptively adjusted based on semantic centrality to highlight the important features and topologies. (3) To improve user alignment, it is necessary to optimize the embedding vector of users. Friends in the same social network have similar semantics, as do aligned users in the social network to be aligned. We aggregated the important semantic features of similar neighbors by using a multi-headed graph attention network, then used contrastive learning on the same social network views and the alignment views. This approach can reduce the semantic similarity between users in the social network view while enhancing semantic similarity between aligned users in the aligned view. The social network user-alignment effect can be effectively improved by enhancing the semantic features of users using these three aspects. The contributions of the work are summarized as follows.

- Multi-level data analysis can improve the mining of users' semantic features. We extract meta-semantic features, specifically, users' preferences and cities of residence from UGCs and check-ins, and then extract high-level semantic features of users from user attributes, UGCs, and check-ins, based on BERT, word2vec, and meta-graph, respectively. The semantic features of users are represented on multiple levels, which reduces the interference of local semantic noise and improves the accuracy of computing user similarity.
- The heterogeneity of different social networks introduces feature and topology noise interference into the calculation of user alignment. Since users' influence and preferences have important impacts on semantic propagation among users, we compute the semantic centrality of users based on these two features and assign appropriate weights to the features and topologies. The model's adaptability to noise is improved by graph-data augmentation to enhance the user-alignment effect.
- As the feature embedding vectors of the same user are not exactly the same across different social networks, the user's embedding vector is optimized by means of semantic fusion and contrastive learning. The features of the surrounding similar neighbors are aggregated using a multi-head graph attention network to enhance the semantic features of the users themselves. Contrastive learning improves the embedding distance of users in the same social network while reducing the embedding distance of aligned users in the social network to be aligned, which ensures the accuracy of the obtained user alignment.

The remainder of this article is organized as follows. The related works are reviewed in Section 2. Subsequently, Section 3 introduces the relevant definitions and user alignment issues. The details of the SENUA algorithm are described in Section 4, followed by Section 5, which presents the experiments. Finally, Section 6 concludes this article.

2. Related Work

2.1. User Alignment

User alignment has been extensively studied. Existing approaches can be classified into three categories: user feature-based, network-topology-based, and hybrid approaches.

In user feature-based approaches, the semantic features of users are mined based on data such as user attributes and UGCs to determine whether they represent the same real user by computing the user similarity [11,16,23–29]. During the account registration process, the username is a required item, which enables the naming habits of users to be mined; thus, the user similarity is most widely computed based on the username. Li et al. [25] analyzed the phonetic and font similarities of Chinese usernames to achieve user alignment. To deeply mine user features, Xing et al. [30] not only analyzed the length, character features, and alphabetic features of usernames, but also mined user preferences from their posted contents to improve user-alignment accuracy.

Network-topology-based approaches compare the friend network similarity of users in the source and target networks to achieve user alignment [18,31–38]. At present, network representation learning methods are commonly used to mine network topology features [35]. This kind of method can achieve user alignment by minimizing the embedding distance after embedding the user's network topology features into a low-latitude vector space [36,37]. However, the embedding vectors of different network topologies are not stable enough. Therefore, network topology is often combined with information propagation [39], genetic algorithms [40], community discovery [38], and generative adversarial networks [18] to enhance user-feature representations.

The user-feature-based approaches focus on the users' personal information and the content they post. The network-topology-based approach focuses on the user's friendships. There is complementarity or redundancy between these two different types of data. Notably, while a single method with a single type of data cannot deeply mine the semantics of users, hybrid methods that combine user features and network topologies can more fully mine the semantic features of users and thereby improve user alignment [9,12–14,22,41–44]. Graph neural networks are commonly used at present to fuse user features and network topologies simultaneously. These methods aggregate the feature vectors of the user's neighbors to enhance the semantic features of the user, and subsequently determine whether two users match based on the similarity of the embedded vectors [42,44]. However, mining the semantic features of users based on graph neural networks also captures feature noise and topological noise in social networks. AFF-LP [45] uses an attention mechanism to extract network topology and temporal features in order to reduce noise interference and improve the accuracy of the algorithm. Notably, this method only considers the effect of network noise, while failing to consider the feature noise due to user feature differences. GATAL [9] removes edges to simulate network noise and randomly changes node features to simulate feature noise. After noise processing, the graph attention network is used to fuse the neighborhood features so that the algorithm can maintain good performance, even under noisy conditions. In addition, the user-alignment algorithm combines graph neural networks with generative adversarial networks to solve the problem of accuracy reduction due to semantic variability [22]. While these studies have made some progress, the noise augmentation method in users' semantic features is random; thus, it is not adaptive to the data propagation characteristics in social networks. Accordingly, the effect on user semantic enhancement needs to be improved.

2.2. Text Feature Extraction

There are huge amounts of text, images, video, and other multi-source data in social networks. Image and video have a high computational overhead and difficult semantic extraction. Scholars often mine text features through natural language processing [46,47]. The text contains more semantic features, which are usually mined by two steps: sequence annotation [48] and vector embedding. Since the number and completeness of words in short and long texts differ greatly, it is more effective to annotate them at different levels [49]. Shao et al. [50] analyzed the data structure based on latent variables in random fields and constructed two frameworks for sequence annotation at the word and sentence levels, respectively. The commonly used text feature embedding methods include word2vec [51], FastText [52], BERT [53], etc. BERT is a transformer-based language representation model. It performs self-supervised training by masking parts of words to mine text features. Currently, text-embedding methods are often combined with attention mechanisms to enhance the completeness and accuracy of extracted features. Our proposed user alignment approach deeply incorporates attention mechanisms to enhance the semantic features of similar users.

2.3. Graph Representation Learning

Graph representation learning includes node embedding, graph neural networks, and generative graph models [54]. The node embedding contains an encoder–decoder, random wandering, and matrix decomposition. This type of method is a shallow embedding model, with which is difficult to capture the deep features of nodes. It also has limitations such as high overhead and inadequate feature mining. Graph neural networks embed user features into vector space by propagating, aggregating, and updating features between nodes. This class of methods is an end-to-end deep embedding model that can perform feature mining directly based on graph data and helps to mine deep features of nodes. Deep generative models include variational autoencoders, generative adversarial networks, and autoregressive models. Normally, this class of methods usually optimizes node vectors by confronting encoders and decoders with each other. The degree of similarity between friends has a significant impact on the accuracy of user alignment. Graph neural networks can adaptively aggregate neighboring features and enhance the user's features. Using graph neural networks has greater benefits for user alignment.

2.4. Graph Contrastive Learning

Contrastive learning has already received widespread research attention and made significant achievements in many tasks, such as natural language processing [55] and computer vision [56]. In recent years, contrastive learning has been applied to graph representation learning, which is referred to as graph contrastive learning. In graph contrastive learning, multiple views are generated via graph-data augmentation, and then these nodes are embedded into the vector space by encoding and projection; finally, the embedding effect is optimized by contrastive learning. You et al. [57] designed four graph-data augmentation methods: node dropout, edge perturbation, feature masking, and subgraph sampling. Hassani et al. [58] used a diffusion kernel for data augmentation, enabling each node to sense more global information. Notably, existing graph-data augmentation methods use a uniform transformation for topologies and features, which can lead to poor performance. Therefore, Zhu et al. [59] proposed an adaptive data augmentation scheme that preserves important features and topologies during augmentation.

User alignment based on either user characteristics or network topology alone is necessarily limited. The fusion of these two types of data can effectively enhance user semantic features and improve the user-alignment effect. Considering the reliability of the data and the overhead of the algorithm, we deeply mine the semantic features of users from user attributes, UGCs, and check-ins. In addition, we propose a modified graph contrastive learning approach to achieve social network user alignment; this approach uses semantic centrality in graph-data augmentation to improve the algorithm's self-adaptation to noise, and enhances the semantic feature similarity of aligned users via contrastive learning in multiple views.

3. Preliminaries

In this section, we introduce the related definitions and the user-alignment issue. The symbols used in this article and the corresponding meanings are summarized in Table 1.

Table 1. Definitions of symbols.

Notation	Definition
G^S,G^T	Source social network,target social network.
U	Set of users in the social network.
E	Edge set of the social network.
A	User features of the social network.
A_p, A_c, A_ℓ	User attributes, UGCs, and user check-ins.
$A_p^{name}, A_p^{area}, A_p^{pref}$	User name, city of residence, and user preference.
u_i	The *i*th user.
$\boldsymbol{V}$	Embedding vectors of user semantic features.
$\mathbb{R}$	Vector space.
D	Feature dimension.
N	Total number of users in the network.
M	Aligned user pairs.
$\boldsymbol{R}_{shar}$	Preference sharing matrix.
$\xi(u_i)$	Semantic centrality of user u_i.
$p^e_{u_iu_j}$	Topology sampling probability.
p^f_d	Feature masking probability.

3.1. Semantic Social Network View

Social networks contain huge amounts of user data. Based on the reliability, discernment, and data scale of these data, we selected user attributes A_p, user-generated contents A_c, and user check-in A_ℓ as the basis for discerning aligned users, which ensures that sufficient semantic information is available for the represented users. The user attributes A_p contain the username A_p^{name}, city of residence A_p^{area}, and the user preferences

A_p^{pref}. We use only the text of posts as UGCs to avoid the huge overhead associated with the task of analyzing images and videos. User check-ins refer to time and place at which a user makes a post. In this paper, we consider a semantic social network view $G = (U, E, A)$. $U = \{u_1, u_1, \cdots, u_n\}$ represents the set of n nodes, and each node represents a user; $E = \{e_{ij} = (u_i, u_j)|u_i, u_j\}$ denotes the set of edges. This is an $n * n$ matrix that represents the friend relationships between n users. If $e_{ij} = 1$, users u_i and u_j are friends; otherwise, they are not friends. Based on the number of edges connected to node u_j, we can get the degree of node u_j as $\sum_{i=1}^{n} e_{ij}$. The user features A are represented by a triplet $A = \{A_p, A_c, A_\ell\}$; these elements, respectively, represent user attributes, user-generated contents, and user check-ins.

3.2. Semantic Enhancement User Alignment

Typically, a user has multiple social network accounts. In this paper, we aim to solve the problem of matching social network accounts belonging to the same person, as shown in Figure 1. In order to distinguish the semantic views corresponding to different social networks, the source and target social networks to be aligned are represented by $G^S = (U^S, E^S, A^S)$ and $G^T = (U^T, E^T, A^T)$, respectively. The two views with semantic gaps include noise; we use graph-data augmentation to reduce the impact of the noise. Moreover, to improve the alignment accuracy, GAT and contrastive learning are used to enhance the semantic features of the users. Finally, we determine whether two users represent the same real user based on user similarity, that is, aligned user pairs $M = \{(u_i, u_j)|u_i \in U^S, u_j \in U^T\}$.

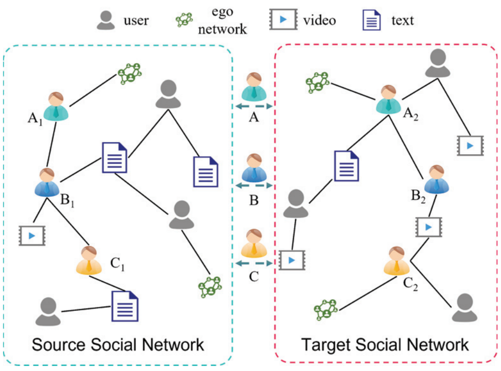


Figure 1. User-alignment diagram.

3.3. Multi-View Graph Contrastive Learning

Graph contrastive learning typically involves four steps: data augmentation, encoding, projection, and contrastive learning. (1) Two differing views are generated from the original view by data augmentation; (2) each view is encoded by a graph neural network; (3) the nodes of two views are mapped to the same vector space; (4) the consistency of the same node in different views is maximized by means of contrastive learning. To achieve user alignment, we propose a modified multi-view graph contrastive learning approach. Its input includes the source social network G^S and the target social network G^T. After data augmentation is performed for both views, the view to be aligned and the augmented view are encoded as vectors. In the comparative learning stage, we not only contrast the augmented views of G^S and G^T, respectively, but also contrast the aligned views G^S and G^T.

In addition, to improve the effect of graph contrastive learning on user alignment in social networks, we propose a semantic centrality attention that considers the impacts of user influence and user preferences on user-alignment effects in social networks.

During data augmentation and encoding, the weights are adaptively adjusted to highlight the important semantic features of users.

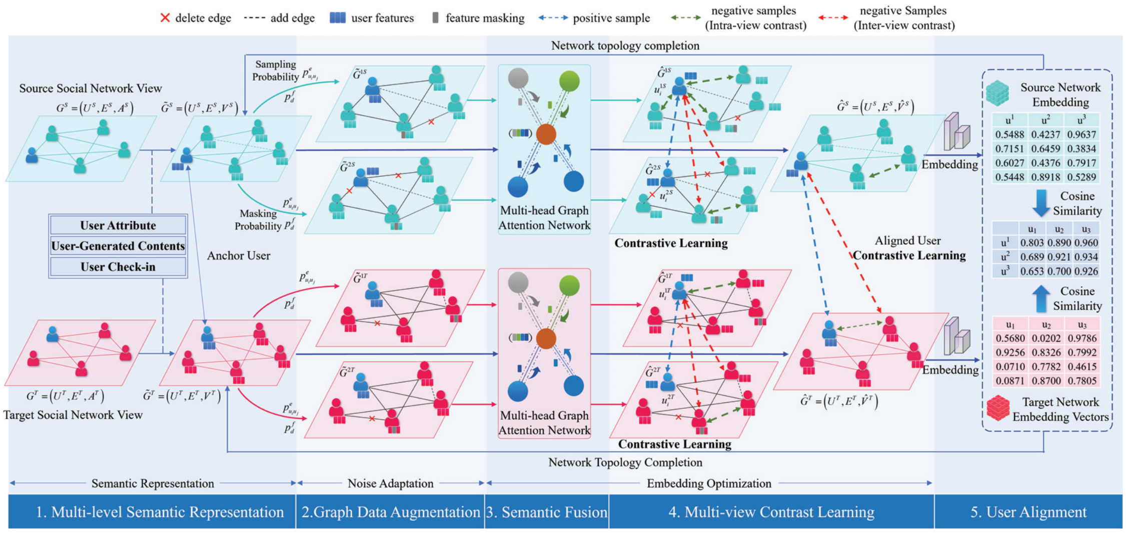
4. SENUA Algorithm

In this section, we first provide an the overview of SENUA, and then present the details of each component.

4.1. Overview of SENUA

In this paper, we propose the SENUA algorithm to improve the accuracy of social network user alignment by enhancing the semantic features of users. The overall framework is illustrated in Figure 2, and the specific algorithm of SENUA is presented in Algorithm 1. SENUA takes as input the source social network view G^S and the target social network view G^T to be aligned. To improve the alignment effect, we enhance the semantic features of users in three aspects: semantic representation, noise adaptation, and embedding optimization. The process of SENUA consists of five steps. (1) Adequate user semantic feature representation can reduce the interference of the local semantic gap on global semantics. Taking user behavior, spatio-temporal information, and user relationships into account, multi-dimensional semantic features of users are extracted from user attributes, UGCs, and check-ins via semantic analysis. (2) Due to the variability between different social networks, the extracted semantic features often contain noise, which can affect the user-alignment effect. The algorithm's noise-adaptation capability can be improved through the use of graph-data augmentation for features and topologies in multiple views. Notably, the effect of graph-data augmentation is not stable for different networks or downstream tasks. Accordingly, to improve the effectiveness of data augmentation in social network user alignment, we propose semantic centrality attention to adaptively adjust the data augmentation weights. Since the probability of data spreading among users with high influence and the same preference is higher, these users usually have more common semantic features. During graph-data augmentation, computing semantic centrality based on influence and user preferences can help to ensure that important user semantic features are retained. (3) When attempting to determine whether a user is an aligned user based on their semantic features, the key lies in how to deeply mine the similar semantic features of aligned users. Users who communicate more frequently on social networks tend to have more similar semantic features. Graph neural network-based fusion of semantic features of neighbors can thereby enhance the representations of individual users. (4) Highly similar users in the same social network can interfere with user alignment. Through the use of contrastive learning in multiple views, we not only reduce the semantic similarity between users in the same social network, but also enhance the semantic similarity between aligned users, which can optimize the feature embedding vectors of users. (5) Based on the optimized multi-view embedding vectors, user similarity is computed using the cosine distance. If the similarity reaches a threshold value, the two users are considered as aligned users. Since many operations are the same for the source social network view G^S and the target social network view G^T, if S and T are not used to distinguish between the views in what follows, this will mean that both networks have to perform this operation.

In brief, the differences of the proposed algorithm are: (1) Multiple embedding methods are combined to fully represent user semantic features through low-level and high-level semantic feature extraction. It can reduce the influence of local noise. (2) Calculating the semantic centrality of users based on their preferences and user influence, and using it to compute the probability of topology and feature augmentation in graph-data augmentation. (3) Computing feature aggregation weights in graph attention networks based on the semantic centrality of users. (4) The application scenario of contrastive learning is extended from a single social network to multiple social networks. Enhance similarity between aligned users through multi-view contrastive learning. (5) Top-k highly similar users are selected as aligned users, and then the missing network topology is completed by aligned users.


delete edge
add edge
user features
feature masking
positive sample
negative samples (Intra-view contrast)
negative Samples (Inter-view contrast)
Network topology completion
Source Social Network View
$G^S=(U^S,E^S,A^S)$
$\tilde{G}^S=(U^S,E^S,V^S)$
Sampling Probability $p^e_{u_iu_j}$
p^f_d
Masking Probability p^f_d
User Attribute
User-Generated Contents
User Check-in
Anchor User
$\tilde{G}^{1S}$
$\tilde{G}^{2S}$
Multi-head Graph Attention Network
$\hat{G}^{1S}$
$\hat{G}^{2S}$
Contrastive Learning
$\hat{G}^S=(U^S,E^S,\hat{V}^S)$
Embedding
Aligned User Contrastive Learning
Source Network Embedding
Cosine Similarity
Target Network Embedding Vectors
$G^T=(U^T,E^T,A^T)$
$\tilde{G}^T=(U^T,E^T,V^T)$
Target Social Network View
$\tilde{G}^{1T}$
$\tilde{G}^{2T}$
$\hat{G}^{1T}$
$\hat{G}^{2T}$
$\hat{G}^T=(U^T,E^T,\hat{V}^T)$
Network Topology Completion
Semantic Representation
Noise Adaptation
Embedding Optimization
1. Multi-level Semantic Representation
2.Graph Data Augmentation
3. Semantic Fusion
4. Multi-view Contrast Learning
5. User Alignment

u^1	u^2	u^3
0.5488	0.4237	0.9637
0.7151	0.6459	0.3834
0.6027	0.4376	0.7917
0.5448	0.8918	0.5289

	u_1	u_2	u_3
u^1	0.803	0.890	0.960
u^2	0.689	0.921	0.934
u^3	0.653	0.700	0.926

u_1	u_2	u_3
0.5680	0.0202	0.9786
0.9256	0.8326	0.7992
0.0710	0.7782	0.4615
0.0871	0.8700	0.7805

Figure 2. The framework of the proposed algorithm.

Algorithm 1: Social network user alignment.

```
**Input:** social network $G^S$ and $G^T$
**Output:** aligned user $M$ in $G^S$ and $G^T$
(1) Semantic representation: UGCs $A_c \xrightarrow{\text{LDA}}$ user preference $A_p^{pref}$;
Check-In $A_\ell \xrightarrow{Bayesian}$ resident city $A_p^{area}$; $A_p = \left\{A_p^{pref}, A_p^{area}, A_p^{name}\right\}$
$A_p, A_c, A_\ell \rightarrow$ feature vectors $\boldsymbol{V}_p, \boldsymbol{V}_c, \boldsymbol{V}_\ell$; $\boldsymbol{V}_p, \boldsymbol{V}_c, \boldsymbol{V}_\ell \xrightarrow{merge} \boldsymbol{V}$
**for** *epoch* $\leftarrow 1, 2 \cdots$ **do**
    (2) Graph data augmentation:
    $\tilde{G}^S \xrightarrow{p^e_{u_iu_j}, p^f_d} \tilde{G}^{1S}\tilde{G}^{2S}$; $\tilde{G}^T \xrightarrow{p^e_{u_iu_j}, p^f_d} \tilde{G}^{1T}\tilde{G}^{2T}$;
    (3) Converge neighborhood features by GAT with semantic centrality;
    (4) Contrastive learning: $\hat{G}^S \longleftrightarrow \hat{G}^T; \hat{G}^{1S} \longleftrightarrow \hat{G}^{2S}; \hat{G}^{1T} \longleftrightarrow \hat{G}^{2T}$;
    Compute the loss objective $\mathcal{J}$ with contrastive learning;
    Update parameters by applying stochastic gradient ascent to maximize $\mathcal{J}$;
    (5) Obtain node embeddings of $\hat{G}^S$ and $\hat{G}^T$;
    Computer user similarity sim$\left(u_i^S, u_j^T\right)$;
    Find top $k$ anchor users greater than the threshold by comparing similarity;
    Complete the missing network topology;
**end**
```

4.2. *Multi-Level Semantic Representation*

There are two problems with adequately representing the semantic features of users in user alignment studies. (1) Absent or fake user attributes. When users register accounts on multiple social networks for privacy protection, user attributes may be empty or forged except for the username. (2) Inadequate semantic feature mining. The embedding of user features into the low-dimensional vector space may result in some semantic features' absence. For example, to make a computer understand human language, representing the meaning of a whole sentence with a vector will necessarily lose some semantics of the sentence. To address these two issues, we propose a multi-level semantic feature representation, outlined as shown in Figure 3.

Given two social network views to be aligned, two kind of meta-semantic features, user preferences and resident cities, are extracted from UGCs and check-ins, respectively. High-level user semantic features are extracted from three dimensions, user attributes, UGCs, and check-ins, and then embedded and fused to obtain the user's feature embedding vector $\boldsymbol{V}$. Feature extraction from multiple levels and dimensions can effectively enhance the semantic features of users and improve user-alignment effects.

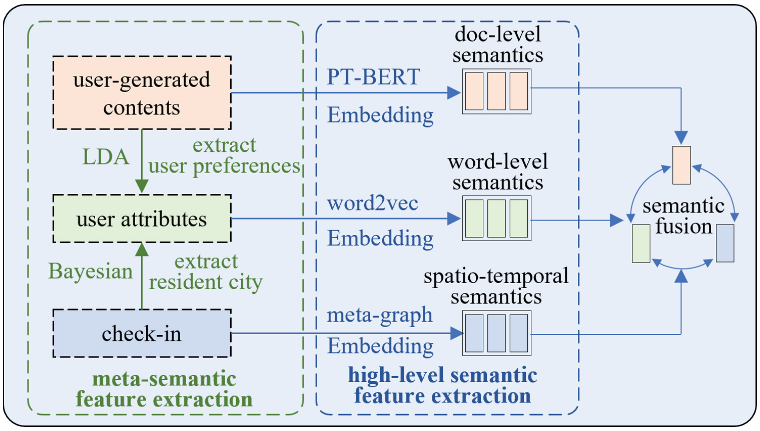


Figure 3. Multi-level semantic feature representation.

4.2.1. Meta-Semantic Feature Extraction

User attributes are highly discriminative but contain few semantic features. It is not possible to confidently conclude that two users on different social networks are the same person by looking only at the age and gender. Therefore, in this paper, users' preferences and cities of residence extracted from UGCs and check-ins are used to supplement users' attributes for the subsequent user alignment task. Here, user preference refers to the user's fondness for something, and the city of residence refers to the location from which the user most frequently posts on social networks. These two meta-semantic features are extracted from UGCs and user check-ins, not filled in by users themselves; thus, they can represent user features more reliably. It can be used to compute user similarity more accurately and improve the effect of user alignment.

Extraction of User Preferences: UGCs refer to posts made by users that contain more user behavior characteristics. With the latent Dirichlet allocation (LDA) topic model, the topics of posts can be extracted from UGCs. LDA is a probabilistic topic model that analyzes the words in a document to obtain the topic of each document and its percentage. Most existing studies use a single LDA topic model for a single social network without considering the variability of users' posts across different social networks. This approach accordingly limits the representational power when analyzing multiple social network topics. Therefore, we extract cross-view topics from the social network views to be aligned based on C-LDA [60]. The user-view and view-word distributions are employed to represent the user's social network view preferences and the differences in language styles across different views. Each view sets a polynomial distribution of background subject words to reduce the interference generated by meaningless noise words in the document. To improve the similarity of subject terms and the association between users across social network views, we retain subject terms with high co-occurrence frequencies in different views and add them as user preferences A_p^{pref} to user attributes A_p.

City of residence extraction: User check-ins can be used to reliably determine the times and places at which users make posts. However, the precise positioning of user check-ins in different social networks is often inconsistent, which may interfere with user alignment. If the user's city of residence is analyzed based on the time and location of the check-in, this can fuzzify the precise location features and improve the robustness of the algorithm. Therefore, based on the Bayesian recommendation algorithm [61], we extract users' cities of residence from multiple views based on their preferences A_p^{pref}, check-ins A_ℓ, and social connections E. The preference-based city of residence probability is obtained based on the location of users with the same preference; the influence-based city of residence probability is obtained based on the influence of friends in the social network; the distance-based city of residence probability is obtained based on the distance between users' check-in locations; the linear sum of these three probabilities forms the final city of residence probability. The city with the highest probability is the determined to be the city of residence of the user A_p^{area} and is added to the user attribute A_p.

4.2.2. Word-Level Semantic Representation

After meta-semantic feature extraction, user attribute A_p includes username A_p^{name}, city of residence A_p^{area}, and interest preference A_p^{pref}. Since these words are not related to each other, global semantic features do not need to be considered. Therefore, based on word2vec [51] we vectorize the user attributes to extract word-level semantics. The user attributes are divided into words, after which, stop words (such as "a" and "the") are dropped. Each word is represented by a Huffman encoding, making the encoding of the more frequent words shorter, which can improve the training efficiency of our algorithm. There are more repetitive words describing city and preference in user attributes, and the dataset is small, which is suitable for training word vectors with CBOW—a language model of word2vec. After the CBOW model training, we get the feature vectors V_p^{name}, V_p^{area}, and V_p^{pref}, corresponding to the username, city of residence, and interest preferences. After merging these features together, the feature vector corresponding to the user attributes is as follows:

$$\boldsymbol{V}_p = \left[v_{1p}, v_{2p}, \cdots, v_{ip}, \cdots, v_{np}\right] \in \mathbb{R}^{D\times n},$$

where v_{ip} represents the word-level semantic embedding vector of user i. $\mathbb{R}$ denotes the vector space, and d denotes the feature dimension of the embedding vector.

4.2.3. Document-Level Semantic Representation

Compared with user attributes, UGCs contain more semantic features, such as sentiment and writing patterns. These semantic features facilitate user alignment; however, the included local semantic noise may also interfere with the alignment effect. The semantic features of UGCs cannot be fully mined using word2vec. Notably, the embedding vector trained based on the BERT [53] method contains more semantic features, which can reduce the noise information in UGCs. Therefore, based on PT-BERT [62], we extract document-level semantics from UGCs. The original sentence embedding is obtained by BERT, after which a pseudo-sentence embedding of corresponding length is generated. The original embedding and the pseudo-embedding are used to the final embedding vector based on the attention mechanism. Unbiased encoders are trained using contrast learning in true and false embedding vectors, which can enhance the semantic features of sentences. After training, the user-generated contents A_c are converted into the corresponding feature vector:

$$\boldsymbol{V}_c = [v_{1c}, v_{2c}, \cdots, v_{ic}, \cdots, v_{nc}] \in \mathbb{R}^{D\times n},$$

where v_{ic} represents the document-level semantic embedding vector of user i.

4.2.4. Spatiotemporal Semantic Representation

It is not easy to deeply mine the association between two users based solely on the user location at the time of posting. Therefore, we combine time and space by using ACTOR [63] to deeply mine the user's spatio-temporal semantics and thereby improve the user-alignment effect. The times and locations of check-ins and users are used as nodes to construct a heterogeneous network. According to different types of node linkage patterns, such as $T1 - U1 - U2 - T2$, temporal and spatial features are embedded into the same vector space. Deeper semantic features can be captured by maintaining the higher-order proximity of different levels. After training, the user check-in A_ℓ is transformed into a spatio-temporal semantic embedding vector:

$$\boldsymbol{V}_\ell = [v_{1\ell}, v_{2\ell}, \cdots, v_{i\ell}, \cdots, v_{n\ell}] \in \mathbb{R}^{D \times n},$$

where $v_{I\ ell}$ represents the spatio-temporal semantic embedding vector of user $\boldsymbol{i}$.

Through meta-semantic feature extraction, we obtain the user preferences A_p^{pref} and cities of residence A_p^{area}. Adding them to the user attribute A_p can reduce the negative impact on the user-alignment effect of missing or false of user attributes. Through multi-dimensional user feature semantic analysis, we extract the corresponding word-level feature embedding vector $\boldsymbol{V}_p$, document-level feature embedding vector $\boldsymbol{V}_c$, and spatio-temporal feature embedding vector $\boldsymbol{V}_\ell$ from the user attributes A_p, user-generated contents A_c, and user check-ins A_ℓ. These feature embedding vectors are fused and averaged to obtain the embedding vector $\boldsymbol{V}$, representing user features. Meanwhile, the original views of the source and target social networks are converted to embedded views. The process is as follows:

$$G^S = \left(U^S, E^S, A^S\right) \to \tilde{G}^S = \left(U^S, E^S, \boldsymbol{V}^S\right).$$

$$G^T = \left(U^T, E^T, A^T\right) \to \tilde{G}^T = \left(U^T, E^T, \boldsymbol{V}^T\right).$$

4.3. Graph-Data Augmentation with Semantic Noise Adaption

Data augmentation is a kind of data expansion and enhancement method. In the field of image processing, data augmentation refers to increasing the sample size by transforming the image. In graph networks, graph-data augmentation is achieved by adding perturbations to edges and features [64]. Across different social networks, the semantic features and topologies of users exhibit some variability. The semantic noise included in the social network view reduces the accuracy of user alignment. To address this problem, we improve the generalization capability of the algorithm by employing graph-data augmentation for the semantic features and topologies of the users in the embedded view. The existing graph-data augmentation methods are not well adapted to the dynamic data diffusion characteristics in social networks, meaning that the user alignment is insufficiently effective. Therefore, we propose a graph-data augmentation method with a semantic centrality attention mechanism to ensure a reasonable distribution of augmentation weights. This enables the augmented view to improve the algorithm's self-adaptation to noise while ensuring that important topologies and features remain unchanged. Below, we describe the aspects of semantic centrality, topology-level semantic augmentation, and feature-level semantic augmentation.

4.3.1. Semantic Centrality

Users who communicate more frequently on social networks tend to be more semantically similar. Based on the similar semantics of friends, the semantic features of users themselves can be made more complete, which can improve user alignment. Users' influence and preferences each have a significant impact on their communication. Due to the power-law distribution characteristic of social networks, most users usually have a small number of friends. Users who are followed by more people tend to have more influence. Moreover, users with similar preferences communicate with each other more frequently. Therefore, we compute semantic centrality attention weights based on influence and preference. The critical user features and network topology in a given view can be retained by increasing the probability of masking the features of users with low influence and the probability of removing the topology of users with different preferences.

In undirected graphs, degree indicates the number of friends of a user. We use degree centrality to indicate the importance of a user in a social network. The computation formula is as follows:

$$Degree\ Centrality = \frac{k_i}{N-1}, \tag{1}$$

where k_i denotes the degree of user i and N denotes the total number of users in this network. The degree centrality of users in the social network relationship graph measures user influence, and the degree centrality of users in the preference sharing relationship graph measures the degree of user preference. Preference sharing relationships are constructed from user-preference relationships and social network relationships [65]. As shown in Figure 4, the network topology E of the embedded view represents the social network relationships. The user-preference relationships $\boldsymbol{R}_{pref}$ are constructed according to the user and the corresponding user preferences A_p^{pref}. According to the user preferences, users with common preferences are constructed as preference sharing relationships. The formula can be expressed as follows:

$$\boldsymbol{R}_{shar} = \left(\boldsymbol{R}_{pref}\boldsymbol{R}_{pref}^T\right) \circ E \tag{2}$$

where E denotes social relationships, and the Hadamard product $\circ E$ is used to ensure that the constructed preference sharing relationships belongs to a subset of E. The matrix $\boldsymbol{R}_{pref}\boldsymbol{R}_{pref}^T$ multiplied together can link users with the same preferences.

The semantic centrality $\xi(u_i)$ of user u_i can be represented as

$$\xi(u_i) = deg_{user}(u_i) + deg_{shar}(u_i) \tag{3}$$

where $deg_{user}(\cdot)$ denotes the degree centrality of user u_i in the social network relationship graph, and $deg_{shar}(\cdot)$ denotes the degree centrality of user u_i in the preference sharing relationship graph.

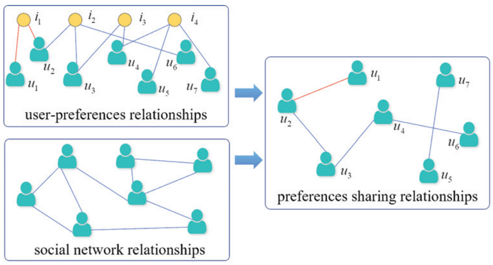

i_1
i_2
i_3
i_4
u_1
u_2
u_3
u_4
u_5
u_6
u_7
user-preferences relationships
social network relationships
preferences sharing relationships

Figure 4. Construction process of preferences a sharing relationship.

4.3.2. Topology-Level Semantic Augmentation

Users' friendships are inconsistent across social networks, and this topological noise can lead to semantic gaps for the same user from different views. We accordingly perform topology-level semantic augmentation based on the semantic centrality of the user, which constructs a new edge set $\tilde{E}$ from the network topology E of the embedded view with sampling probability $p^e_{u_iu_j}$. This reduces the influence of network topology noise on user alignment. The sampling probability $p^e_{u_iu_j}$ refers to the probability of preserving the topology (u_i, u_j), which reflects the importance of the edge that connects user u_i and user u_j.

We indicate the degree of topological importance based on the average of the semantic centrality of users u_i and u_j. The weight of the topology is the average of the semantic centrality of the connected users, namely, $w^e_{u_iu_j} = \left(\xi(u_i) + \xi(u_j)\right)/2$. To reduce the effect of the power-law distribution property of the social network on the drop probability, we take the logarithms of topological weights, namely, $\lambda^e_{u_iu_j} = \log w^e_{u_iu_j}$. The probabilities are normalized by the following equation:

$$p^e_{u_iu_j} = \min\left(\frac{\lambda^e_{u_iu_j} - \lambda^e_{\text{min}}}{\lambda^e_{\text{max}} - \lambda^e_{\text{min}}}, p^e_\tau\right) \tag{4}$$

where λ^e_{max} and λ^e_{min} denote the maximum and minimum values of the topological weights $w^e_{u_iu_j}$, respectively. p^e_τ is the truncation probability, indicating that the topology is not allowed to fall below the probability p^e_τ; this prevents damaging the topology of the network with lower sampling probability.

4.3.3. Feature-Level Semantic Augmentation

The contents of users' posts for the same event are inconsistent across social network views. This feature noise can cause semantic gaps for the same user in different views. Based on the user's semantic centrality, feature-level semantic augmentation can reduce the negative impact of feature noise on user alignment. We zero out certain dimensions of users' features that are unimportant, which improves the algorithm's adaptability to feature noise. To ensure randomness, we obtain $\tilde{\boldsymbol{m}} \in \{0, 1, \}^F$ by randomly sampling from a vector of Bernoulli distribution with probability $1 - p^f_d$, and then generate the feature vectors $\tilde{\boldsymbol{V}}$. The computing process is as follows:

$$\tilde{\boldsymbol{V}} = [\boldsymbol{v}_1 \circ \tilde{\boldsymbol{m}}, \boldsymbol{v}_2 \circ \tilde{\boldsymbol{m}}, \cdots, \boldsymbol{v}_{\text{n}} \circ \tilde{\boldsymbol{m}}]^T \tag{5}$$

where $\boldsymbol{v}_{\text{n}}$ denotes the corresponding feature vector of user n. The symbol $\circ$ is the Hadamard product, which denotes that the user features and the random vector $\tilde{\boldsymbol{m}}$ are multiplied by elements.

To ensure that the generated feature vector $\tilde{\boldsymbol{V}}$ retains the important user semantic features, we compute the weight of a certain dimension feature based on the semantic centrality. If the d-th dimension feature frequently appears in user features with high semantic centrality, then the weight of that dimension is higher. The computational formula is as follows:

$$w^f_d = \sum_{u \in U} |v_{ud}| \cdot \xi(u), \tag{6}$$

where v_{ud} denotes the feature value of the d-th dimension of user u in the embedded view. The larger the absolute value, the more important the feature of the dimension.

To reduce the order of magnitude effect of high weight dimensions on low weight dimensions, we take the logarithms of the weights of the features, namely, $\lambda^f_d = \log w^f_d$. The probabilities are normalized by the following equation:

$$p^f_d = 1 - \min\left(\frac{\lambda^f_d - \lambda^f_{\text{min}}}{\lambda^f_{\text{max}} - \lambda^f_{\text{min}}}, p^f_\tau\right), \tag{7}$$

where λ^f_{max} and λ^f_{min} denote the maximum and minimum values, respectively, of the d-dimensional feature weights. p^f_τ is the feature truncation probability, indicating that masking features are not allowed above the probability p^f_τ, which prevents corrupting the user features of the embedded view.

The probabilities of topology-level semantic augmentation and feature-level semantic augmentation are stochastic. The embedded view $\tilde{G}$ generates two augmented views $\tilde{G}^1$, and $\tilde{G}^2$, after two rounds of random graph-data augmentation. The topology and features of both views are distinct, which can improve the algorithm's ability to adapt to noise. The augmentation process of the embedded view is as follows:

$$\tilde{G}^S = \left(U^S, E^S, \boldsymbol{V}^S\right) \rightarrow \begin{cases} \tilde{G}^{1S} = \left(U^S, E^{1S}, \tilde{\boldsymbol{V}}^{1S}\right) \\ \tilde{G}^{2S} = \left(U^S, E^{2S}, \tilde{\boldsymbol{V}}^{2S}\right) \end{cases};$$

$$\tilde{G}^T = \left(U^T, E^T, \boldsymbol{V}^T\right) \rightarrow \begin{cases} \tilde{G}^{1T} = \left(U^T, E^{1T}, \tilde{\boldsymbol{V}}^{1T}\right) \\ \tilde{G}^{2T} = \left(U^T, E^{2T}, \tilde{\boldsymbol{V}}^{2T}\right) \end{cases};$$

4.4. Multi-Head Attention Semantic Fusion

The effect of user alignment depends on the similarity of the aligned users. If there is lower semantic similarity among the aligned users, the accuracy of the algorithm will be reduced. Due to the variability of user features in social networks, the extracted semantic features cannot accurately represent users. Moreover, users and friends often share similar semantic features with each other. Therefore, we implement feature-topology adaptation fusion using a multi-head graph attention network. GAT [66] can adaptively fuse social network topology and neighbor features with different weights, and also further mine users' semantic features deeply based on a multi-head mechanism. We combine semantic centrality and GAT to increase the weight of fusing similar neighbor features. This can improve the accuracy of user alignment. This approach merges the semantic features of neighbors to enhance the features of the nodes themselves and improve the accuracy of user alignment. As some users may have excessive numbers of friends in the augmented view, fusing more neighbor features using an ordinary GNN trends to give rise to an overfitting phenomenon. Therefore, we use GAT to fuse the semantic features of our neighbors.

The semantic features of users are already available in the embedding view and the corresponding augmented view. We use v_i and v_j to denote the embedding vectors of users u_i and u_j, respectively. The attention factor for these two users is computed as follows.

$$e_{ij} = LeakyReLU\left(\alpha(Wv_i, Wv_j)\xi\left(u_j\right)\right). \tag{8}$$

This coefficient reflects the importance of user u_j to user u_i. In the equation, we use a linear transformation with parameters $W \in \mathbb{R}^{D' \times D}$, along with a self-attentive mechanism α to adaptively adjust the weights. To preserve important features, the user's semantic centrality $\xi\left(u_j\right)$ is used to measure the importance of its neighbors. Finally, a nonlinear layer LeakyReLU is added to serve as the activation function. To facilitate the comparison of attention weights across users, we normalize the attention of our neighbor u_j using the softmax function:

$$\alpha_{ij} = softmax_j(e_{ij}) = \frac{exp(e_{ij})}{\sum_{u_k \in \tilde{N}(u_i)} exp(e_{ik})}, \tag{9}$$

where $\tilde{N}(u_i)$ is the first-order neighbor of user u_i.

To improve the semantic fusion capability of the GAT, we use K independent attention heads for computation and concatenation. The computation process is as follows.

$$v_i' = \Big\|_{k=1}^{K} \sigma\left(\sum_{u_j \in \tilde{N}(u_i)} \alpha_{ij}^{(k)} W^{(k)} v_j\right), \tag{10}$$

where $\|$ indicates that the splicing operation is utilized in the features, and K indicates the number of heads in the multi-head attention.

The averaging operation is used at the final level. The computation process is as follows:

$$v_i' = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{u_j \in \tilde{N}(u_i)} \alpha_{ij}^{(k)} W^{(k)} v_\mathrm{j}\right). \tag{11}$$

The embedded view and the corresponding augmented view are semantically fused and constructed as a contrastive view, which facilitates the usage of contrastive learning among the views in the next section. The specific view transformation process is as follows:

$$\begin{cases} \tilde{G}^S = \left(U^S, E^S, \boldsymbol{V}^S\right) \to \hat{G}^S = \left(U^S, E^S, \hat{\boldsymbol{V}}^S\right) \\ \tilde{G}^T = \left(U^T, E^T, \boldsymbol{V}^T\right) \to \hat{G}^T = \left(U^T, E^T, \hat{\boldsymbol{V}}^T\right) \end{cases};$$

$$\begin{cases} \tilde{G}^{1S} = \left(U^S, E^{1S}, \tilde{\boldsymbol{V}}^{1S}\right) \to \hat{G}^{1S} = \left(U^S, E^{1S}, \hat{\boldsymbol{V}}^{1S}\right) \\ \tilde{G}^{2S} = \left(U^S, E^{2S}, \tilde{\boldsymbol{V}}^{2S}\right) \to \hat{G}^{2S} = \left(U^S, E^{2S}, \hat{\boldsymbol{V}}^{2S}\right) \end{cases};$$

$$\begin{cases} \tilde{G}^{1T} = \left(U^T, E^{1T}, \tilde{\boldsymbol{V}}^{1T}\right) \to \hat{G}^{1T} = \left(U^T, E^{1T}, \hat{\boldsymbol{V}}^{1T}\right) \\ \tilde{G}^{2T} = \left(U^T, E^{2T}, \tilde{\boldsymbol{V}}^{2T}\right) \to \hat{G}^{2T} = \left(U^T, E^{2T}, \hat{\boldsymbol{V}}^{2T}\right) \end{cases};$$

4.5. Multi-View Contrastive Learning

Computing the similarity of users requires the semantic features of users to be embedded in the Euclidean space. The effect of generated embedding vectors on user alignment depends not only on the differential semantics of the same social network, but also on the similar semantics of the aligned users in the social network to be aligned. Therefore, we perform comparison learning across in multiple comparison views. The similar features of aligned users and the different features of non-aligned users are compared in order to optimize the embedding effect, achieve user semantic feature enhancement, and improve the alignment accuracy.

We apply contrastive learning to three pairs of views: (1) source contrastive views $\hat{G}^{1S}$ and $\hat{G}^{2S}$ generated by the source social network; (2) target contrastive views $\hat{G}^{1T}$ and $\hat{G}^{2T}$ generated by the target network; (3) source-target contrastive views (alignment views) $\hat{G}^S$ and $\hat{G}^T$, constructed by the source and target social networks. In contrastive learning, it is necessary to construct positive and negative samples, which include positive samples, inter-view negative samples, and intra-view negative samples. The following description is based on the source comparison views $\hat{G}^{1S}$ and $\hat{G}^{2S}$. As these two contrastive views are constructed based on the source social network and the set of users is unchanged, we construct u_i^{1S} and u_i^{2S}, which belonging to the same real user as positive sample pairs. The user u_i^{1S} and the other users of the contrastive view $\hat{G}^{2S}$ are constructed as inter-view negative sample pairs; and the user u_i^{1S} and the other users of the contrastive view $\hat{G}^{1S}$ are constructed as intra-view negative sample pairs. The positive and negative samples of the target contrastive view $\hat{G}^{1T}$ and $\hat{G}^{2T}$ are constructed in the same way as the source contrastive view. To make the embedding vectors of aligned users more similar, we use the aligned users in the aligned views $\hat{G}^S$ and $\hat{G}^T$ as positive samples.

The contrastive views of the same social network perform contrastive learning to enhance the differential features of different users. The alignment views perform contrastive learning to enhance the similar features of known aligned users. This method effectively reduces the semantic gap and improves the alignment accuracy. By constructing the loss function based on InfoNCE Loss [64], we aim to improve the mutual information of positive samples as the goal of contrastive learning, which makes the positive sample pairs more similar. The loss function $\mathcal{L}$ of a positive sample pair $\left(u_i^{\varphi}, u_i^{\gamma}\right)$ can be defined as follows:

$$\begin{gathered} \mathcal{L}\left(u_i^{\varphi}, u_i^{\gamma}\right) = \log \frac{e^{\theta(u_i^{\varphi}, u_i^{\gamma})/\tau}}{\underbrace{e^{\theta(u_i^{\varphi}, u_i^{\gamma})/\tau}}_{\text{positive pair}} + \underbrace{\sum_{k \neq i} e^{\theta(u_i^{\varphi}, u_k^{\gamma})/\tau}}_{\text{inter-view}} + \underbrace{\sum_{k \neq i} e^{\theta(u_i^{\varphi}, u_k^{\varphi})/\tau}}_{\text{intra-view}}}; \\ s.t.(\varphi, \gamma) = \left\{\left(\hat{G}^S, \hat{G}^T\right), \left(\hat{G}^{1S}, \hat{G}^{2S}\right), \left(\hat{G}^{1T}, \hat{G}^{2T}\right)\right\}. \end{gathered} \tag{12}$$

We set a temperature coefficient τ to adjust the penalty strength of the inter-view and intra-view negative sample pairs, which prevents the user alignment model from falling into a local optimum solution in training. $\theta(u^1, u^2) = s(g(u^1), g(u^2))$ is used to compute the user similarity.

The loss function $\mathcal{L}\left(u_i^{\varphi}, u_i^{\gamma}\right)$ is computed for the loss of users in the contrastive views $\hat{G}^S, \hat{G}^{1S}, \hat{G}^{1T}$. Since the positive samples of the contrastive views $\hat{G}^S$ and $\hat{G}^T$ are aligned users and the positive samples of the other two pairs of contrastive views are the same users, these three pairs of contrastive views can be viewed as mirror-symmetric. Therefore, the loss of the contrastive views $\hat{G}^T, \hat{G}^{2S}, \hat{G}^{2T}$ can be defined as $\mathcal{L}\left(u_i^{\gamma}, u_i^{\varphi}\right)$. Our overall objective is to maximize the mean of all positive sample pairs. Accordingly, the overall loss function $\mathcal{J}$ is computed as follows.

$$\mathcal{J} = \frac{1}{2N}\sum_{i=1}^{N}\left[\mathcal{L}\left(u_{\mathrm{i}}^1, u_{\mathrm{i}}^2\right) + \mathcal{L}\left(u_{\mathrm{i}}^2, u_{\mathrm{i}}^1\right)\right], \tag{13}$$

In this section, we continually reduce the value of this loss function to optimize the embedding vectors of the contrastive views $\hat{G}^S$ and $\hat{G}^T$. User-alignment accuracy can be improved by enhancing the similar semantic features of aligned users and the difference features of non-aligned users.

4.6. User Alignment

In this section, we compute the user similarity based on the embedding vectors of views $\hat{G}^S$ and $\hat{G}^T$. If the similarity reaches the alignment threshold, the two users of different social networks are determined to be the same real-world user. The cosine distance is used to measure the similarity of users u_i^S and u_j^T. The calculation formula is as follows:

$$\mathrm{sim}\left(u_i^S, u_j^T\right) = \frac{\hat{V}_i^S \cdot \hat{V}_j^T}{\left\|\hat{V}_i^S\right\|\left\|\hat{V}_j^T\right\|}, \tag{14}$$

where $\hat{V}_i^S$ and $\hat{V}_i^T$ denote the feature vectors of users u_i and u_j in the aligned views $\hat{G}^S$ and $\hat{G}^T$, respectively.

Based on the user similarity equation, we can compute the similarity of all users in the two social networks and represent them by the matrix V^{sim}. If V_{ij}^{sim} is greater than the alignment threshold, users u_i and u_j are considered to be aligned users.

To make better use of the inter-layer link relationships, we add the top k similar aligned users that reach the similarity threshold to the known aligned user pairs M. Suppose there are two pairs of aligned users who are friends in the source network, but no link between them has been established in the target network; we can then complement the missing topology of the target network based on the aligned users. This can enhance user semantic features and improve user-alignment accuracy.

5. Experiments

Experiments were conducted on real-world social networks to evaluate the effectiveness of the proposed SENUA model when dealing with the user alignment problem. Moreover, an ablation study and comparisons of similarity before and after the experiment are conducted and discussed.

5.1. Dataset and Experimental Setup

5.1.1. Dataset

To prove the effectiveness of the algorithm, the Douban–Weibo datasets [43] and DBLP17-DBLP19 datasets [44] are used to validate the experiment. Douban–Weibo dataset contains social network topology, user attributes, and user-generated contents. DBLP is a computer science bibliography that includes author's name, school, city, and papers. The statistics are presented in Table 2.

Table 2. Statistics of the datasets.

Datasets	Networks	Users	Edges	Min Degree	Ave Degree	Max Degree	Anchors	Source
Social networks	Douban	9734	200,467	1	43	1723	9514	[43]
	Weibo	9514	196,978	1	34	2501		
coauthor networks	DBLP17	9086	51,700	2	5.7	144	2832	[44]
	DBLP19	9325	47,775	2	5.1	138		

Similar users in the same social network and similar users across social networks can affect alignment. To visualize the interference, we took 50 pairs of aligned users from both datasets and represent the user similarity with a heat map, as shown in Figure 5. Green represents the Douban–Weibo datasets, and blue represents the DBLP17-DBLP19 datasets. The labels of the 6 subgraphs indicate the social networks in which users are registered. The scale of the coordinate axis represents the user ID. Figure 5a,b,d,e represent the comparison of users in the same social network. The users of the horizontal and vertical axes are in accordance. Figure 5c,f show the comparison of aligned users in the social network to be aligned. The diagonal line indicates the similarity of the aligned user pairs. The deeper the color in the graph, the higher the degree of similarity. The figure shows that there are a large number of highly similar users in the same social network, which can interfere with user alignment. Compared with Douban–Weibo, the interference user color is lighter and the alignment user color is deeper in DBLP17-DBLP19. It is easier to achieve user alignment in DBLP17-DBLP19 datasets. We aimed to improve the color depth of the diagonal lines in Figure 5c,f. Increasing the color depth of the diagonal in Figure 5c,f is our goal. We ensured the accuracy of user alignment by reducing noise in social networks and optimizing embedding effects.

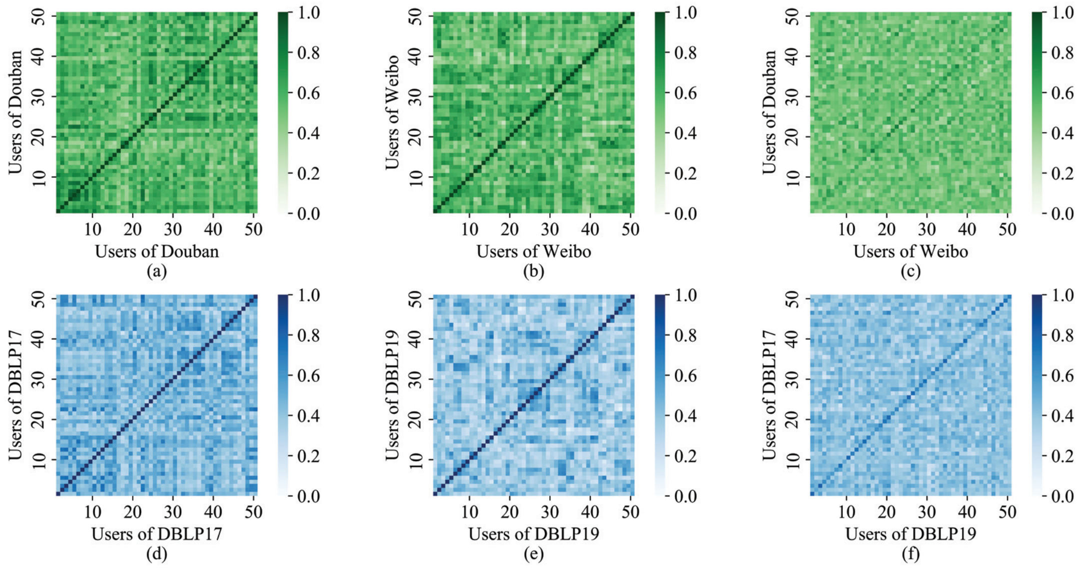


Figure 5. Visualization of user similarity before training: (**a**) Douban; (**b**) Weibo; (**c**) Douban-Weibo; (**d**) DBLP17; (**e**) DBLP19; (**f**) DBLP17-DBLP19.

5.1.2. Parameter Settings

After treating the following–followed relationship as an undirected edge, we expand the directed edges of the dataset into undirected edges. We extracted user semantic features from user attributes, UGCs, and user check-ins with an embedding dimension of 256. The projection before user alignment comprises two fully connected layers, where the hidden dimension was 512. The edge sampling probability in graph augmentation was 0.3. The feature masking probability was 0.2. The temperature parameter τ was 0.2 for contrastive learning based on InfoNCE.

5.1.3. Evaluation Indicators

Hit-Precisionk was used as the performance metric for this experiment. This metric represents the average score of the top k positive samples in the prediction results, which can represent the prediction accuracy of our algorithm. The computation formula is as follows:

$$\mathrm{Hit} - \mathrm{Precision}@k = \frac{1}{|\mathcal{C}|}\sum_{x\in\mathcal{C}}\frac{k-(\mathrm{hit}(x)-1)}{k}, \tag{15}$$

where $\mathcal{C}$ indicates the set of candidate users, and $\mathrm{hit}(x)$ indicates the location of the positive sample among the top-k recommended candidate users.

5.2. Baseline Methods

To verify the performance of this algorithm, we chose the following user alignment algorithms as the baselines.

- GraphUIL [21] encodes the local and global network structures, then achieves user alignment by minimizing the difference before and after reconstruction and the match loss of anchor users.
- INFUNE [43] performs information fusion based on the network topology, attributes, and generated contents of users. Adaptive fusion of neighborhood features based on a graph neural network is performed to improve user-alignment accuracy.
- MAUIL [44] uses three layers of user attribute embedding and one layer of network topology embedding to mine user features. User alignment is performed after mapping user features from two social networks to the same space.
- SNAME [67] effectively mines user features based on three embedding methods: intentional neural network, fuzzy c-mean clustering, and graph drawing embedding.

5.3. Experimental Results

Figure 6 presents the heat map of user similarity of two datasets after SENUA training. The diagonal lines indicate the similarity of aligned users, and the other regions indicate the similarity of non-aligned users. Compared with the pretraining Figure 5c,f, it can be observed that the diagonal colors are significantly deeper, and the colors of the remaining positions are significantly lighter. Overall, SENUA reduces the interference of highly similar users on the user-alignment effect and accordingly improves the alignment effect. Figure 7 presents the similarity comparison of aligned users before and after training. The horizontal axis represents the users to be aligned, and the vertical axis is the user similarity. As the figure makes clear, the similarity of aligned users is significantly improved after training, and the similarity changes are more stable. The multi-head attention semantic fusion makes the embedding vector more stable, and contrastive learning in aligned views enhances the similarity of aligned users, which plays an important role in improving user alignment accuracy.

To demonstrate the effectiveness of our algorithm, we compare the user-alignment accuracy of each algorithm based on the Douban–Weibo and DBLP17-DBLP19 datasets, as shown in Figure 8. The horizontal axis is the ratio of the training set to the total dataset. The vertical axis is the performance metric hit-precision30, which indicates the existence probability of aligned users among the 30 similar users recommended for the user.

This can represent the prediction accuracy of aligned users in different social networks. The results show that SENUA outperformed other baseline methods in user alignment, with an average improvement of 6.27%. This shows that multi-view graph contrastive learning can improve the effectiveness of social network user alignment. The overall performance in Figure 8b is significantly better than that in Figure 8a. User alignment can also achieve better results when the training ratio of DBLP is 10%. In Figure 5d,e, there are fewer highly similar users in the same social network, and the user alignment is less affected by noise interference. Compared with Figure 5c, the diagonal line of Figure 5f is darker, and other areas are lighter in color. In the DBLP17-DBLP19 dataset, the aligned users are subject to less interference, which results in better user alignment in this dataset. Our algorithm is not optimal when the training ratio is 10%. As the training ratio increases, the alignment accuracy continues to improve. Better user alignment is obtained when the training ratio is high. Graph attention networks and contrastive learning all require sufficient data to accurately discover the feature patterns of users. We reduce the local noise interference by multi-level user feature representation, and then effectively enhance the semantic features of users by semantic fusion and semantic contrasting.

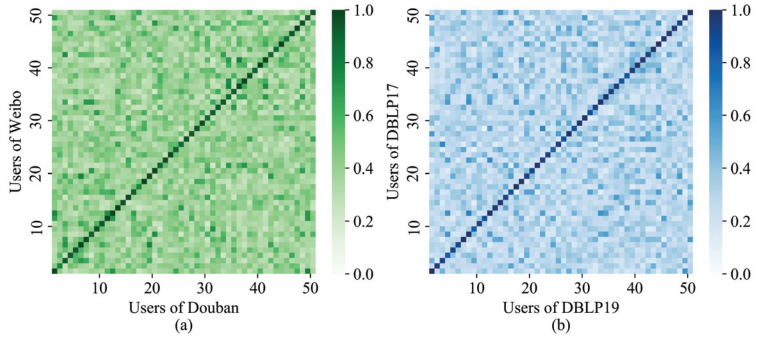


Figure 6. Visualization of trained user similarity: (**a**) Douban-Weibo; (**b**) DBLP17-DBLP19.

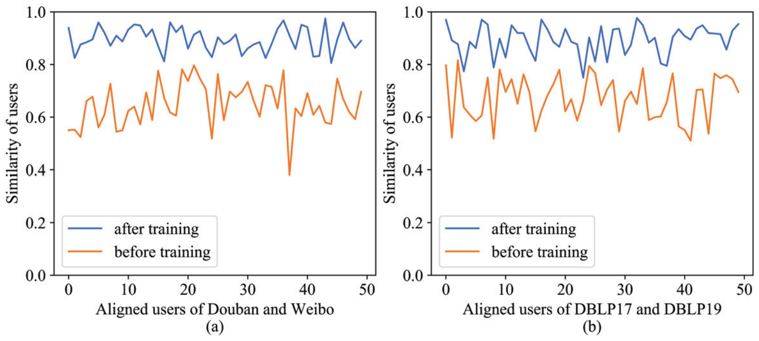


Figure 7. Comparison of user similarity before and after training: (**a**) Douban-Weibo; (**b**) DBLP17-DBLP19.

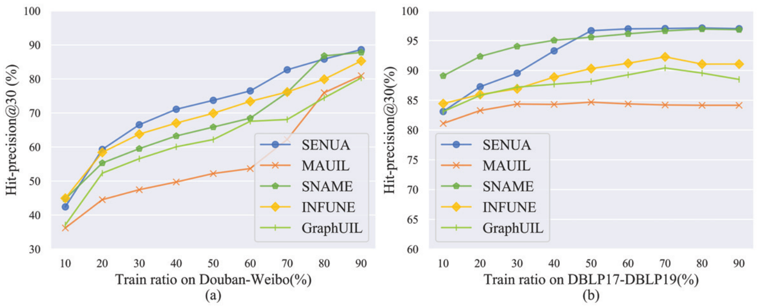


Figure 8. Comparisons with baselines: (**a**) Douban-Weibo; (**b**) DBLP17-DBLP19.

We fixed the ratio of the training set to the total dataset to 0.9. Subsequently, the effects of GAT and graph-data augmentation on user alignment were measured, as shown in Figure 9. The accuracy decreases slightly at one layer of GAT and without graph-data augmentation. The graph-data augmentation improves the peak accuracy of our algorithm, although the impact on the accuracy is small. With semantic centrality attention, graph-data augmentation can reduce noise interference in social networks while preserving important features and topology. After the number of layers of GAT is adjusted from one to two, the user-alignment accuracy decreases significantly. If the number of GAT layers is too high, users will fuse more neighborhood features, which will reduce the feature variability among users and lead to difficulties in user alignment.

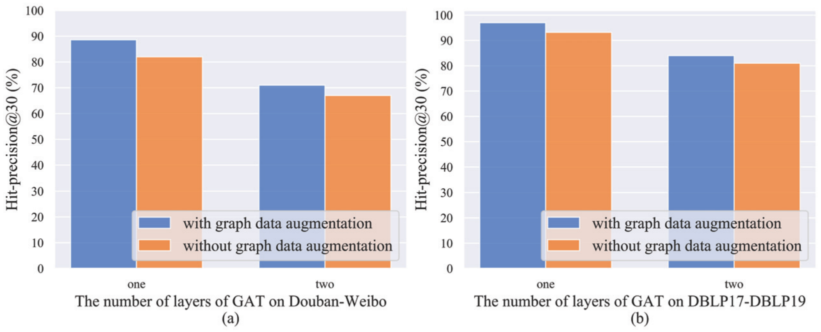


Figure 9. The impacts of GAT and graph-data augmentation on user alignment: (**a**) Douban-Weibo; (**b**) DBLP17-DBLP19.

6. Conclusions

In this paper, we proposed a semantic-enhancement-based social network user alignment algorithm, SENUA, to reduce the semantic-gap problem caused by social network variability. The interference of local semantic noise on user alignment is reduced through the use of multi-level semantic representations. To reduce the feature noise and topological noise in the aligned views, we improved the algorithm's ability to adapt to semantic noise by using graph-data augmentation. Appropriate weights are assigned to the user's semantic features and topology with the semantic centrality of the user, which enables important semantic features to be preserved. The embedding vectors of users are optimized based on multi-head graph attention networks and multi-view contrastive learning. By increasing the embedding distance between users in the same social network views while decreasing the embedding distance of aligned users in the aligned views, we can effectively enhance the semantic features of users and improve the alignment effect. To verify the performance of our model, we compared it with several baseline methods on the Douban–Weibo and DBLP17-DBLP19. Experimental results show that the effectiveness of SENUA is 6.27% higher than that of the baseline methods on average. As these results show, SENUA enhances user alignment through semantic enhancement in many ways. However, semantic fusion and multi-view contrastive learning generate a high computing overhead. In our future work, we plan to improve the efficiency and accuracy of user alignment based on causal inference.

Author Contributions: Conceptualization, Y.H. and P.Z.; formal analysis, P.Z., H.W. and H.M.; funding acquisition, L.X., H.W. and H.M.; methodology, Y.H. and Q.Z.; project administration, Y.H.; resources, H.W. and H.M.; software, P.Z.; supervision, Q.Z. and L.X.; validation, Y.H., P.Z. and Q.Z.; visualization, P.Z. and Q.Z.; writing—original draft, Y.H.; writing—review and editing, Y.H., Q.Z. and L.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work is fully supported by the National Natural Science Foundation of China (62171180, 62072158, 62272146), the Program for Innovative Research Team in University of Henan Province (21IRTSTHN015), in part by the Key Science and the Research Program in University of Henan Province (21A510001), Henan Province Science Fund for Distinguished Young Scholars (222300420006), and the Science and Technology Research Project of Henan Province under Grant (222102210001).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study. Written informed consent has been obtained from the patient(s) to publish this paper.

Data Availability Statement: Not applicable.

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

References

1. Magnani, M.; Hanteer, O.; Interdonato, R.; Rossi, L.; Tagarelli, A. Community Detection in Multiplex Networks. *ACM Comput. Surv.* **2022**, *54*, 1–35. [CrossRef]
2. Pan, Y.; He, F.; Yu, H. Learning Social Representations with Deep Autoencoder for Recommender System. *World Wide Web* **2020**, *23*, 2259–2279. [CrossRef]
3. Kou, H.; Liu, H.; Duan, Y.; Gong, W.; Xu, Y.; Xu, X.; Qi, L. Building Trust/Distrust Relationships on Signed Social Service Network through Privacy-Aware Link Prediction Process. *Appl. Soft Comput.* **2021**, *100*, 106942. [CrossRef]
4. Li, S.; Yao, L.; Mu, S.; Zhao, W.X.; Li, Y.; Guo, T.; Ding, B.; Wen, J.R. Debiasing Learning Based Cross-domain Recommendation. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual, 14–18 August 2021; ACM: Singapore, 2021; pp. 3190–3199. [CrossRef]
5. Zhang, A.; Chen, Y. A Real-Time Detection Algorithm for Abnormal Users in Multi Relationship Social Networks Based on Deep Neural Network. In *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*; Liu, S., Ma, X., Eds.; Springer International Publishing AG: Cham, Switzerland, 2022; Volume 416, pp. 179–190. [CrossRef]
6. Wang, Y.; Shen, H.; Gao, J.; Cheng, X. Learning Binary Hash Codes for Fast Anchor Link Retrieval across Networks. In Proceedings of the World Wide Web Conference (WWW '19), San Francisco, CA, USA, 13–17 May 2019; pp. 3335–3341. [CrossRef]
7. Qin, T.; Liu, Z.; Li, S.; Guan, X. A Two-Stagse Approach for Social Identity Linkage Based on an Enhanced Weighted Graph Model. *Mob. Netw. Appl.* **2020**, *25*, 1364–1375. [CrossRef]
8. Yuan, Z.; Yan, L.; Xiaoyu, G.; Xian, S.; Sen, W. User Naming Conventions Mapping Learning for Social Network Alignment. In Proceedings of the 2021 13th International Conference on Computer and Automation Engineering (ICCAE), Melbourne, Australia, 20–22 March 2021; pp. 36–42. [CrossRef]
9. Xiao, Y.; Hu, R.; Li, D.; Wu, J.; Zhen, Y.; Ren, L. Multi-Level Graph Attention Network Based Unsupervised Network Alignment. In Proceedings of the 2021 IEEE 46th Conference on Local Computer Networks (LCN), Edmonton, AB, Canada, 4–7 October 2021; pp. 217–224. [CrossRef]
10. Tang, R.; Jiang, S.; Chen, X.; Wang, W.; Wang, W. Network Structural Perturbation against Interlayer Link Prediction. *Knowl.-Based Syst.* **2022**, *250*, 109095. [CrossRef]
11. Cai, C.; Li, L.; Chen, W.; Zeng, D. Capturing Deep Dynamic Information for Mapping Users across Social Networks. In Proceedings of the 2019 IEEE International Conference on Intelligence and Security Informatics (ISI), Shenzhen, China, 1–3 July 2019; pp. 146–148. [CrossRef]
12. Fang, Z.; Cao, Y.; Liu, Y.; Tan, J.; Guo, L.; Shang, Y. A Co-Training Method for Identifying the Same Person across Social Networks. In Proceedings of the 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Montreal, ON, Canada, 14–16 November 2017; pp. 1412–1416. [CrossRef]

13. Zhong, Z.X.; Cao, Y.; Guo, M.; Nie, Z.Q.; Aaai. CoLink: An Unsupervised Framework for User Identity Linkage. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence/30th Innovative Applications of Artificial Intelligence Conference/8th AAAI Symposium on Educational Advances in Artificial Intelligence, New Orleans, LO, USA, 2–7 February 2018; Association Advancement Artificial Intelligence: Palo Alto, CA, USA, 2018; pp. 5714–5721.
14. Zeng, W.; Tang, R.; Wang, H.; Chen, X.; Wang, W. User Identification Based on Integrating Multiple User Information across Online Social Networks. *Secur. Commun. Netw.* **2021**, *2021*, 5533417. [CrossRef]
15. Qu, Y.; Ma, H.; Wu, H.; Zhang, K.; Deng, K. A Multiple Salient Features-Based User Identification across Social Media. *Entropy* **2022**, *24*, 495. [CrossRef] [PubMed]
16. Feng, J.; Zhang, M.Y.; Wang, H.D.; Yang, Z.Y.; Zhang, C.; Li, Y.; Jin, D.P.; Assoc Comp, M. DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data. In Proceedings of the World Wide Web Conference (WWW '19), San Francisco, CA, USA, 13–17 May 2019;Association of Computing Machinery: New York, NY, USA, 2019; pp. 459–469. [CrossRef]
17. Xue, H.; Sun, B.; Si, C.; Zhang, W.; Fang, J. DBUL: A User Identity Linkage Method across Social Networks Based on Spatiotemporal Data. In Proceedings of the 2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI), Washington, DC, USA, 1–3 November 2021; pp. 1461–1465. [CrossRef]
18. Zhou, F.; Li, C.; Wen, Z.; Zhong, T.; Trajcevski, G.; Khokhar, A. Uncertainty-aware Network Alignment. *Int. J. Intell. Syst.* **2021**, *36*, 7895–7924. [CrossRef]
19. Tang, R.; Miao, Z.; Jiang, S.; Chen, X.; Wang, H.; Wang, W. Interlayer Link Prediction in Multiplex Social Networks Based on Multiple Types of Consistency Between Embedding Vectors. *IEEE Trans. Cybern.* **2021**, 1–14. *early access.* [CrossRef]
20. Zheng, C.; Pan, L.; Wu, P. JORA: Weakly Supervised User Identity Linkage via Jointly Learning to Represent and Align. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, 1–12. *early access.* [CrossRef]
21. Zhang, W.; Shu, K.; Liu, H.; Wang, Y. Graph Neural Networks for User Identity Linkage. *arXiv* **2019**, arXiv:1903.02174.
22. Chen, X.; Song, X.; Peng, G.; Feng, S.; Nie, L. Adversarial-Enhanced Hybrid Graph Network for User Identity Linkage. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual, 11–15 July 2021; Association of Computing Machinery: New York, NY, USA, 2021; pp. 1084–1093. [CrossRef]
23. Deng, K.; Xing, L.; Zheng, L.; Wu, H.; Xie, P.; Gao, F. A User Identification Algorithm Based on User Behavior Analysis in Social Networks. *IEEE Access* **2019**, *7*, 47114–47123. [CrossRef]
24. Li, Y.; Peng, Y.; Ji, W.; Zhang, Z.; Xu, Q. User Identification Based on Display Names Across Online Social Networks. *IEEE Access* **2017**, *5*, 17342–17353. [CrossRef]
25. Li, Y.; Cui, H.; Liu, H.; Li, X. Display Name-Based Anchor User Identification across Chinese Social Networks. In Proceedings of the 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON, Canada, 11–14 October 2020; pp. 3984–3989. [CrossRef]
26. Li, Y.J.; Zhang, Z.; Peng, Y. A Solution to Tweet-Based User Identification Across Online Social Networks. In *Advanced Data Mining and Applications, Lecture Notes in Artificial Intelligence*; Springer International Publishing Ag: Cham, Swizerland, 2017; Volume 10604, pp. 257–269. [CrossRef]
27. Sharma, V.; Dyreson, C. LINKSOCIAL: Linking User Profiles Across Multiple Social Media Platforms. In Proceedings of the 2018 IEEE International Conference on Big Knowledge (ICBK), Singapore, 17–18 November 2018; IEEE: New York, NY, USA, 2018; pp. 260–267. [CrossRef]
28. Zhou, X.; Yang, J. Matching User Accounts Based on Location Verification across Social Networks. *Rev. Int. Metod. Numer. Para Calc. Diseno Ing.* **2020**, *36*, 7. [CrossRef]
29. Kojima, K.; Ikeda, K.; Tani, M. Short Paper: User Identification across Online Social Networks Based on Similarities among Distributions of Friends' Locations. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 4085–4088. [CrossRef]
30. Xing, L.; Deng, K.; Wu, H.; Xie, P.; Gao, J. Behavioral Habits-Based User Identification Across Social Networks. *Symmetry* **2019**, *11*, 19. [CrossRef]
31. Qu, Y.; Xing, L.; Ma, H.; Wu, H.; Zhang, K.; Deng, K. Exploiting User Friendship Networks for User Identification across Social Networks. *Symmetry* **2022**, *14*, 110. [CrossRef]
32. Amara, A.; Taieb, M.A.H.; Aouicha, M.B. Identifying I-Bridge Across Online Social Networks. In Proceedings of the 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), Hammamet, Tunisia, 30 October–3 November 2017; pp. 515–520. [CrossRef]
33. Yu, J.; Gao, M.; Li, J.; Yin, H.; Liu, H. Adaptive Implicit Friends Identification over Heterogeneous Network for Social Recommendation. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, Torino, Italy, 22–26 October 2018; Association of Computing Machinery: New York, NY, USA, 2018; pp. 357–366. [CrossRef]
34. Feng, S.; Shen, D.; Nie, T.; Kou, Y.; He, J.; Yu, G. Inferring Anchor Links Based on Social Network Structure. *IEEE Access* **2018**, *6*, 17340–17353. [CrossRef]
35. Zhang, D.; Yin, J.; Zhu, X.; Zhang, C. Network Representation Learning: A Survey. *IEEE Trans. Big Data* **2020**, *6*, 3–28. [CrossRef]
36. Zhou, X.; Liang, X.; Du, X.; Zhao, J. Structure Based User Identification across Social Networks. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1178–1191. [CrossRef]

37. Zhou, X.; Liang, X.; Zhao, J.; Zhiyuli, A.; Zhang, H. An Unsupervised User Identification Algorithm Using Network Embedding and Scalable Nearest Neighbour. *Clust. Comput.* **2019**, *22*, 8677–8687. [CrossRef]
38. Zhang, J.; Yuan, Z.; Xu, N.; Chen, J.; Wang, J. Two-Stage User Identification Based on User Topology Dynamic Community Clustering. *Complexity* **2021**, *2021*, 5567351. [CrossRef]
39. Cheng, A.; Liu, C.; Zhou, C.; Tan, J.; Guo, L. User Alignment via Structural Interaction and Propagation. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [CrossRef]
40. Li, W.; He, Z.; Zheng, J.; Hu, Z. Improved Flower Pollination Algorithm and Its Application in User Identification Across Social Networks. *IEEE Access* **2019**, *7*, 44359–44371. [CrossRef]
41. Ma, J.; Qiao, Y.; Hu, G.; Huang, Y.; Wang, M.; Sangaiah, A.K.; Zhang, C.; Wang, Y. Balancing User Profile and Social Network Structure for Anchor Link Inferring Across Multiple Online Social Networks. *IEEE Access* **2017**, *5*, 12031–12040. [CrossRef]
42. Yang, Y.; Yu, H.; Huang, R.; Ming, T. A Fusion Information Embedding Method for User Identity Matching Across Social Networks. In Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Guangzhou, China, 8–12 October 2018; pp. 2030–2035. [CrossRef]
43. Chen, S.Y.; Wang, J.H.; Du, X.; Hu, Y.Q. A Novel Framework with Information Fusion and Neighborhood Enhancement for User Identity Linkage. In *Frontiers in Artificial Intelligence and Applications, Proceedings of the 24th European Conference on Artificial Intelligence (ECAI), Online/Santiago de Compostela, Spain, 29 August–8 September 2020*; Ios Press: Amsterdam, The Netherlands, 2020; Volume 325, pp. 1754–1761. [CrossRef]
44. Chen, B.; Chen, X. MAUIL: Multilevel Attribute Embedding for Semisupervised User Identity Linkage. *Inf. Sci.* **2022**, *593*, 527–545. [CrossRef]
45. Shu, J.; Shi, J.; Liao, L. Link Prediction Model for Opportunistic Networks Based on Feature Fusion. *IEEE Access* **2022**, *10*, 80900–80909. [CrossRef]
46. Lin, J.C.W.; Shao, Y.; Zhou, Y.; Pirouz, M.; Chen, H.C. A Bi-LSTM Mention Hypergraph Model with Encoding Schema for Mention Extraction. *Eng. Appl. Artif. Intell.* **2019**, *85*, 175–181. [CrossRef]
47. Lin, J.C.W.; Shao, Y.; Fournier-Viger, P.; Hamido, F. BILU-NEMH: A BILU Neural-Encoded Mention Hypergraph for Mention Extraction. *Inf. Sci.* **2019**, *496*, 53–64. [CrossRef]
48. Lin, J.C.W.; Shao, Y.; Djenouri, Y.; Yun, U. ASRNN: A Recurrent Neural Network with an Attention Model for Sequence Labeling. *Knowl.-Based Syst.* **2021**, *212*, 106548. [CrossRef]
49. Lin, J.C.W.; Shao, Y.; Zhang, J.; Yun, U. Enhanced Sequence Labeling Based on Latent Variable Conditional Random Fields. *Neurocomputing* **2020**, *403*, 431–440. [CrossRef]
50. Shao, Y.; Lin, J.C.W.; Srivastava, G.; Jolfaei, A.; Guo, D.; Hu, Y. Self-Attention-Based Conditional Random Fields Latent Variables Model for Sequence Labeling. *Pattern Recognit. Lett.* **2021**, *145*, 157–164. [CrossRef]
51. Chugh, M.; Whigham, P.A.; Dick, G. Stability of Word Embeddings Using Word2Vec. In Proceedings of the AI 2018: Advances in Artificial Intelligence, Wellington, New Zealand, 11–14 December 2018; Mitrovic, T., Xue, B., Li, X., Eds.; Springer International Publishing Ag: Cham, Switzerlnad, 2018; Volume 11320, pp. 812–818. [CrossRef]
52. Kang, H.; Yang, J. Performance Comparison of Word2vec and fastText Embedding Models. *J. Digit. Contents Soc.* **2020**, *21*, 1335–1343. [CrossRef]
53. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for aLanguage Understanding. *arXiv* **2018**, arXiv:1810.04805.
54. Hamilton, W.L. Graph Representation Learning. *Synth. Lect. Artif. Intell. Mach. Learn.* **2020**, *14*, 1–159. [CrossRef]
55. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; IEEE: Seattle, WA, USA, 2020; pp. 9726–9735. [CrossRef]
56. Gao, T.; Yao, X.; Chen, D. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online/Punta Cana, Dominican Republic, 7–11 November 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 6894–6910. [CrossRef]
57. You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; Shen, Y. Graph Contrastive Learning with Augmentations. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 5812–5823.
58. Hassani, K.; Khasahmadi, A.H. Contrastive Multi-View Representation Learning on Graphs. In Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, 12–18 July 2020; PMLR: 2020; pp. 4116–4126.
59. Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; Wang, L. Graph Contrastive Learning with Adaptive Augmentation. In Proceedings of the Web Conference 2021, Online, 19–23 April 2021; ACM: Ljubljana Slovenia, 2021; pp. 2069–2080. [CrossRef]
60. Liu, B.; Zhang, P.; Lu, T.; Gu, N. A Reliable Cross-Site User Generated Content Modeling Method Based on Topic Model. *Knowl.-Based Syst.* **2020**, *209*, 106435. [CrossRef]
61. Ye, M.; Yin, P.; Lee, W.C.; Lee, D.L. Exploiting Geographical Influence for Collaborative Point-of-Interest Recommendation. In Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information—SIGIR '11, Beijing, China, 24–28 July 2011; ACM Press: New York, NY, USA, 2011; p. 325. [CrossRef]

62. Tan, H.; Shao, W.; Wu, H.; Yang, K.; Song, L. A Sentence Is Worth 128 Pseudo Tokens: A Semantic-Aware Contrastive Learning Framework for Sentence Embeddings. *arXiv* **2022**. [CrossRef]
63. Liu, Y.; Ao, X.; Dong, L.; Zhang, C.; Wang, J.; He, Q. Spatiotemporal Activity Modeling via Hierarchical Cross-Modal Embedding. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 462–474. [CrossRef]
64. van den Oord, A.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2019**, arXiv:1807.03748.
65. Yu, J.; Yin, H.; Gao, M.; Xia, X.; Zhang, X.; Viet Hung, N.Q. Socially-Aware Self-Supervised Tri-Training for Recommendation. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual Event/Singapore, 14–18 August 2021; ACM: Rochester, NY, USA, 2021; pp. 2084–2092. [CrossRef]
66. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. *arXiv* **2018**, arXiv:1710.10903.
67. Le, V.V.; Tran, T.K.; Nguyen, B.N.T.; Nguyen, Q.D.; Snasel, V. Network Alignment across Social Networks Using Multiple Embedding Techniques. *Mathematics* **2022**, *10*, 3972. [CrossRef]

Article

PBQ-Enhanced QUIC: QUIC with Deep Reinforcement Learning Congestion Control Mechanism

Zhifei Zhang [1,2,3,*], Shuo Li [1,2,3], Yiyang Ge [1,2,3], Ge Xiong [2,4], Yu Zhang [5] and Ke Xiong [1,2,3,*]

1 Engineering Research Center of Network Management Technology for High Speed Railway of Ministry of Education, School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
2 Collaborative Innovation Center of Railway Traffic Safety, Beijing Jiaotong University, Beijing 100044, China
3 National Engineering Research Center of Advanced Network Technologies, Beijing Jiaotong University, Beijing 100044, China
4 China Software and Technical Service Co., Ltd., Beijing 100081, China
5 Institute of Economics and Energy Supply and Demand, State Grid Energy Research Institute Co., Ltd., Beijing 102209, China
* Correspondence: zhfzhang@bjtu.edu.cn (Z.Z.); kxiong@bjtu.edu.cn (K.X.)

Abstract: Currently, the most widely used protocol for the transportation layer of computer networks for reliable transportation is the Transmission Control Protocol (TCP). However, TCP has some problems such as high handshake delay, head-of-line (HOL) blocking, and so on. To solve these problems, Google proposed the Quick User Datagram Protocol Internet Connection (QUIC) protocol, which supports 0-1 round-trip time (RTT) handshake, a congestion control algorithm configuration in user mode. So far, the QUIC protocol has been integrated with traditional congestion control algorithms, which are not efficient in numerous scenarios. To solve this problem, we propose an efficient congestion control mechanism on the basis of deep reinforcement learning (DRL), i.e., proximal bandwidth-delay quick optimization (PBQ) for QUIC, which combines traditional bottleneck bandwidth and round-trip propagation time (BBR) with proximal policy optimization (PPO). In PBQ, the PPO agent outputs the congestion window (CWnd) and improves itself according to network state, and the BBR specifies the pacing rate of the client. Then, we apply the presented PBQ to QUIC and form a new version of QUIC, i.e., PBQ-enhanced QUIC. The experimental results show that the proposed PBQ-enhanced QUIC achieves much better performance in both throughput and RTT than existing popular versions of QUIC, such as QUIC with Cubic and QUIC with BBR.

Keywords: QUIC; congestion control; deep reinforcement learning; BBR

Citation: Zhang, Z.; Li, S.; Ge, Y.; Xiong, G.; Zhang, Y.; Xiong, K. PBQ-Enhanced QUIC: QUIC with Deep Reinforcement Learning Congestion Control Mechanism. *Entropy* **2022**, *25*, 294. https://doi.org/10.3390/e25020294

Academic Editor: Donald J. Jacobs

Received: 19 December 2022
Revised: 23 January 2023
Accepted: 2 February 2023
Published: 4 February 2023

1. Introduction

At present, computer networks remain the essential platform for information interaction, where the transport layer plays an influential role. The emergence of modern applications, such as video live broadcast and Internet of Things (IoT), has imposed higher demands on throughput, packet loss rate, and network delay. The development of wireless transmission technologies such as 5G and WiFi has made the network environment even more complex. The Transmission Control Protocol (TCP), as a widely used protocol, experiences problems such as large handshake delay, head-of-line (HOL) blocking, and protocol solidification, which increasingly affect network performance. Compared with TCP, the User Datagram Protocol (UDP) is more efficient for real-time transmission but lacks reliability.

In 2012, Google proposed the Quick UDP Internet Connection (QUIC) protocol [1], which realizes orderly, quick, and reliable transport services in user mode based on UDP. The QUIC protocol reduces the handshake latency to zero round-trip time (RTT) by caching ServerConfig. Moreover, the QUIC protocol natively supports multiplexing techniques where streams on the same connection do not influence each other, which solves the HOL blocking problem. Additionally, the QUIC protocol decouples the congestion control algorithm from the protocol stack, which is more flexible than TCP. Currently, the QUIC protocol has become one focus of researchers in both academia and industry.

For both TCP and QUIC protocols, congestion control is one of the core mechanisms. The basic logic of TCP congestion control contains congestion sensing and the corresponding disposal pattern, which improves the network utilization by estimating the bandwidth-delay product (BDP) of the link and adjusting the send rate of clients. Since the first congestion control algorithm, Tahoe [2], was proposed, there have been dozens of congestion control algorithms developed that are suitable for different scenarios. The operation rules of traditional congestion control algorithms are shown in Figure 1. Congestion control algorithms work on the sender, sensing the network state and changing the congestion window or transmission rate. Currently widely used control algorithms are heuristic and cannot be optimally executed in dynamic and changeable network environments.

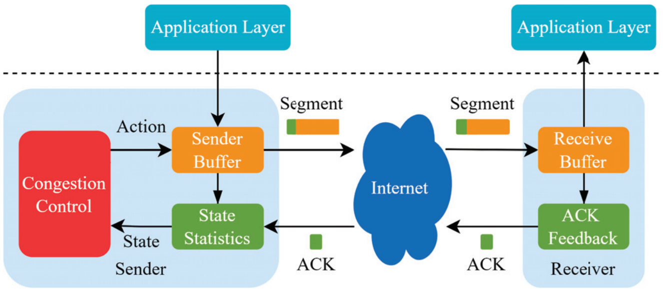


Figure 1. A schematic of congestion control in TCP.

Recently, researchers have started to design TCP congestion control algorithms using machine learning algorithms. There are numerous versions of TCP based on supervised learning, such as TCP with DeePCCI [3] and TCP with LSTM-PTI [4]. Naturally, supervised-learning-based algorithms are only used for passive congestion identification, and training them requires a great deal of labeled data. As the network environment changes, so do the network characteristics. In this case, the supervised learning congestion control algorithm is not suitable for the changing network environment and may no longer be effective. In light of the advantages of the reinforcement learning (RL) method in sequence decision [5,6], researchers all over the world have been trying to apply it to congestion control. So far, some well-known congestion control methods, such as Remy [7], performance-oriented congestion control (PCC)-Vivace [8], Q-learning TCP (QTCP) [9], Orca [10] and Aurora [11] algorithms have been proposed for the TCP protocol. For the QUIC protocol, researchers worldwide have proposed some modified versions of QUIC, i.e., QUIC-go [12], Microsoft QUIC (MsQUIC) [13], Modified QUIC [14], and Quiche [15]. QUIC-go is an implementation of the QUIC protocol in Go. It keeps up to date with the latest request for comments (RFC) and is easy to deploy. MsQUIC is a Microsoft implementation of the QUIC protocol. It optimizes the maximal throughput and minimal latency. Modified QUIC proposes a modification to the handshaking mechanism to minimize the time required to update the CWnd, which results in a smooth variation in the CWnd. This makes modified QUIC protocol easy to deploy and achieves better performance in terms of throughput. Quiche is an implementation of the QUIC protocol in Rust and C. It performs well across different applications, such nginx and curl.

However, for these popular versions of QUIC, only heuristic congestion control algorithms are applied. Most heuristic congestion control algorithms commonly perceive the network state based on simple network models and adopt a fixed strategy, i.e., shrinking the window when packet loss occurs or RTT increases, and augmenting the window when an acknowledge character (ACK) is received. Such a simple strategy makes heuristic congestion control algorithms unable to achieve optimal results in dynamic and changeable network environments. For example, in the WiFi scenario, where obstacles and human activity affect network quality, packet loss events do not imply the occurrence of congestion. However, heuristic congestion control algorithms cannot distinguish that. This issue leads to poor performance of the QUIC protocol in terms of throughput and latency when the network settings change. Therefore, to enable the QUIC to achieve better performance in different network environments, we tried to design DRL-enhanced QUIC using the merits of DRL in environment awareness and decision making.

The contributions of this work can be summarized as follows:

- First, we developed a novel congestion control mechanism, referred to as proximal bandwidth-delay quick optimization (PBQ) by combining proximal policy optimization (PPO) [16] with traditional BBR [17]. It is able to effectively improve the convergence speed and link stability during the training phase. We then applied the presented PBQ to the QUIC protocol and formed a new version of QUIC, i.e., PBQ-enhanced QUIC, which aims to enhance its adaptivity and throughput performance.
- Second, for the purpose of reducing the action space and establishing the connection of the values of the congestion window (CWnd) for each interaction, we used continuous action ratio as the output action of PBQ's agent. Additionally, in the design of the utility function, we used a relatively simple formulation of the objective function as the optimization objective and introduced a delay constraint. By doing so, our proposed PBQ-enhanced QUIC achieves higher throughput and maintains a low RTT.
- Third, we built a reinforcement learning environment for QUIC on the network simulation software ns-3, where we trained and tested PBQ-enhanced QUIC. The experimental results showed that our presented PBQ-enhanced QUIC achieves much better RTT and throughput performance than existing popular versions of QUIC, such as QUIC with BBR and QUIC with Cubic [18].

The organization of the rest of this paper is as follows: Section 2 describes the QUIC protocol, including the handshake phase, multiplexing, connection migration, and congestion control. Section 3 gives the design of PBQ and PBQ-enhanced QUIC. Section 4 introduces the training link and testing link and evaluates the performance of PBQ-enhanced QUIC compared with that of QUIC using different congestion control algorithms. Finally, Section 5 gives the conclusions.

2. QUIC Protocol

2.1. The Basic Concept of QUIC

The QUIC protocol is a secure and reliable data transfer protocol first proposed by Google in 2012, which was released as a standardized version of QUIC, RFC 9000 [19], by the Internet Engineering Task Force (IETF). The bottom layer of QUIC is UDP, which makes it compatible with current network protocols. At the same time, the QUIC protocol is compatible with traditional congestion control algorithms in TCP. The QUIC protocol has numerous improvements over TCP, mainly in handshaking, multiplexing, and connection migration.

2.2. Handshake

TCP is a plaintext transport protocol and transport layer security (TLS) is required to implement data encryption transmission. In TLS 1.2, two RTTs are required for the TLS handshake phase and three RTTs for the total handshake delay. Under TLS 1.3, the TLS handshake phase requires one RTT, and the handshake phase still requires two RTTs.

Comparatively, the QUIC protocol optimizes the handshake process. In the handshake phase, QUIC incorporates the transmission parameters into the encryption part. When the first connection is established, the client sends authentication- and encryption-related information to the server. It takes only one RTT to establish a connection. Alternatively, we can use quantum key distribution to replace the original encryption information exchange [20]. When the connection is established once again, the client uses the preshared key to establish an encrypted connection with the server during zero RTT.

2.3. Multiplexing

Multiplexing is multiple streams on a connection at the same time, commonly on a Hyper Text Transfer Protocol (HTTP) stream. For HTTP/2, a TCP connection can support multiple HTTP streams. However, HOL blocking in the TCP protocol causes interference between HTTP streams, which affects their performance. In advance, multiple streams can be created on a single QUIC connection, which reduces the handshake frequency. The QUIC protocol is implemented based on UDP. Streams on the same QUIC connection are independent of each other, which solves the HOL blocking problem.

2.4. Connection Migration

One TCP connection is known to be identified by a quintuple, i.e., <IP_{source}, $PORT_{source}$, IP_{dest}, $PORT_{dest}$>, where IP_{source} denotes the source IP address, $PORT_{source}$ denotes the source port, IP_{dest} denotes the destination IP address, and $PORT_{dest}$ denotes the destination port [21]. If one term in the quadruple changes, the connection is broken. By using a 64-bit random ID as the connection identifier, the QUIC protocol avoids the effect of network switching. The simple process of connection migration is shown in Figure 2.

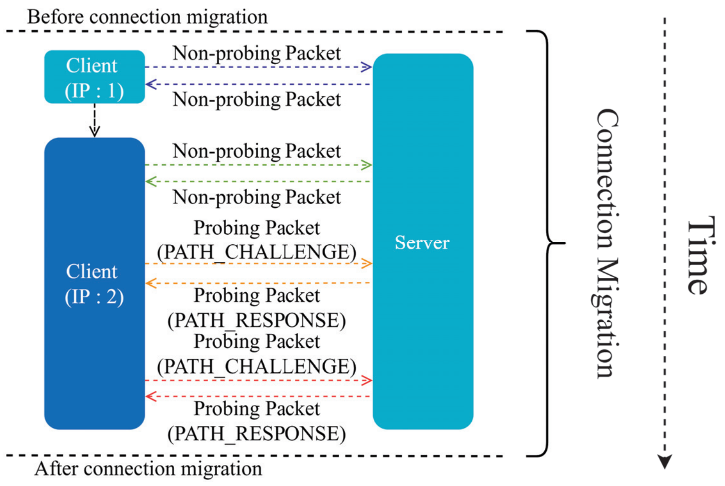

Before connection migration
Client (IP : 1)
Non-probing Packet
Non-probing Packet
Non-probing Packet
Non-probing Packet
Probing Packet (PATH_CHALLENGE)
Probing Packet (PATH_RESPONSE)
Probing Packet (PATH_CHALLENGE)
Probing Packet (PATH_RESPONSE)
Client (IP : 2)
Server
Connection Migration
Time
After connection migration

Figure 2. Connection migration.

Particularly, before the client's IP changes, the endpoints communicate via a nondetection packet. After IP changes, the path detection is performed to verify the reachability before connection migration starts. If the path detection fails, the connection migration cannot be performed; otherwise, a fresh connection is established. The IP address is verified between the client and the server, and the endpoint holding the latest IP address of the peer migrates. When migration occurs, the congestion control part and the reliable transport protocol estimation part of the path need to be reset. After the connection migrates, it sends a nonprobe packet.

2.5. Congestion Control in QUIC

The QUIC protocol is a reliable UDP-based data transfer protocol, which makes it compatible with existing network protocols. The congestion control algorithm in TCP is implemented in kernel mode, and if an upgrade is performed, the kernel needs to be recompiled. However, the QUIC protocol straddles kernel mode and user mode, and its congestion control part is in user mode. Thus, it can be easily upgraded. In particular, the

QUIC protocol supports the configuration of different congestion control algorithms for applications, making it possible to optimize for specific applications.

3. The PBQ-QUIC

In this section, we describe PBQ and its application to QUIC to form the PBQ-enhanced QUIC.

3.1. The Basic Idea of PBQ

For clarity, Figure 3 shows the workflow of PBQ. PBQ combines PPO with BBR to accurately identify the network state. It can effectively identify congestion and packet loss events. PBQ is divided into three main parts: Monitoring, Decision, and Pacer modules. The Monitoring module collects environment state and sends it to the Decision module. The Decision module outputs actions, including a_t and the pacing rate, according to the network state. The Pacer module distributes actions to the corresponding senders. In the Decision module, the Controller distributes the network state to the PPO and the BBR. The PPO part outputs a_t according to s_t using the new policy and sends it to the environment through Pacer. The replay memory stores the experience of past interactions. After multiple interactions, the policy networks and the value network are updated using the past interaction experience. Inspired by Orca, our Decision module adopts a two-level regulation mechanism, which is shown in Figure 4. The underlying BBR algorithm performs a classical decision-making behavior driven by ACK. The DRL agent evaluates the network congestion and predicts the BDP according to the state output by the Monitoring module.

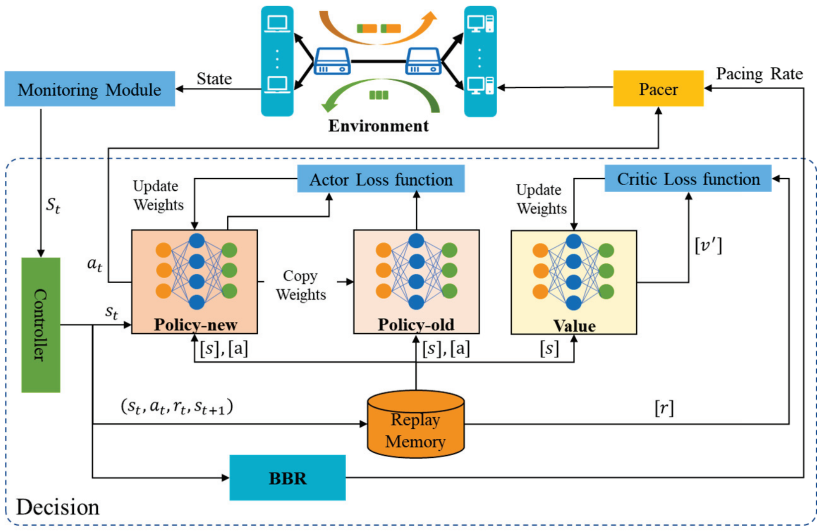


Figure 3. The framework of PBQ.

- Monitoring Module

We set the network state collection interval to 100 ms, and the designed state quantity is shown in Table 1. Because the interval is not strictly equal to 100 ms, we also count the intervals. The remaining states are the statistics within each interval.

Table 1. State statistics description.

State	Description
$CWnd_t$	Current congestion window
$interval_t$	Cata update interval
$deliveryRate_t$	Average delivering rate (throughput)
RTT_t	Averaged RTT
$packetLoss_t$	Average loss rate of packets

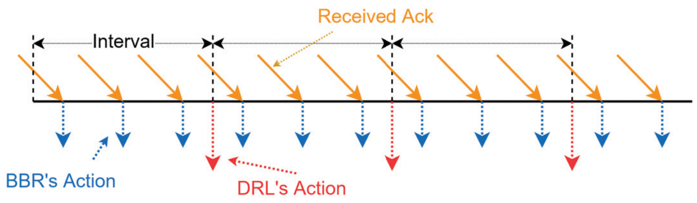


Figure 4. The two-level control logic of PBQ's enhanced QUIC.

- Reward Function Design

We trained PBQ using a linear function with constraints. First, we define a linear utility function as $\alpha * deliveryRate_t - \beta * packetLoss_t$, where α denotes the coefficient of $deliveryRate_t$, $deliveryRate_t$ denotes the delivery rate in time t, β denotes the coefficient of $packetLoss_t$, and $packetLoss_t$ denotes the packet loss rate in time t. Then, we formulate an optimization problem to maximize the linear utility function under a given RTT constraint, i.e., $RTT_t \leq minRTT_t\ , t = 1, 2, 3 \ldots, n$, where RTT_t denotes the last RTT in time t, and $minRTT_t$ denotes the minimum RTT from the establishment of the connection to time t.

$$\begin{array}{ll} \max & \alpha * deliveryRate_t - \beta * packetLoss_t \\ s.t. & RTT_t \leq minRTT_t\ , t = 1, 2, 3 \ldots, n \quad . \end{array} \tag{1}$$

We use the maximization objective as the base reward function:

$$R_t = \alpha * deliveryRate_t - \beta * packetLoss_t \tag{2}$$

With the delay constraint, the final reward function is

$$\begin{gathered} R_t = \alpha * deliveryRate_t - \beta * packetLoss_t \\ R_t = \left\{ \begin{array}{cc} R_t & RTT_t \leq \gamma * minRTT_t \\ R_t - \eta & RTT_t > \gamma * minRTT_t \end{array} \right. \end{gathered} \tag{3}$$

where γ denotes the penalty threshold, and η denotes the penalty term when $RTT_t > \gamma * minRTT_t$.

- Action Design

Reinforcement learning action types can be divided into two categories, discrete and continuous actions. We first used discrete actions, but analyzing the experimental results, we found that when using discrete actions, the output of the agent considerably fluctuated, and it was difficult to achieve better performance in the initial stages of the interaction. In general, the discrete action does not favor the early stability of the network links. The final solution in this study draws on the additive increase multiplicative decrease (AIMD) [22] idea in traditional congestion control, and we set the action output as $CWndRatio$. The mapping relationship between $CWndRatio$ and the congestion window is as follows:

$$newCWnd = CWnd * 2^{CWndRatio} \tag{4}$$

The relationship between the values of CWnd for each interaction was established, which reduces the fluctuation of the action and ensures the excellent performance of the model.

- Learning Algorithm for PBQ

For the DRL agent, PPO is employed, which is a classical actor-critic algorithm. We used tanh as the activation function in both actor and critic neural networks. Because the state transitions in traditional congestion algorithms are not complex, we preferred to build a simpler neural network and train on it. Finally, we constructed a three-layer neural network, where the hidden layer contains 64 neurons.

For clarity, the PBQ training phase is summarized in Algorithm 1. On lines 1–2 of Algorithm 1, the training episode $Episodes$ is set and the replay memory D is initialized, which is used to store state, action and reward. The policy parameters θ_{new} and value parameters ϕ are also initialized with random weights. We set policy parameters θ_{old} equal to θ_{new}. Then, on lines 4–16, at each episode, we first reset the environment and obtain the initial state s_0 and initial reward r_0. We use PPO to regularly output α_t according to the new policy $\pi_{\theta_{new}}$. We map α_t to $CWnd_t$, which is also the estimated link BDP. Then, $CWnd_t$ is preformed into environment. In each step, PBQ collects network states s_t and s_{t+1}, action value a_t, and the corresponding reward r_t, and updates the replay memory D. The policy parameters θ_{new} and value functions ϕ are updated when the number of steps accumulates to the policy update threshold $update\_timestep$. The underlying BBR method is driven by ACK to control the specific pacing rate of the clients.

Algorithm 1 PBQ's learning algorithm.

```
1: Initialize replay memory $D$ and training episode $Episodes$
2: Initialize policy parameters $\theta_{new}$, value parameters $\phi$, $\theta_{old} \leftarrow \theta_{new}$
3: for $episode$ = 1 to $Episodes$ do
4:     Reset environment, obtain initial state $s_0$ and initial reward $r_0$
5:     $i \leftarrow 1$
6:     for t = 1 to $steps$ do
7:         if time to play PPO action then
8:             Run policy $\pi_{\theta_{new}}$, obtain PPO action $\alpha_t$
9:             Map $\alpha_t$ to $CWnd_t$, $CWnd_t = CWnd_{t-1} * 2^{\alpha_t}$
10:            Perform $CWnd_t$
11:        else
12:            Play a BBR action, perform $pacingRate_t$
13:        end if
14:        Collect $\{s_t, a_t, r_t, s_{t+1}\}$, update $D$
15:        $t = t + 1$
16:        if $t$ % $update\_timestep == 0$ then
17:            $\theta_{old} \leftarrow \theta_{new}$, update $\theta_{new}$ and $\phi$
18:        end if
19:    end for
20: end for
```

- Pacer

The Pacer module is responsible for distributing the actions output by the Decision module to the corresponding clients. The action distribution is driven by an update callback for the CWnd and an update callback for the pacing rate. For pacing rate, when the client receives an ACK or packet loss occurs, the underlying BBR adjusts the pacing rate and triggers a pacing rate update callback. For CWnd, the Decision module receives the updated network state from the Monitoring module; the RL part outputs α and calculates the current CWnd; then, the callback function is called to update the CWnd on the client via Pacer module. As shown in Figure 5, the Pacer module distributes actions to the corresponding clients based on their ids.

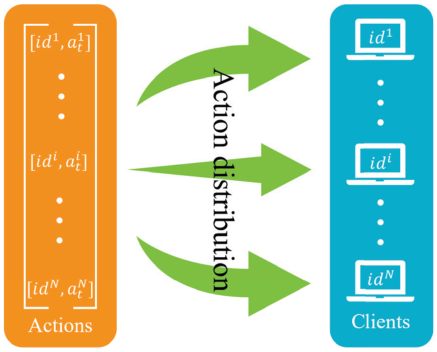


Figure 5. Action distribution in the Pacer module.

3.2. *PBQ-Enhanced QUIC*

Figure 6 shows the differences in congestion control between PBQ-enhanced QUIC and traditional QUIC. QUIC uses Cubic as the default congestion control algorithm. Cubic is a heuristic algorithm that is driven by events at the sender side, including ACK receipt, packet loss, etc. The policy of Cubic is based on packet loss events, which leads to poor performance in scenarios where packet loss is present. PBQ is a combination of the DRL method PPO and BBR with the advantages of both. Our deployment adopts the client/server (C/S) mode, which only modifies QUIC in client and adopts the standard QUIC implementation in the server. We deeply integrated PBQ into the QUIC client. We added the Monitoring module for network state statistics, the Decision module for rate control, and the Pacer module for rate distribution. The Decision and Pacer modules are asynchronously executed, and they do not affect the behavior of the clients.

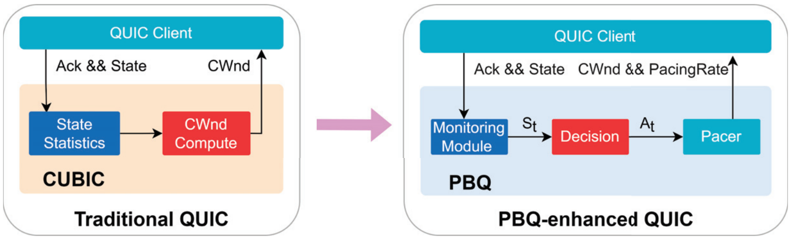


Figure 6. PBQ-enhanced QUIC.

We specifically designed states and actions. The Monitoring module periodically collects the state of the environment and sends the processed data to the Decision module. We collected a number of environment states, including pacing rate, current RTT and minimum RTT, interval, packet loss rate, CWnd, and so on. The output of the Pacer module includes the CWnd and pacing rate of the senders. The BBR regulates the pacing rate of the clients, and the DRL agent gives the congestion window as the predicted BDP. Then, the Pacer module changes the sending behavior of the clients.

4. Simulation Performance

We tested the PBQ-enhanced QUIC in various scenarios on the open-source network simulator ns-3 [23] and compared the results with those of QUIC using different congestion control algorithms. We wrote simulation scripts in C++ and implemented our DRL agent based on MindSpore in python. Then, we ran the comparison experiments on a Dell PowerEdge R840 server with 256 G memory, 64 cores, and a GeForce RTX 3090 GPU. We tested the model on a large number of links with different parameters and analyzed the sensitivity of PBQ-enhanced QUIC to the number of link flows and packet loss rate.

When the testing and training links were quite different, the PBQ-enhanced QUIC still performed well.

4.1. Our Simulation Environment

The application of DRL methods is inseparable from the interaction with the environment. Several researchers have implemented their own simulation environments, but they all target the TCP domain, and there is no mature solution for the simulation environment of the QUIC protocol alone. At the same time, we think that the development of a network simulation environment is a huge and meticulous project that should be focused on the study of congestion control algorithms; therefore, we chose the ns-3 platform to build the training and testing environment. We built our own simulation environment based on NS3-gym [24] and Quic-NS-3 [25] to test PBQ-enhanced QUIC.

4.2. Training

We also applied the PPO method to the congestion control part of QUIC. We trained PPO and PBQ on the link shown in Figure 7. For training, we set N to two, which means that two traffic flows shared a link. The link parameters are shown in Table 2. Our model converged in 50 epochs, each consisting of 1200 steps. In contrast, the experimental results showed that after combining with BBR, the model converged faster, and the throughput and RTT performance were more stable in the initial interaction.

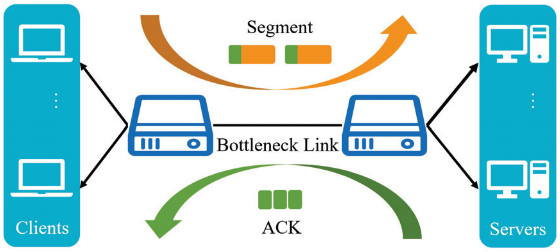
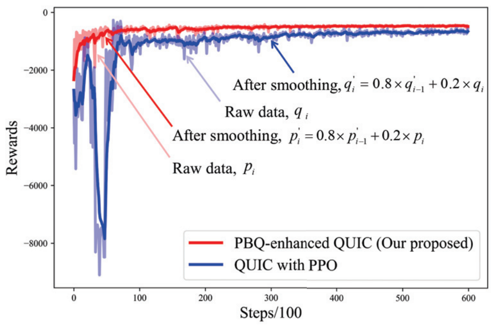


Figure 7. The simulation link of PBQ.

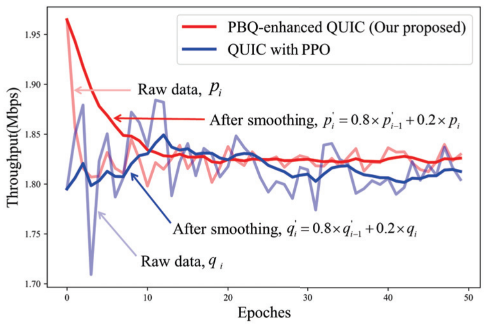
Table 2. Training link.

Attribute	Value
Number of flows	2
Bottleneck bandwidth	2 Mbps
RTT	30 ms
Queue capacity	75 Kilobytes
Queue scheduling algorithm	First Input First Output (FIFO)

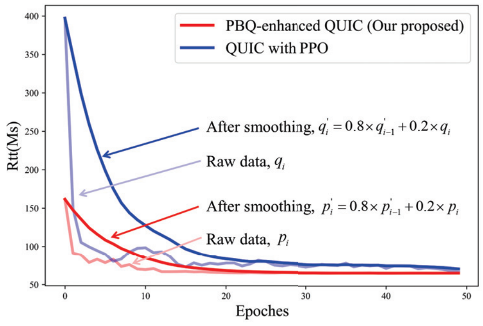
As illustrated in Figure 8b,c, the algorithm we propose can effectively improve the stability of the pretraining action while ensuring the quality of the link. As shown in Figure 8a, the combination with the BBR effectively speeds up the speed of convergence, which solves the problems we described before.



(**a**)



(**b**)



(**c**)

Figure 8. Training process after 600 episodes: (**a**) training reward; (**b**) training throughput; (**c**) training RTT.

4.3. Testing

Similarly, we compared PBQ-enhanced QUIC with QUIC using current learning-based congestion control algorithm Remy and heuristic algorithms such as Bic [26], Cubic, Low Extra Delay Background Transport (LEDBAT) [27], NewReno [28], Vegas [29], and BBR. [id = S.L.] The link parameters are shown in Table 3.

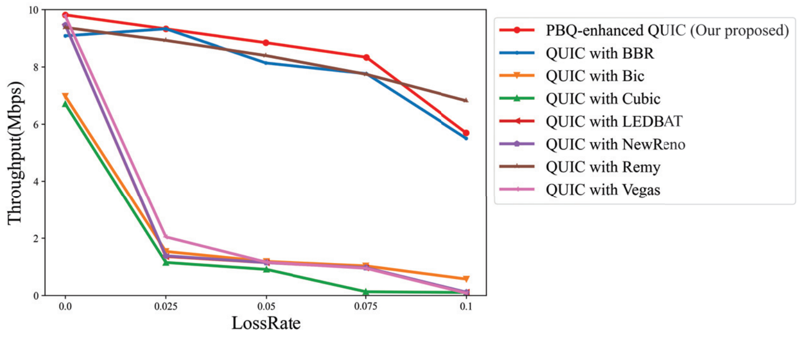
Table 3. Testing link.

Attribute	Value
Number of flows	2∼8
Bottleneck bandwidth	10 Mbps
RTT	100 ms
Packet loss rate	0%∼10%
Queue capacity	75 Kilobytes
Queue scheduling algorithm	FIFO

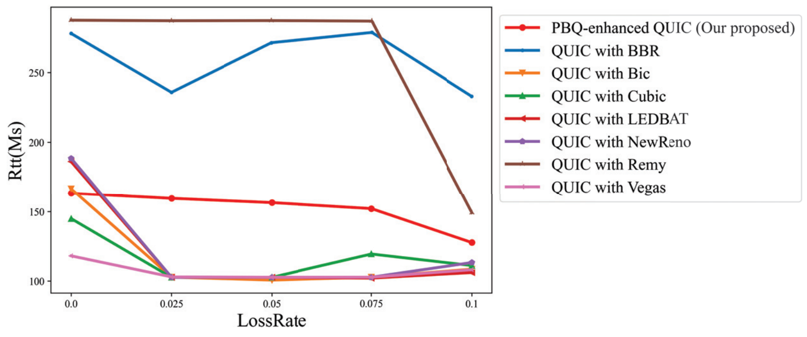
In traditional algorithms, BBR is based on network modeling, while the remaining algorithms take packet loss events as congestion occurrence signals. In this part, we still used the dumbbell link in Figure 7. To demonstrate the adaptability of our algorithm, there was a large deviation between our test and training links. Specifically, we determined the performance of PBQ-enhanced QUIC in different scenarios, including its excellent adaptation and tolerance to packet loss and number of flows.

4.4. Packet Loss Rate

We tested the proposed algorithm using a previously trained model in a different network scenario. The throughput changes with the packet loss rate as illustrated in Figure 9a, and the RTT is shown in Figure 9b.



(**a**)



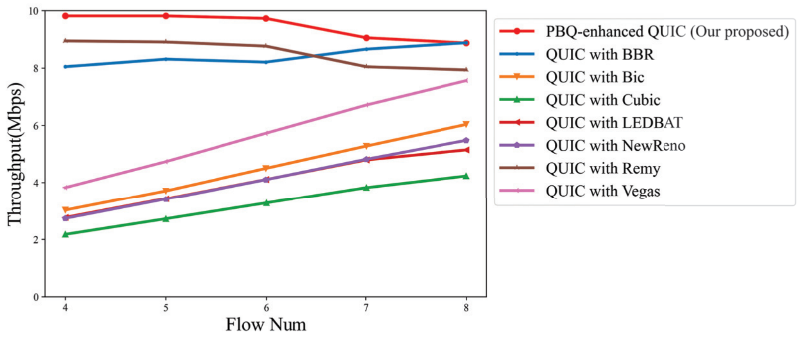
(**b**)

Figure 9. Link quality of schemes across different packet loss rates with flow number 2: (**a**) throughput; (**b**) RTT.

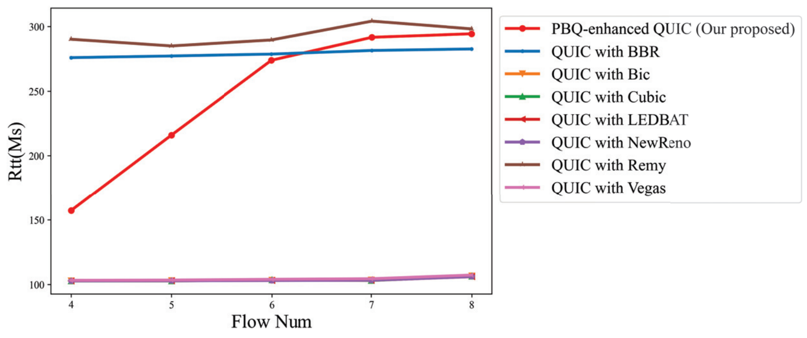
When packet loss occurs, QUIC with congestion control algorithms that take packet loss events as a signal, such as QUIC with Cubic and QUIC with NewReno, frequently enter fast recovery, resulting in low throughput and low RTT. While QUIC with BBR requires constant probing of the network, QUIC with Remy relies on a state-action table. It is difficult for them to balance throughput and RTT. Our algorithm can efficiently identify the overall state of the network and can output relevant actions by periodically collecting the network state. As the packet loss rate increases, the throughput performance of our algorithm is less affected, and it is the best among the different packet loss rates. Furthermore, our proposed algorithm achieves higher throughput analso has better packet loss tolerance ability than QUIC with Remy and QUIC with BBR.

4.5. Flow Number

Then, the packet loss rate was set to 2.5%, and we modified the number of flows and compared the throughput and delay performance of different QUIC implementations. The results are shown as Figure 10a,b. Figure 10a represents the total link throughput. Figure 10b indicates the average RTT.



(**a**)



(**b**)

Figure 10. Link quality of schemes for different flow numbers with packet loss rate of 2.5%: (**a**) throughput; (**b**) RTT.

Owing to the presence of random packet loss, algorithms that take packet loss events as congestion occurrence signals fail to accurately identify the cause of congestion. In the policies of these congestion control methods, the current link can only support a low throughput; thus, the throughput is at a low level. In this case, the flows barely affect each other. As a result, as shown in Figure 10a, the total throughput of the link linearly increases with the number of flows. They cannot accurately estimate the link BDP and occupy the full link bandwidth, so the average link RTT is close to the default RTT. Compared with QUIC with BBR and QUIC with Remy, PBQ-enhanced QUIC reduces the detection process of link BDP and achieves excellent throughput and RTT by optimizing the reward function.

5. Conclusions

In this study, we first developed a two-level regulatory mechanism PBQ, which combines the heuristic algorithms BBR and DRL to approximate PPO. The convergence rate of the model is accelerated by using BBR to estimate the specific transmission rate. Moreover, we proposed PBQ-enhanced QUIC, an implementation of QUIC that uses PBQ as a congestion control algorithm. Unlike QUIC with heuristic congestion control algorithms, our QUIC implementation learns congestion control rules from experience by using RL signals. Therefore, our QUIC implementation can be better adapted to various network settings.

As shown in Section 4.2, the combination with BBR can effectively speed up the convergence of the PPO during the training phase on the premise of ensuring link quality. As shown in Sections 4.2 and 4.4, compared with other QUIC versions, PBQ-enhanced QUIC achieves higher throughput performance in various network settings. PBQ-enhanced also has better RTT performance than QUIC with BBR and QUIC with Remy. We think that combining DRL methods with traditional algorithms to design congestion control mechanisms for QUIC will be a major trend in the future, and PBQ-enhanced QUIC provides a new idea to do so.

In future work, we plan to build a testbed in real-world networks and test PBQ-enhanced QUIC on it. We will improve the PBQ-enhanced QUIC based on its performance in real-world networks. Moreover, different applications have different requirements on network metrics, so we plan to design congestion control algorithms for different applications by taking advantage of the feature that the congestion control part of QUIC is implemented in user mode. We expect different applications to perform efficiently on the same network.

Author Contributions: Conceptualization, Z.Z. and S.L.; methodology, Z.Z. and S.L.; software, S.L. and Y.G.; validation, S.L. and G.X.; formal analysis, Z.Z. and S.L.; investigation, S.L. and Y.Z.; resources, Z.Z. and G.X.; data curation, S.L. and Y.Z.; writing—original draft preparation, S.L.; writing—review and editing, S.L. and K.X.; visualization, S.L. and Y.G.; supervision, Z.Z. and K.X.; project administration, Z.Z; funding acquisition, Z.Z. and K.X. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded in part by CAAI-Huawei MindSpore Open Fund Grant No. CAAIXSJLJJ-2021-055A and in part by the Fundamental Research Funds for the Central Universities Grant No. 2022JBZY021.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

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

Abbreviations

The following abbreviations are used in this manuscript:

TCP	Transmission Control Protocol
HOL	Head-of-line
QUIC	Quick UDP Internet Connection
RTT	Round-trip time
DRL	Deep reinforcement learning
PBQ	Proximal bandwidth-delay quick optimization
BBR	Bottleneck bandwidth and round-trip propagation time
PPO	Proximal policy optimization

CWnd	Congestion window
IoT	Internet of Things
UDP	User Datagram Protocol
BDP	Bandwidth-delay product
RL	Reinforcement learning
PCC	Performance-oriented congestion control
QTCP	Q-learning TCP
MsQUIC	Microsoft QUIC
RFC	Request for comments
ACK	Acknowledge character
IETF	Internet Engineering Task Force
TLS	Transport layer security
HTTP	Hyper Text Transfer Protocol
AIMD	Additive increase multiplicative decrease
C/S	Client/server
LEDBAT	Low extra delay background transport

References

1. Langley, A.; Riddoch, A.; Wilk, A.; Vicente, A.; Krasic, C.; Zhang, D.; Yang, F.; Kouranov, F.; Swett, I.; Iyengar, J.; et al. The QUIC Transport Protocol: Design and Internet-Scale Deployment. In Proceedings of the ACM SIGCOMM, Los Angeles, CA, USA, 21–25 August 2017.
2. Jacobson, V. Congestion avoidance and control. *ACM SIGCOMM Comput. Commun. Rev.* **1988**, *18*, 314–329. [CrossRef]
3. Sander, C.; Rüth, J.; Hohlfeld, O.; Wehrle, K. Deepcci: Deep learning-based passive congestion control identification. In Proceedings of the 2019 Workshop on Network Meets AI & ML, Beijing, China, 23 August 2019; pp. 37–43.
4. Chen, X.; Xu, S.; Chen, X.; Cao, S.; Zhang, S.; Sun, Y. Passive TCP identification for wired and wireless networks: A long-short term memory approach. In Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, 24–28 June 2019; pp. 717–722.
5. Brown, N.; Sandholm, T. Safe and nested subgame solving for imperfect-information games. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11 .
6. Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv* **2017**, arXiv:1712.01815.
7. Winstein, K.; Balakrishnan, H. TCP ex Machina: Computer-Generated Congestion Control. In Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM, Hong Kong, China, 27 August 2013.
8. Dong, M.; Meng, T.; Zarchy, D.; Arslan, E.; Gilad, Y.; Godfrey, P.B.; Schapira, M. PCC Vivace: Online-Learning Congestion Control. In Proceedings of the 15th Usenix Symposium on Networked Systems Design and Implementation (Nsdi'18) 2018, Renton, WA, USA, 9–11 April 2018; pp. 343–356.
9. Li, W.; Zhou, F.; Chowdhury, K.R.; Meleis, W. QTCP: Adaptive Congestion Control with Reinforcement Learning. *IEEE Netw. Sci. Eng.* **2019**, *6*, 445–458. [CrossRef]
10. Abbasloo, S.; Yen, C.-Y.; Chao, H.J. Classic Meets Modern: a Pragmatic Learning-Based Congestion Control for the Internet. In Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication, Online, 10–14 August 2020; pp. 632–647.
11. Jay, N.; Rotman, N.; Godfrey, B.; Schapira, M.; Tamar, A. A Deep Reinforcement Learning Perspective on Internet Congestion Control. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 3050–3059.
12. Clemente L, Seemann M. Quic-go. Available online: https://github.com/lucas-clemente/quic-go (accessed on 3 August 2016).
13. Microsoft. msquic. Available online: https://github.com/microsoft/msquic (accessed on 26 October 2019).
14. Kharat, P.; Kulkarni, M. Modified QUIC protocol with congestion control for improved network performance. *IET Commun.* **2021**, *15*, 1210–1222. [CrossRef]
15. Cloudflare. Quiche. Available online: https://github.com/cloudflare/quiche (accessed on 1 February 2021).
16. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017** arXiv:1707.06347.
17. Cardwell, N.; Cheng, Y.; Gunn, C.S.; Yeganeh, S.H.; Jacobson, V. BBR: Congestion-Based Congestion Control: Measuring bottleneck bandwidth and round-trip propagation time. *Queue* **2016**, *14*, 20–53. [CrossRef]
18. Ha, S.; Rhee, I.; Xu, L. CUBIC: A New TCP-Friendly High-Speed TCP Variant. *ACM SIGOPS Oper. Syst. Rev.* **2008**, *42*, 64–74. [CrossRef]
19. Iyengar, J.; Thomson, M. *RFC 9000 QUIC: A UDP-Based Multiplexed and Secure Transport. Omtermet Emgomeeromg Task Force*; ACM Digital Library: New York, NY, USA, 2021.
20. Yan, P.; Yu, N. The QQUIC Transport Protocol: Quantum-Assisted UDP Internet Connections. *Entropy* **2022**, *24*, 1488. [CrossRef]
21. Kurose, J.F.; Ross, K.W. *Computer Networking: A Top-Down Approach*, 7th ed.; Pearson FT Press: Upper Saddle River, NJ, USA, 2016.

22. Floyd, S.; Kohler, E.; Padhye, J. *Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)*; RFC: Marina del Rey, CA, USA, 2006; pp. 2070–1721.
23. Henderson, T.R.; Lacage, M.; Riley, G.F.; Dowell, C.; Kopena, J. Network simulations with the ns-3 simulator. *Sigcomm Demonstr*. **2008**, *14*, 527.
24. Gawlowicz, P.; Zubow, A. ns-3 meets OpenAI Gym: The Playground for Machine Learning in Networking Research. In Proceedings of the 22nd International Acm Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, Miami Beach, FL, USA, 25–29 November 2019; pp. 113–120. [CrossRef]
25. De Biasio, A.; Chiariotti, F.; Polese, M.; Zanella, A.; Zorzi, M. A QUIC Implementation for ns-3. In Proceedings of the 2019 Workshop on NS-3, University of Florence, Florence, Italy, 19 June 2019.
26. Xu, L.; Harfoush, K.; Rhee, I. Binary increase congestion control (BIC) for fast long-distance networks. In Proceedings of the IEEE INFOCOM 2004, Hong Kong, China, 7–11 March 2004; pp. 2514–2524.
27. Rossi, D.; Testa, C.; Valenti, S.; Muscariello, L. LEDBAT: the new BitTorrent congestion control protocol. In Proceedings of the 19th International Conference on Computer Communications and Networks, Zurich, Switzerland, 2–5 August 2010; pp. 1–6.
28. Floyd, S.; Henderson, T.; Gurtov, A. The NewReno Modification to TCP's Fast Recovery Algorithm; RFC: Marina del Rey, CA, USA, 2004; pp. 2070–1721.
29. Brakmo, L.S.; O'Malley, S.W.; Peterson, L.L. TCP Vegas: New techniques for congestion detection and avoidance. *ACM SIGCOMM Comput. Commun. Rev*. **1994**, *24*, 24–35. [CrossRef]

entropy

Article

Use of Composite Multivariate Multiscale Permutation Fuzzy Entropy to Diagnose the Faults of Rolling Bearing

Qiang Yuan [1,2,*], Mingchen Lv [2,*], Ruiping Zhou [1], Hong Liu [2], Chongkun Liang [2] and Lijiao Cheng [2]

1 School of Naval Architecture, Ocean and Energy Power Engineering, Wuhan University of Technology, Wuhan 430070, China
2 School of Naval Architecture and Maritime, Zhejiang Ocean University, Zhoushan 316022, China
* Correspondence: yuanqiang@zjou.edu.cn (Q.Y.); mingchenlv@outlook.com (M.L.)

Abstract: The study focuses on the fault signals of rolling bearings, which are characterized by nonlinearity, periodic impact, and low signal-to-noise ratio. The advantages of entropy calculation in analyzing time series data were combined with the high calculation accuracy of Multiscale Fuzzy Entropy (MFE) and the strong noise resistance of Multiscale Permutation Entropy (MPE), a multivariate coarse-grained form was introduced, and the coarse-grained process was improved. The Composite Multivariate Multiscale Permutation Fuzzy Entropy (CMvMPFE) method was proposed to solve the problems of low accuracy, large entropy perturbation, and information loss in the calculation process of fault feature parameters. This method extracts the fault characteristics of rolling bearings more comprehensively and accurately. The CMvMPFE method was used to calculate the entropy value of the rolling bearing experimental fault data, and Support Vector Machine (SVM) was used for fault diagnosis analysis. By comparing with MPFE, the Composite Multiscale Permutation Fuzzy Entropy (CMPFE) and the Multivariate Multiscale Permutation Fuzzy Entropy (MvMPFE) methods, the results of the calculations show that the CMvMPFE method can extract rolling bearing fault characteristics more comprehensively and accurately, and it also has good robustness.

Keywords: fault diagnostic method; multivariate multiscale permutation fuzzy entropy; composite multivariate multiscale permutation fuzzy entropy; rolling bearing

Citation: Yuan, Q.; Lv, M.; Zhou, R.; Liu, H.; Liang, C.; Cheng, L. Use of Composite Multivariate Multiscale Permutation Fuzzy Entropy to Diagnose the Faults of Rolling Bearing. *Entropy* **2023**, *25*, 1049. https://doi.org/10.3390/e25071049

Academic Editor: Yanchun Liang

Received: 8 June 2023
Revised: 5 July 2023
Accepted: 10 July 2023
Published: 12 July 2023

1. Introduction

Rolling bearings are one of the most important components of mechanical equipment, and their safety and reliability are crucial for the operation of machines and industrial production. Due to long-term use and complex environmental factors, rolling bearing failures are inevitable and can lead to shortened remaining useful life or damage and casualties of property [1–3]. As a result, demand for bearing failure diagnosis is also increasing. The timely diagnosis of defects and the choice of repairs or replacements are beneficial in theory and practice for the safety of production [4].

Currently, academics in different countries are conducting extensive research into rolling bearing problems. Among them, the extraction of the characteristics of fault parameters is a key link in the process of using machine learning methods for fault diagnosis research. Rolling bearing failure signals have features such as nonlinearity, periodic transience, and low signal-to-noise ratio. It is difficult to apply common methods such as time domain analysis, frequency domain analysis, and time-frequency domain analysis to these features. Therefore, it is proposed to utilize the statistical analysis advantage of entropy in the field of rotating machinery fault diagnosis. Specifically, by calculating the entropy value of rolling bearing fault data, a more comprehensive and accurate assessment of the complexity of the fault data can be achieved. This approach effectively reflects the fault characteristics of the bearings and improves the efficiency of rotating machinery fault classification. Thus, it serves as a highly effective method for extracting features from rolling bearing fault data [5,6].

Zhuang et al. [7], Minhas et al. [8], and Mantas et al. [9] have been successfully applied to mechanical error diagnosis entropy algorithms such as sample entropy, fuzzy entropy, and permutation entropy in a series of ways [10]. However, the above-mentioned research only analyzed fault signals on a single scale, ignoring information implied by different scale factors. Rohit et al. [11] combined multiscale permutation entropy and adaptive neuro-fuzzy classifier to extract MPE features and demonstrated the potential of the proposed method in diagnosing early bearing faults. Multiscale overcomes the defect that entropy calculations under a single scale condition are not sufficient to analyze time series comprehensively. However, there are some limitations in processing only a single index of entropy at multiple scales, ignoring information contained in other coarse-grained sequences under the same scale factor. Traditional multiscale algorithms are limited by data length. As the scale increases, entropy errors increase gradually, and the probability of unknown entropy increases. Mohamed et al. [12] combined the fuzzy entropy of empirical modal decomposition, principal component analysis, and self-organizing graph neural network for the fault diagnosis of bearings, and the results not only correctly assessed the degradation of rolling bearings but also identified highly sensitive defects for different types of bearing failures. Jin et al. [13] proposed an improved segmented multivariable multiscale fuzzy entropy as a flawed feature with a rate of up to 99% flaw detection. Coarse-grained time selects composite coarse-grained time to solve the problem of losing other coarse-grained sequences under the same scale factor. Furthermore, in complex composite coarse grains at different scales, only coarse grains based on mean cause a loss of useful information, resulting in a calculated entropy value that cannot fully and accurately characterize the complexity of the time series [14]. As a result, the poly coarse-grained form was proposed to solve the above problems [15–17].

This paper presents a novel method for calculating permutation fuzzy entropy by combining the high calculation accuracy of fuzzy entropy with the strong noise resistance of permutation entropy. The proposed method incorporates the concept of sorting symbolization into the calculation process of fuzzy entropy. The permutation fuzzy entropy calculation method is enhanced through composite multiscale processing to address the issue of unstable entropy calculation at large scale factors. To ensure the comprehensive and accurate extraction of information from fault signal samples, the composite multiscale permutation fuzzy entropy method is utilized. The article introduces the concept of a multivariate coarse-grained method as a solution to the problem of loss of fault feature information during entropy calculation. The three coarse-grained forms based on mean, root, mean square, and variance are presented, and a Composite Multivariate Multiscale Permutation Fuzzy Entropy (CMvMPFE) method is established. The article then applies this model to calculate the entropy value of the extraction of fault features for four bearing states' data on the public rolling bearing test bench of Case Western Reserve University in the United States, and the results of the experimental data analysis indicate that the CMvMPFE model proposed in this article is capable of efficiently extracting four bearing state feature parameters and accurately distinguishing four states. The calculations demonstrate that this model has a high level of accuracy in entropy calculation and is also highly resistant to noise. Furthermore, the model takes into account the stability, continuity, and completeness of the entropy calculation. Overall, the research demonstrates the effectiveness of CMvMPFE in characterizing bearing states, making it a valuable tool for studying rolling bearing fault feature extraction and identification. The study covers the following main topics:

(1) In this paper, a new fault feature extraction entropy calculation method called CMvMPFE is proposed for signals with non-linear characteristics, periodic impact, and low signal-to-noise ratio. The method is based on fuzzy entropy and permutation entropy and introduces the concepts of composite multiscale and multivariate coarse-grained.

(2) The MPFE, CMPFE, MvMPFE, and the proposed CMvMPFE methods are used to calculate entropy values and extract features from four states of bearing test data. The extracted feature parameters are divided into test data and training data. The SVM

model is trained using the training data, and the trained SVM is used to classify and identify the bearing states of the experimental data. The accuracy of fault identification is compared and analyzed.

(3) This study utilizes the CMvMPFE proposed in this article to perform fault feature extraction entropy calculation and analysis of different distribution locations of the same fault type and different fault depths of the same fault location.

The rest of this paper is organized as follows. Section 2 introduces the composite multivariate multiscale permutation fuzzy entropy method and basic fuzzy entropy and permutation entropy methods and introduces the CMvMPFE-SVM fault diagnosis method. Section 3 introduces the relevant feature extraction results and the CMvMPFE-SVM diagnosis results based on the CWRU dataset. Section 4 introduces the robustness analysis results of the CMvMPFE (rms). Finally, the conclusion is drawn in Section 5.

2. Composite Multivariate Multiscale Permutation Fuzzy Entropy

2.1. Fuzzy Entropy and Permutation Entropy

2.1.1. Fuzzy Entropy

Fuzzy entropy (FE) is an improvement upon sample entropy, where the unit step function in sample entropy is replaced by the exponential function $e^{-(d/r)n}$, known as the fuzzy function. This replacement ensures the continuity of entropy values and maximizes the self-similarity values of the vectors. It is used to measure the similarity between two vectors. The definition is as follows:

(1) For a time series $u(i) = \{u(1), u(2), \ldots, u(N)\}$ of length N, the first step is to determine the vector dimension m. Next, we reconstruct the time series u to obtain a new sequence:

$$X_i^m = u(i), u(i+1), \ldots, u(i+m-1) - u_0(i),\ i = 1, 2, \ldots, N-m+1. \tag{1}$$

The continuous sequence of m elements starting from the *i*-th point, where m represents the embedding dimension, is subtracted by the mean value $u_0(i)$, where

$$u_0(i) = \frac{1}{m} \sum_{j=0}^{m-1} u(i+j). \tag{2}$$

(2) The maximum distance d_{ij}^m between two reconstruction vectors X_i^m and X_j^m is defined as $d_{ij}^m = d\left[X_i^m, X_j^m\right]$.

$$\begin{gathered} d_{ij}^m = d\left[\boldsymbol{X}_i^m, \boldsymbol{X}_j^m\right] = \underset{k\in(0,m-1)}{max} |(u(i+k) - u_0(i)) - (u(j+k) - u_0(j))| \\ i, j = 1, 2, \ldots, N-m, i \neq j \end{gathered} \tag{3}$$

(3) Fuzzy membership function is defined as:

$$A_{ij}^m = e^{\left[-(\frac{d_{ij}^m}{r})^n\right]}. \tag{4}$$

The fuzzy membership function is an exponential function, with n representing its boundary gradient and r representing the similarity tolerance, typically taken within the range of 0.1 to 0.25 times the standard deviation (SD) of the original data.

(4) We define the function as follows:

$$C_i^m n, r = \frac{\sum_{j=1, j\neq i}^{N-m+1} A_{ij}^m}{N-m}; \tag{5}$$

Therefore, we obtain

$$\phi^m(n,r,N) = \frac{\sum_{i=1}^{N-m+1} C_i^m(n,r)}{N-m+1}. \tag{6}$$

(5) Similarly, after increasing the vector dimension to m + 1, we can repeat the above steps to obtain $\phi^{m+1}(n,r,N)$:

$$\phi^{m+1}(n,r,N) = \frac{\sum_{i=1}^{N-m+1} C_i^{m+1}(n,r)}{N-m+1}. \tag{7}$$

(6) Fuzzy entropy is defined as follows:

$$FE(m,n,r,N) = ln\phi^m(n,r,N) - ln\phi^{m+1}(n,r,N) \tag{8}$$

2.1.2. Permutation Entropy

Permutation entropy (PE), similar to FE, was initially designed to quantify the complexity of a time series. However, PE does not consider the specific numerical values of the time series data. Instead, it introduces the notion of permutations based on comparisons of adjacent data points [18]. Its definition is as follows:

(1) For a time series of length N: $X(i),\ i = 1,2,\ldots,N$, perform phase space reconstruction to obtain the following time series matrix:

$$\begin{bmatrix} x(1) & x(1+t) & \ldots & x(1+(m-1)t) \\ x(2) & x(2+t) & \ldots & x(2+(m-1)t) \\ \ldots & \ldots & \ldots & \ldots \\ x(i) & x(i+t) & \ldots & x(i+(m-1)t) \\ \ldots & \ldots & \ldots & \ldots \\ x(K) & x(K+t) & \ldots & x(K+(m-1)t) \end{bmatrix}, i = 1,2,\ldots,K, \tag{9}$$

where t is the time delay, usually taken as 1; m is the permutation dimension, generally taken in the range of 3~7. K = N − (m − 1)t; and each row represents a reconstructed component, with a total of K reconstructed components.

(2) Sort the m data of each reconstructed component in ascending order; that is,

$$x(i+(j_1-1)t) \leqslant x(i+(j_2-1)t) \leqslant \cdots \leqslant x(i+(j_m-1)t). \tag{10}$$

If $x(i+(j_{i1}-1)t) = x(i+(j_{i2}-1)t)$, then sort according to the size of the value i. Thus, each reconstructed vector $X(i)$ is mapped to a new symbolic sequence $U(i)$.

(3) Calculate the probability P_i of each symbol sequence occurring, where $\sum P = 1$, and $I = 1,2,\ldots,\ k,\ k \leq m!$. There are m! different ways to arrange m different symbols.

(4) By analogy with the definition of information entropy, calculate permutation entropy as follows:

$$H_p(m) = -\sum_{i=1}^{k} P_i ln P_i, \tag{11}$$

when $P_i = \frac{1}{m!}$, each symbol has an equal probability, and at this time, the permutation entropy $H(m)$ is at its maximum, which is $ln(m!)$.

(5) Normalization. For convenience, the entropy value range is adjusted to 0 to 1 and normalized:

$$H_p = \frac{H(m)}{ln(m!)}\ . \tag{12}$$

The variation of the permutation entropy (H_p) reflects and amplifies the local subtle changes in the time series. A higher value of H_p indicates a higher level of randomness or complexity in the time series. Conversely, a lower value of H_p suggests a more regular or predictable behavior in the time series.

2.2. Permutation Fuzzy Entropy

In this study, we proposed a Permutation Fuzzy Entropy (PFE) algorithm that combines the accuracy of fuzzy entropy with the simplicity and noise resistance of permutation entropy. The algorithm achieves this by introducing the symbolization idea of permutation entropy into fuzzy entropy. The first step of the algorithm involves performing a sorting symbolization on the original time series to generate a new sequence. To obtain the permutation fuzzy entropy of the original sequence, first, create a new sequence that reflects the immediate trend of the original time series through values between 1 and m!. This ensures that the trend of the new sequence is consistent with the trend or complexity of the original time series. Next, calculate its fuzzy entropy using the following specific calculation steps:

(1) According to (9), reconstruct the phase space of a time series of length N: $X(i)$, $i = 1, 2, \ldots, N$ to obtain a time series matrix.

(2) According to (10), sort the m data of each reconstructed component in ascending order. When $j1 < j2$, there is $x(i-(j_1-1)t) \leq x(i-(j_2-1)t)$ [19,20]. In this way, after rearranging, a total of m! symbol sequences are obtained. The m! symbol sequences, respectively, corresponded to the values between 1 and m!, so that the original time series $X(i)$ is transformed into a new sequence $U(i)$, with each element taking values between 1 and m!:

$$U(i) : 1 \leq i \leq N-(m-1)t. \tag{13}$$

(3) Calculate the fuzzy entropy of the new sequence $U(i)$ [21]; thus, the PFE of the original sequence can be obtained.

2.3. Composite Multivariate Multiscale Permutation Fuzzy Entropy

Composite Multiscale Permutation Fuzzy Entropy (CMPFE) is a method that combines coarse-grained sequences and PFE to extend single-scale PFE to multiscale MPFE, which reflects the complexity of multiscale time series. The method involves extending a single coarse-graining method to a multivariate coarse-graining method to reduce the loss of information in time series. The specific process is as follows.

For the new sequence $U(i)$ obtained after sorting and symbolizing as described above, it is subjected to coarse-graining to obtain the coarse-grained sequence $\boldsymbol{y}_j(\tau)$:

$$\boldsymbol{y}_j(\tau) = \frac{1}{\tau}\sum\nolimits_{i=(j-1)\tau+1}^{j\tau} x_i,\ 1 \leqslant j \leqslant \frac{N}{\tau}, \tag{14}$$

where τ is the scale factor. When $\tau = 1$, the coarse-grained sequence is the original sequence. N/τ represents the length of the sequence after coarse-graining at different time scales ($\tau > 1$). The coarse-graining process when $\tau = 2$ is shown in Figure 1.

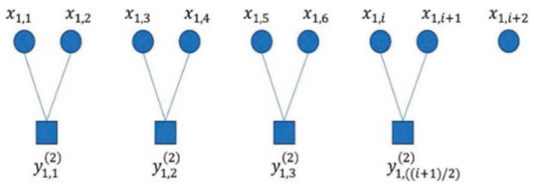


Figure 1. The coarse-graining process of MPFE when $\tau = 2$.

As the scale factor increases, the length of the coarse-grained sequence decreases. This highlights the need for a sufficiently long time series to ensure accurate results in multiscale analysis. However, using MPFE may result in a loss of information from other coarse-grained forms at the same scale, leading to significant deviations in calculation results. Therefore, it is important to enhance the coarse-graining process of MPFE. When the scale factor is τ, the composite coarse-grained sequence $y_{k,j}^{(\tau)}$ is obtained:

$$y_{k,j}^{(\tau)} = \frac{1}{\tau} \sum\nolimits_{i=(j-1)\tau+k}^{j\tau+k-1} x_i,\ 1 \leqslant j \leqslant \frac{N}{\tau}, 1 \leqslant k \leqslant \tau, \tag{15}$$

where k is the sliding sequence number at a certain time scale τ. For a single scale factor τ, only one coarse-grained sequence is generated, while the composite generates τ coarse-grained sequences by smoothing the moving order. Figure 2 illustrates the coarse-graining process when τ = 2.

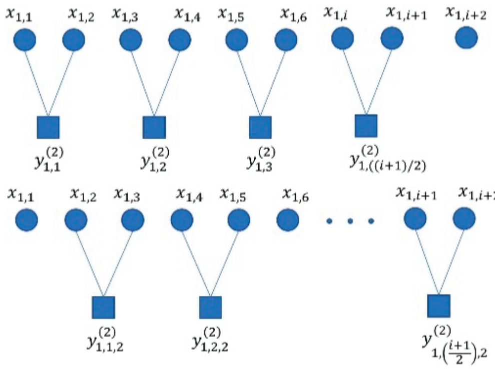


Figure 2. The coarse-graining process of CMvMPFE whe τ = 2.

To calculate the CMPFE, first, calculate the fuzzy entropy of τ coarse-grained sequences separately at the same scale factor; then, average the τ multiscale permutation fuzzy entropy values of multiscale permutation.

$$CMPFE = \frac{1}{\tau} \sum_{k=1}^{\tau} PFE \tag{16}$$

The CMPFE method, based on mean coarse-graining, still has limitations in dealing with fault signals. To address this issue, a new multivariate coarse-graining method is proposed. This method utilizes three different forms of coarse-graining: mean, root mean square (RMS), and variance (VAR). The coarse-graining processes for each form are as follows.

Coarse graining based on MEAN:

$$y_{k,j}^{(\tau)} = \frac{1}{\tau} \sum\nolimits_{i=(j-1)\tau+k}^{j\tau+k-1} x_i. \tag{17}$$

Coarse-graining based on RMS:

$$y_{k,j}^{(\tau)} = \sqrt{\frac{1}{\tau} \sum\nolimits_{i=(j-1)\tau+k}^{j\tau+k-1} x_i^2}. \tag{18}$$

Coarse-graining based on VAR:

$$y_{k,j}^{(\tau)} = \frac{1}{\tau} \sum\nolimits_{l=(j-1)\tau+k}^{j\tau+k-1} (x_i - \overline{x})^2, \overline{x} = \frac{1}{\tau} \sum\nolimits_{i=(j-1)\tau+1}^{j\tau} x_i. \tag{19}$$

To ensure comprehensive analysis, this study analyzed the Multivariate Multiscale Permutation Fuzzy Entropy (MvMPFE) and compared it with the CMPFE and CMvMPFE feature entropy values. The four characteristic entropy values of MPFE, CMPFE, MvMPFE, and CMvMPFE are used to extract bearing fault characteristics and compare their effectiveness. The best representation is then selected as a fault characteristic for fault diagnosis.

2.4. Parameter Seslection

In the CMvMPFE algorithm, there are a total of 7 parameters involved: ① time delay τ; ② permutation dimension m; ③ embedding dimension M; ④ similarity tolerance r; ⑤ boundary gradient n of the fuzzy function; ⑥ scale factor τ; and ⑦ sample length N.

Based on previous studies [22,23], it is suggested to select a dimension of permutation m ranging from 3 to 7. After conducting experiments, we discovered that the optimal result was achieved when m = 7 and τ = 1. It is important to note that if r is too large, valuable information may be lost; while if it is too small, the desired effect may not be achieved. Typically, r is set to 0.1~0.25 SD (where SD represents the standard deviation of the original time series). In this paper, the similarity tolerance r = 0.15 SD is used. Increasing the embedding dimension M includes more information in the reconstruction process but also increases calculation length and amount. According to [24], M = 2 is used. The boundary gradient n is a weight in calculating the similarity between fuzzy entropy vectors. According to [25], it is set to n = 2. The scale factor can be set to a value greater than 10. To better observe the information on a larger scale, τ = 16 is taken here.

2.5. Diagnostic Process

The method for bearing fault diagnosis based on CMvMPFE (rms) and SVM involves the following steps:

(1) Signal acquisition, wherein bearing vibration signals of four states are collected, and three sets of data for each state are selected, with each set containing 2048 data points;

(2) Features extraction, wherein parameters are set, and MPFE, CMPFE, MvMPFE, and CMvMPFE are calculated for each set of data to construct a feature vector set;

(3) Fault diagnosis, wherein the obtained feature vector set is randomly divided into training and test sets. The training samples are used to train the SVM classifier model, and the test set is used to diagnose faults using the trained SVM model, resulting in the fault diagnosis outcome.

Figure 3 shows the fault diagnosis process.

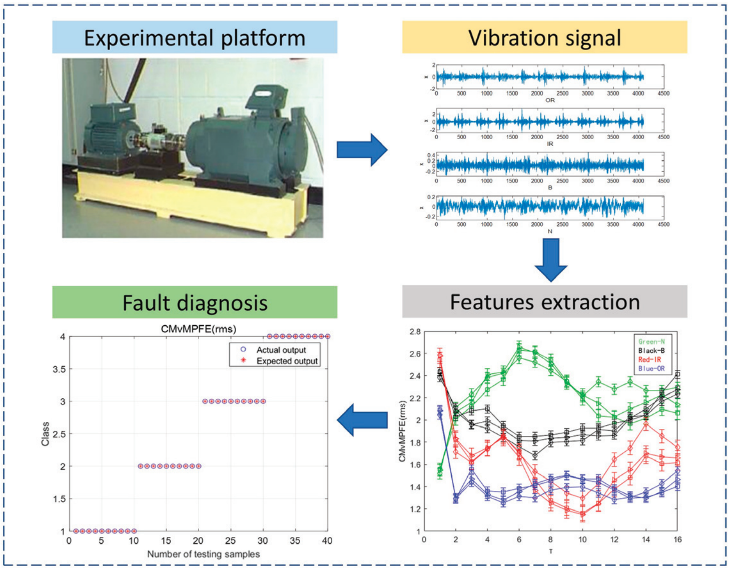


Figure 3. Fault diagnosis process.

2.6. CMvMPFE Procedure

```
CMvMPFE-SVM Procedure
load CWRU dataset
Data = data fragment with length 2048
tau = scale factor
for tau = 1 to 16
for i = 1 to tau
Select the ith to end data from data
Roughening of the selected data
Symbolize the coarse-grained data
Calculates the fuzzy entropy of the symbolized data
end for
Average the entropy values of tau
end for
SVM is used for fault diagnosis and the diagnostic accuracy of the extracted en tropy value
features is calculated
Preprocess the data labels and divides the test and training sets
xlsread (feature vector set)
Train = training sample data
Test = Test sample dataBuild a support vector machine
Obtain the highest diagnostic rate of CMvMPFE and output the diagnostic results
```

3. Verification of Rolling Bearing Fault Feature Extraction Method Based on CMvMPFE

The proposed method, CMvMPFE, improves the accuracy of signal feature extraction compared to MPFE and CMPFE. The resulting entropy feature is more stable. The rolling bearing fault diagnosis method using CMvMPFE and SVM involves the following steps:

Step 1: We selected vibration signal data for four states (inner ring failure (IR), outer ring failure (OR), rolling element failure (B), and normal (N)) based on the characteristics of the rolling bearing test bench data from Case Western Reserve University in the United States.

Step 2: To analyze the influence of different entropy calculation methods on the extraction of fault signal feature samples, the entropy values of vibration signal samples from four rolling bearing states were calculated using four methods: MPFE, CMPFE, MvMPFE, and CMvMPFE.

Step 3: The resulting entropy values were then plotted on characteristic curves, with the scale factor as the abscissa. Through this analysis, we compared and analyzed the effectiveness of the four entropy calculation methods.

3.1. Data Collection

To validate the proposed method, we utilized it for analyzing experimental data obtained from the Bearing Data Centre of Case Western Reserve University in the United States. The experimental test data utilized in this study pertained to rolling bearings. The test system is illustrated in Figure 4, with the 6205-2RSJEM SKF deep groove ball bearing being the model used at the drive end. The rolling bearing was subjected to electric spark machining to create fault points with a diameter of 0.5334 mm (21 mils). Vibration signals were collected at the drive end in four states: normal (N), inner ring (IR), outer ring (OR), and rolling element (B), at a speed of 1797 r/min and a sampling frequency of 12 KHz. A total of 80 sets of data were collected, with 20 sets for each state. It is important to note that the sample length cannot be too short as it can result in large errors in the multiscale process. To ensure accuracy in our analysis, we set the signal length of each state to 2048 data points and collected 20 sets of data for each state, resulting in a total of 80 sets. The resulting time-domain waveforms for normal (N), rolling element failure (B), inner ring failure (IR), and outer ring failure (OR) can be seen in Figure 5:

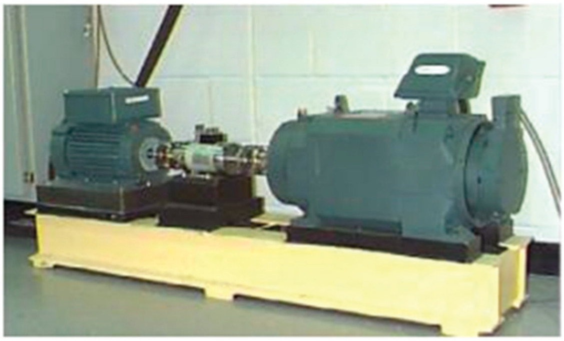


Figure 4. Rolling bearing test system.

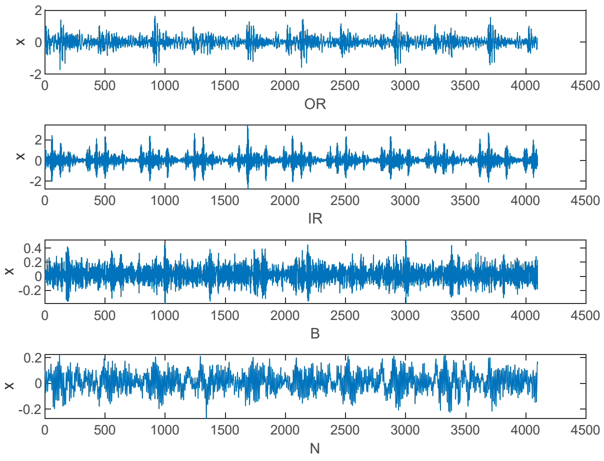


Figure 5. Vibration signals of four states.

From Figure 5, the vibration signals of the three fault states exhibit obvious periodic shocks compared to the normal state. They have a certain degree of regularity, indicating that the fault signals are more self-similar. In addition, there is a distinguishable pattern of effects between different fault states. Therefore, entropy can be used to determine the condition of the bearing. During the process of acquiring signals, noise interference is a common problem, and the complexity of the signal is often high. Analyzing fault-related information at different time scales can be difficult, as there may not be clear differences between the four states on the time-domain waveform. To address this issue, three sets were randomly selected from each of the four states, and twelve sets of data were analyzed under different parameter conditions. The outer ring failure was selected in the 3 o'clock direction.

3.2. *Signal Characteristic Analysis*

Rolling bearings inevitably produce vibrations in both normal and faulty states. The fault signals of bearings manifest as periodic impact signals, where the occurrence of faults is reflected in changes in the frequency and amplitude of the original signal. The spectrograms of the four state signals are shown in the Figure 6. By analyzing the impulsiveness and cyclostationarity, we can gain insights into the presence of faults, anomalies, or disturbances in a system, aiding in diagnosis, prognosis, and decision-making for maintenance or corrective actions. Hence, based on the impulsiveness and gyroscopic stability of bearing vibration signals, Qing Ni et al. [2] proposed a new bearing prognosis scheme. These characteristics of bearing signals align well with the advantages of entropy calculation in statistical analysis. In addition, impulsiveness is also a valid bearing failure feature [26].

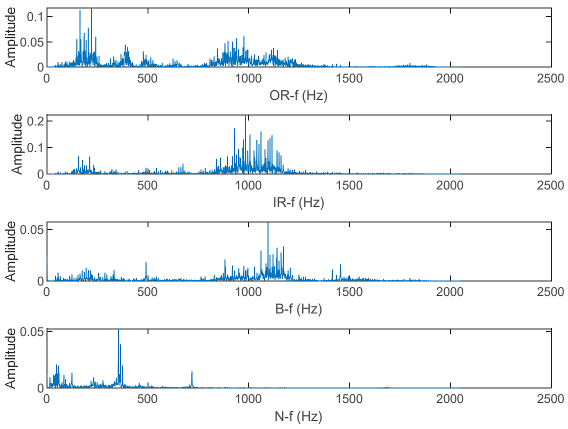


Figure 6. Spectrogram of vibration signals of four states.

Impulsiveness

Impulsiveness refers to the characteristic of a signal or system that exhibits abrupt and sudden changes or impulses. It is associated with the occurrence of instantaneous and high-amplitude events within the signal. Impulsiveness can manifest as sharp spikes or spikes with short durations in the time-domain representation of the signal. It is often characterized by high peak amplitudes and rapid changes in signal values. Understanding the impulsiveness of signals provides valuable information about the dynamic behavior and integrity of systems, helping to ensure their reliability, safety, and performance.

Cyclostationarity

Cyclostationarity refers to the property of a signal where statistical properties exhibit periodic variations over time. The vibration signals of rolling bearings have periodic changes over time, and this change itself is also periodic, resulting in the generation of cyclic stability. In cyclostationary analysis, the focus is on studying the cyclostationary features of a signal, such as cyclic autocorrelation (see Figure 7) and cyclic power spectral density (see Figure 8).

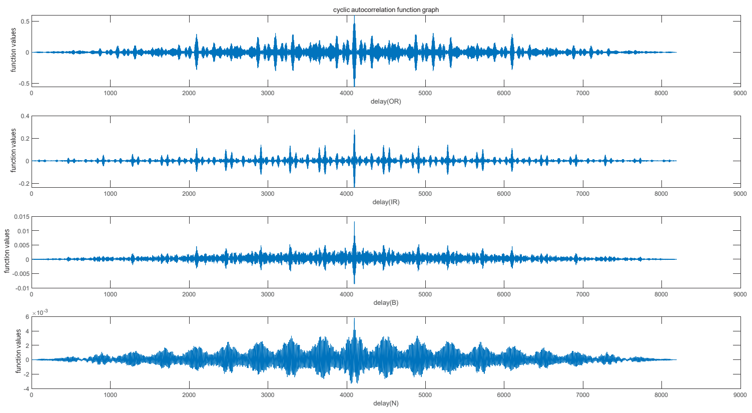


Figure 7. The cyclic autocorrelation function graph of four state signals.

From Figures 7 and 8, the periodicity of the bearing vibration signal during the cyclically stable process and the power under the main frequency components corresponding to each state can be reflected.

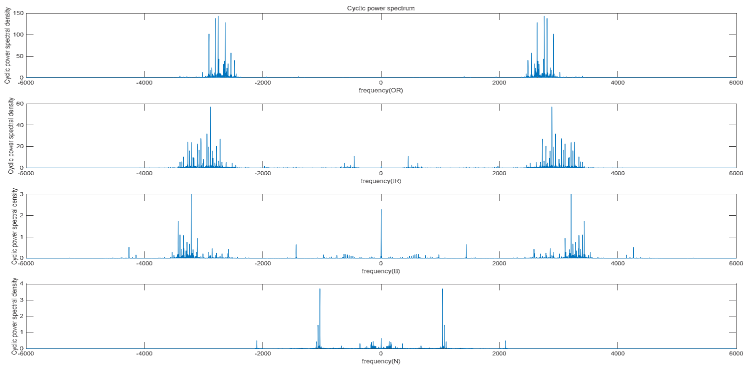


Figure 8. The cyclic power spectrum of four state signals.

3.3. Feature Extraction

For the selected 12 sets of data, we calculated their MPFE, CMPFE, MvMPFE and CMvMPFE, respectively. The entropy curves obtained are shown in Figure 9.

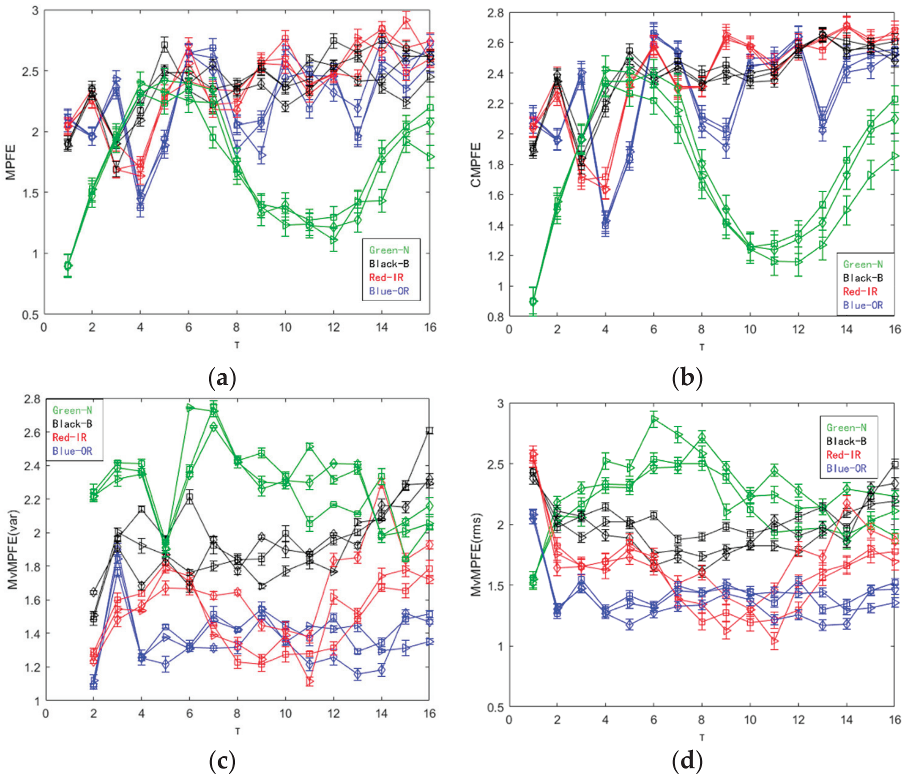


Figure 9. *Cont.*

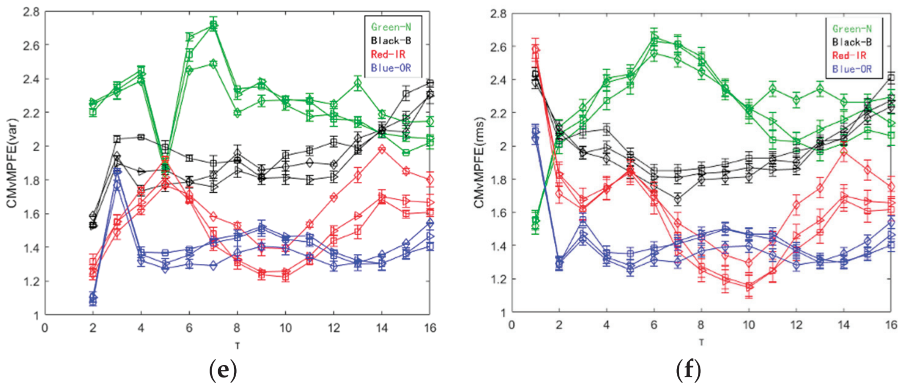


Figure 9. MPFE, CMPFE, MvMPFE (var), MvMPFE (rms), CMvMPFE (var), and CMvMPFE (rms) values of sample data: (**a**) MPFE; (**b**) CMPFE; (**c**) MvMPFE (var); (**d**) MvMPFE (rms); (**e**) CMvMPFE (var); (**f**) CMvMPFE (rms).

The results from Figure 9a indicate that the MPFE curve is disorderly and can only differentiate between normal and faulty states. On the other hand, the entropy value curves for the three fault states are mixed and difficult to distinguish. Figure 9b shows that the CMPFE curve does not show significant improvement compared to the MPFE curve. However, the entropy value curve remains chaotic, making it challenging to effectively identify the bearing fault state. Consequently, the feature extraction for identifying faults is not effective. In comparison to the MPFE and CMPFE curves, the MvMPFE curve depicted in Figure 9c exhibits an improved ability to differentiate between different state signal samples. However, it is limited in its capacity to distinguish between four states, indicating the need for increased granularity and diversified analysis. The MvMPFE curve is able to distinguish between upper and lower positions to some extent and provides a general trend. However, the entropy value distribution of rolling bearing vibration signals in the same state remains relatively dispersed, which limits its effectiveness in extracting rolling bearing fault characteristics.

The comparison shows that the two CMvMPFE curves maintain the correct position distribution and trend of entropy values for different bearing state signals. However, the entropy values for the same state with the scale factor are more concentrated, indicating that CMvMPFE's ability to represent faults is enhanced after compounding. It effectively distinguishes different bearing states and can be used as a rolling bearing fault feature. The CMvMPFE (var) curve lacks smoothness and information completeness when the scale factor is τ = 1. On the other hand, the CMvMPFE (rms) curve is optimal in terms of both curve smoothness and scale information completeness.

3.4. Fault Diagnosis

Based on the analysis, it is evident that CMvMPFE (rms) is an effective method for accurately extracting fault features of bearings. Therefore, in this study, the CMvMPFE (rms) of the signal sample was utilized as the input for the fault classifier for identification. A support vector machine was chosen as the classifier. The study collected 20 sets of vibration data for each type, with 10 sets randomly selected for training and the remaining 10 sets for testing. The training samples were used to train the classifier to obtain a classification model with optimal parameters. The trained classifier was then used to identify faults in the test samples. The diagnostic results of MPFE, CMPFE, MvMPFE (rms), and CMvMPFE (rms) methods are shown in Figure 10.

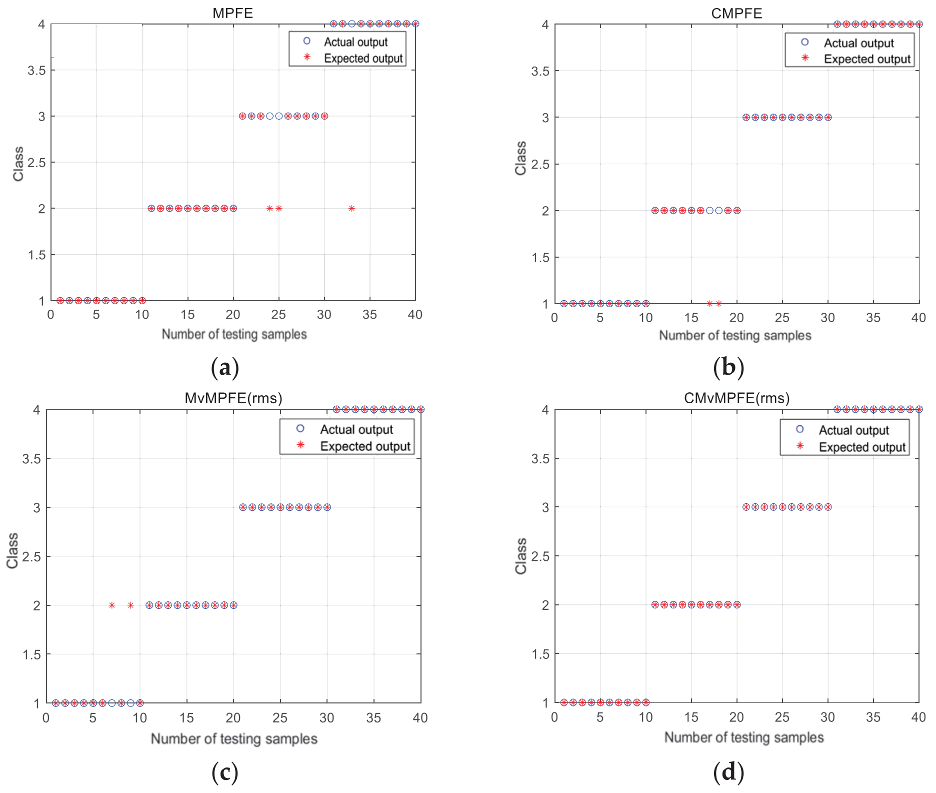


Figure 10. MPFE, CMPFE, MvMPFE (rms), and CMvMPFE (rms) identification results: (**a**) MPFE; (**b**) CMPFE; (**c**) MvMPFE (rms); (**d**) CMvMPFE (rms).

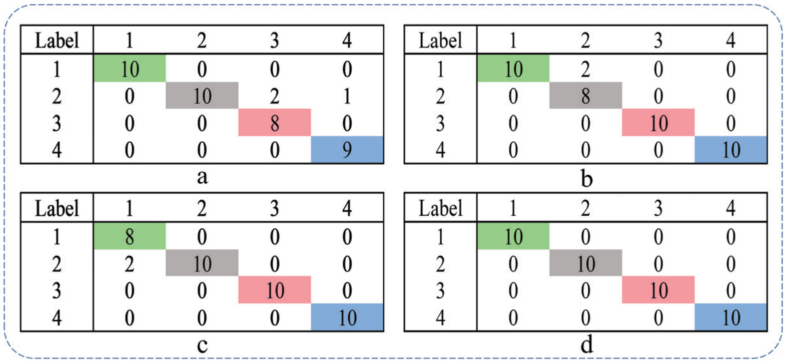
The four states are labeled as follows: 1-N, 2-B, 3-IR, and 4-OR. From Figure 10, it can be observed that the MPFE model has three groups of misclassifications: two groups of inner race faults are misclassified as rolling element faults, and one group of outer race fault is misclassified as a rolling element fault. The CMPFE model exhibits two groups of misclassifications, where rolling element faults are mistakenly classified as normal. Similarly, the MvMPFE (rms) model also has two groups of misclassifications, where two groups of normal states are classified as rolling element faults. On the other hand, the CMvMPFE model achieves perfect classification results for all the data, indicating that CMvMPFE (rms) possesses excellent fault representation capability.

Based on Figure 10, it can be inferred that the recognition accuracies of the four models are 92.5%, 95%, 95%, and 100%, respectively. Similarly, CMvMPFE (rms) achieves the highest accuracy among the four models, as shown in Table 1.

Table 1. Identification accuracy.

Feature Models	Accuracy Rate
MPFE	92.5%
CMPFE	95.0%
MvMPFE (rms)	95.0%
CMvMPFE (rms)	**100%**

Furthermore, considering precision, recall, and F1-Score as evaluation metrics, the confusion matrix for each of the four models, based on their recognition results, is presented in Figure 11.



Label	1	2	3	4
1	10	0	0	0
2	0	10	2	1
3	0	0	8	0
4	0	0	0	9

a

Label	1	2	3	4
1	10	2	0	0
2	0	8	0	0
3	0	0	10	0
4	0	0	0	10

b

Label	1	2	3	4
1	8	0	0	0
2	2	10	0	0
3	0	0	10	0
4	0	0	0	10

c

Label	1	2	3	4
1	10	0	0	0
2	0	10	0	0
3	0	0	10	0
4	0	0	0	10

d

Figure 11. MPFE, CMPFE, MvMPFE (rms), and CMvMPFE (rms) confusion matrix: (**a**) MPFE; (**b**) CMPFE; (**c**) MvMPFE (rms); (**d**) CMvMPFE (rms). (Green indicates the number of normal classified as normal, gray indicates the number of rolling element faults classified as rolling element faults, red indicates the number of inner ring faults classified as inner ring faults, and blue indicates the number of outer ring faults classified as outer ring faults).

Since each sample has equal weight, we choose the micro-average metric. From the confusion matrix, we can obtain the precision (P), recall (R), and F1-score (F1) of the four models, as shown in Table 2.

Table 2. P, R, and F1 values for the four models.

Model	**P**	**R**	**F1**
MPFE	0.925	0.925	0.925
CMPFE	0.95	0.95	0.95
MvMPFE (rms)	0.95	0.95	0.95
CMvMPFE (rms)	1	1	1

When the confusion matrix is a square matrix, we have P = R = F1. From Table 2, we can conclude that CMvMPFE (rms) achieves the highest score, indicating the best performance or superior fault representation capability.

3.5. Comparing Computational Costs

Compared with existing methods, the CMvMPFE method proposed in this paper has the following advantages in terms of computational cost:

(1) Computational resource requirements: the method in this paper is simple and effective, with low hardware cost requirements, and the calculation software only MATLAB 2022b required.
(2) Model size cost: The CMvMPFE method has a small model and fast calculation speed. The calculation time of 12 groups of feature curves is shown in Figure 12. From Figure 12, it can be seen that the time spent on each calculation is basically consistent, the program is stable, and the calculation is reliable.
(3) Algorithm complexity and tuning cost: The algorithm of the method proposed in this paper is simple, and mainly based on the existing fuzzy entropy and permutation entropy, as well as the concept of composite coarse-graining. The subsequent tuning cost is small, and the scale factor, permutation dimension, etc., can be optimized and adjusted. The syntax logic of the algorithm can also be optimized and improved to further speed up the calculation.

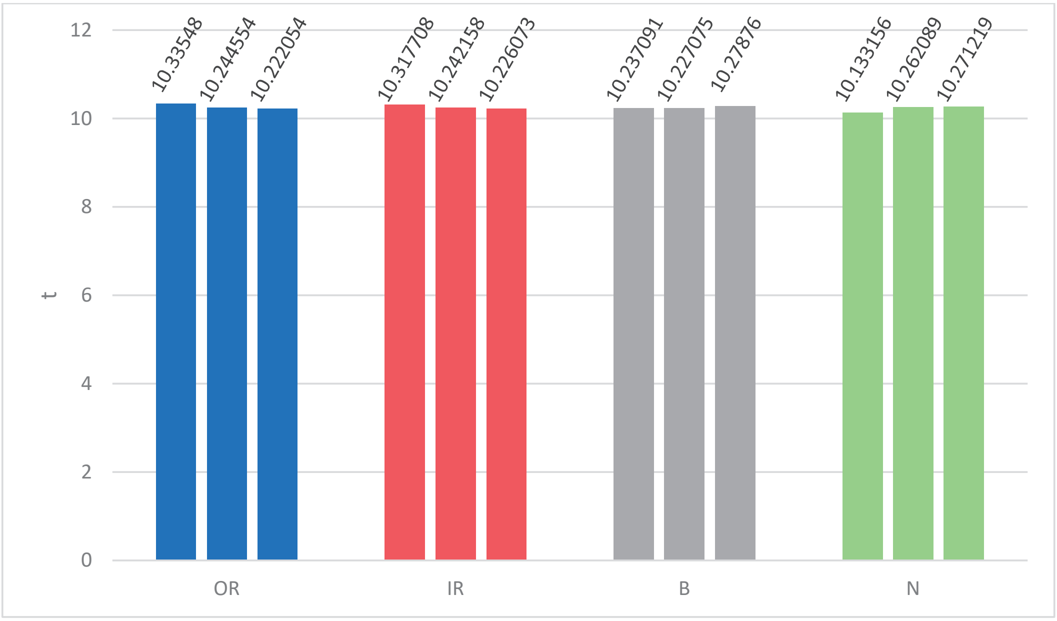


Figure 12. The calculation time of each feature curve for four states.

4. Robustness Analysis of CMvMPFE (rms)

In the fault diagnosis process, it is essential that the fault features are not only capable of effectively representing faults but also possess universality and robustness. This enables the fault extraction feature method to be applied across various scenarios and working conditions, making it an excellent method for fault diagnosis.

4.1. Feature Signal Extraction of the Same Fault Type at Different Distribution Positions

For verifying the same outer ring fault, we selected feature signals from different distribution positions and performed composite multi-element multi-scale permutation fuzzy entropy calculation and analysis on the three positions of 3 o'clock, 6 o'clock, and 12 o'clock. The analysis revealed that the composite multiscale permutation fuzzy entropy based on root mean square provided the best results. Therefore, we selected CMvMPFE (rms) as the feature signal extraction method, and the results are presented in Figure 13.

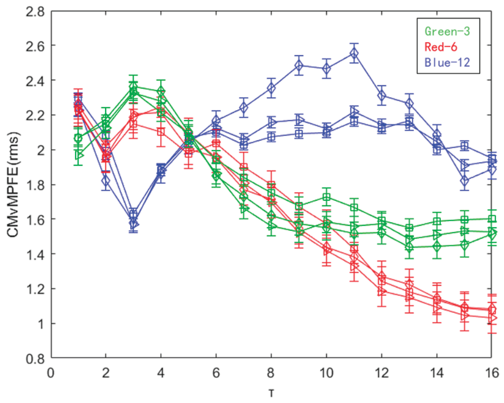


Figure 13. CMvMPFE (rms) of outer ring faults at different distribution positions.

Based on the findings presented in Figure 13, the entropy value distribution of the outer ring faults at the 3 o'clock and 6 o'clock positions show a similar trend, where the values increase first and then decrease with the scale factor. However, the entropy value distribution of the outer ring fault at the 12 o'clock position follows a different pattern, where the values decrease first and then increase and decrease with the scale factor. These results demonstrate that CMvMPFE (rms) is not only capable of distinguishing different state rolling bearing faults, but also effectively identifies faults at different distribution positions, i.e., 3 o'clock, 6 o'clock, and 12 o'clock set on the outer ring, indicating the method's robustness.

4.2. Extraction of Characteristic Signals at the Same Fault Location at Different Fault Depths

Effective fault characteristics should not only be capable of identifying the presence of a fault but also distinguishing between different fault states and their distribution at various locations. Furthermore, it should be able to differentiate between fault depths at the same location. In this study, vibration signal data of the inner ring of a rolling bearing at a speed of 1797 r/min were analyzed under five different working conditions: normal state, and fault depths of 7 mils, 14 mils, 21 mils, and 28 mils. The entropy value of the characteristic parameters of the five different fault depths were calculated using CMvMPFE (rms), and the results are presented in Figure 14.

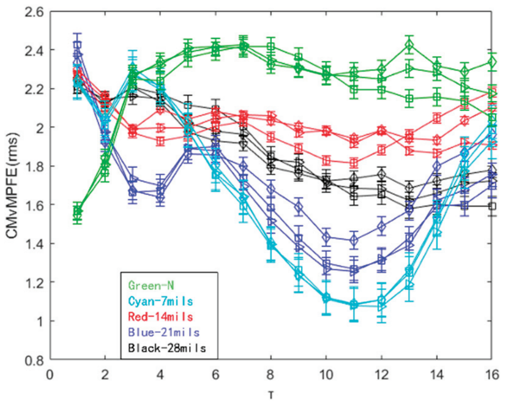


Figure 14. CMvMPFE (rms) of different fault depths of the inner ring at the same speed.

As illustrated in Figure 14, the distribution of entropy values in the normal state initially increases with the scale factor and then remains stable. On the other hand, the entropy value trend for inner ring fault characteristics at various depths decreases as the scale factor increases. The CMvMPFE (rms) method can effectively differentiate between normal and fault states and even distinguish between different fault depths, resulting in a satisfactory outcome.

5. Conclusions

(1) This paper introduces a new entropy algorithm called CMvMPFE (rms) for characterizing the state of vibration signals. The proposed algorithm aims to address the limitations of the incomplete extraction of fault characteristic information under a single scale and the low calculation accuracy and poor anti-interference of multiscale fuzzy and multiscale permutation entropy. The experimental results demonstrate that CMvMPFE (rms) can accurately and completely extract the fault characteristic information of vibration signals. Additionally, the obtained entropy values exhibit better consistency, accuracy, and stability.

(2) This paper proposes a new method for diagnosing faults in rolling bearings. The method is based on CMvMPFE (rms) and SVM, and experimental data are used for

calculation and comparative analysis. The results demonstrate that the proposed method outperforms existing methods in terms of fault characteristic extraction and pattern recognition accuracy.

(3) To verify the robustness and anti-interference ability of the entropy calculation method proposed by CMvMPFE (rms), in this paper, three situations were analyzed and calculated. These included the extraction of characteristic signals at different distribution locations for the same fault type, the extraction of characteristic signals at different fault depths at the same fault location, and the extraction of fault characteristics at different speeds at the same fault depth. Results indicate that the new method proposed in this paper exhibits good robustness and strong anti-interference ability.

However, only three coarse-grained forms is still not comprehensive enough, and the coarse-grained forms need to be expanded.

Author Contributions: Resources, conceptualization, R.Z. and Q.Y.; original draft preparation, Q.Y.; methodology, M.L.; data processing, validation, software, M.L., H.L., C.L. and L.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Projects: National Natural Science Foundation of China Key Project (NO. 51839005), High-tech Ship Research Special Project (NO. K24532-1-2), and Zhoushan Science and Technology Bureau Fund Project (NO. 2023C41022).

Institutional Review Board Statement: Studies not involving humans or animals.

Data Availability Statement: The data in this paper come from a publicly available dataset provided by the Bearing Data Center of Case Western Reserve University in the United States. Data links here: https://engineering.case.edu/bearingdatacenter (accessed on 20 September 2022).

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

References

1. Lei, Y.; Qiao, Z.; Xu, X.; Lin, J.; Niu, S. An underdamped stochastic resonance method with stable-state matching for incipient fault diagnosis of rolling element bearings. *Mech. Syst. Signal Process.* **2017**, *94*, 148–164. [CrossRef]
2. Ni, Q.; Ji, J.; Feng, K. Data-Driven Prognostic Scheme for Bearings Based on a Novel Health Indicator and Gated Recurrent Unit Network. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1301–1311. [CrossRef]
3. Yuan, Q.; Sun, Y.; Zhou, R.; Wen, X.; Dong, L. Prediction and analysis of bearing vibration signal with a novel gray combination model. *Adv. Mech. Eng.* **2020**, *12*, 1687814020919241. [CrossRef]
4. Zheng, J.; Chen, Y.; Tong, J.; Pan, H. Refined generalized multivariate multiscale reverse dispersion entropy of composites and its application in the diagnosis of rolling bearing failure. *China Mech. Eng.* **2023**, *34*, 1315–1325.
5. Feng, K.; Ji, J.; Ni, Q.; Li, Y.; Mao, W.; Liu, L. A novel vibration-based prognostic scheme for gear health management in surface wear progression of the intelligent manufacturing system. *Wear* **2023**, *522*, 204697. [CrossRef]
6. Rostaghi, M.; Khatibi, M.; Ashory, M.; Azami, H. Bearing Fault Diagnosis Using Refined Composite Generalized Multiscale Dispersion Entropy-Based Skewness and Variance and Multiclass FCM-ANFIS. *Entropy* **2021**, *23*, 1510. [CrossRef] [PubMed]
7. Zhuang, D.; Liu, H.; Zheng, H.; Xu, L.; Gu, Z.; Cheng, G.; Qiu, J. The IBA-ISMO Method for Rolling Bearing Fault Diagnosis Based on VMD-Sample Entropy. *Sensors* **2023**, *23*, 991. [CrossRef] [PubMed]
8. Minhas, A.S.; Sharma, N.; Singh, G.; Kankar, P.K.; Singh, S. Improvement in classification accuracy and computational speed in bearing fault diagnosis using multiscale fuzzy entropy. *J. Braz. Soc. Mech. Sci. Eng.* **2020**, *42*, 586. [CrossRef]
9. Landauskas, M.; Cao, M.; Ragulskis, M. Permutation entropy-based 2D feature extraction for bearing fault diagnosis. *Nonlinear Dyn.* **2020**, *102*, 1717–1731. [CrossRef]
10. Yang, C.; Zhang, Q.; Guo, W.; Chen, C. Rolling Bearing Fault Diagnosis Based On Full-Mapping Composite Multiscale Dispersion Entropy. *Bearing* **2023**, 1–11.
11. Tiwari, R.; Gupta, V.; Kankar, P. Bearing fault diagnosis based on multi-scale permutation entropy and adaptive neuro fuzzy classifier. *J. Vib. Control* **2013**, *21*, 461–467. [CrossRef]
12. Zair, M.; Rahmoune, C.; Benazzouz, D. Multi-fault diagnosis of rolling bearing using fuzzy entropy of empirical mode decomposition, principal component analysis, and SOM neural network. *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.* **2018**, *233*, 3317–3328. [CrossRef]
13. Jin, Z.; Xiao, Y.; He, D.; Wei, Z.; Sun, Y.; Yang, W. Fault diagnosis of bearing based on refined piecewise compo-site multivariate multiscale fuzzy entropy. *Digit. Signal Process.* **2023**, *133*, 103884. [CrossRef]
14. Yang, X.; Gong, J.; An, L.; Liu, X. Gear failure diagnosis based on refined multivariate multiscale composite fuzzy entropy of the ensemble. *J. Mech. Electr. Eng.* **2023**, *40*, 335–343.

15. Qin, A.; Mao, H.; Hu, Q.; Zhuang, Q. Bearing Fault Diagnosis Method Based on Ensemble Composite Multi-Scale Dispersion Entropy and Density Peaks Clustering. *IEEE Access* **2021**, *9*, 24373–24389. [CrossRef]
16. Azami, H.; Escudero, J. Refined composite multivariate generalized multiscale fuzzy entropy: A tool for com-plexity analysis of multichannel signals. *Phys. A Stat. Mech. Appl.* **2017**, *465*, 261–276. [CrossRef]
17. Zhuang, M.; Li, G.; Fan, Z.; Kong, D. Fault diagnosis of planetary gearboxes based on HRCMFDE LS and BA-SVM. *J. Mech. Electr. Eng.* **2022**, *39*, 1535–1543.
18. Carrasco, J.; Tan, C.; Escudero, J. Permutation Entropy for Graph Signals. *IEEE Trans. Signal Inf. Process. Over Netw.* **2022**, *8*, 288–300. [CrossRef]
19. Zheng, J.; Cheng, J.; Yang, Y. Multi-scale Permutation Entropy and Its Applications to Rolling Bearing Fault Di-ag nosis. *China Mech. Eng.* **2013**, *24*, 2641–2646.
20. He, Y. Study on the Diagnosis of Vibration Faults of Hydroelectric Generation Sets Based on Stochastic Resonance and Multidimensional Permutation Entropy. Master's Thesis, Xi'an University of Technology, Xi'an, China, 2016.
21. Dong, C. Study on the Fault Feature Extraction Method of Motor Bearing Based on EEMD and Multiscale Fuzzy Entropy. Master's Thesis, Dalian Jiaotong University, Dalian, China, 2017.
22. Zheng, J.; Tu, D.; Pan, H.; Hu, X.; Liu, T.; Liu, Q. A Refined Composite Multivariate Multiscale Fuzzy Entropy and Laplacian Score-Based Fault Diagnosis Method for Rolling Bearings. *Entropy* **2017**, *19*, 585. [CrossRef]
23. Pompe, B. Permutation Entropy: A Natural Complexity Measure for Time Series. *Phys. Rev. Lett.* **2002**, *88*, 174102.
24. Zheng, J.; Chen, M.; Cheng, J.; Yang, Y. Multiscale fuzzy entropy and its application in rolling bearing fault diag nosis. *Vib. Eng.* **2014**, *27*, 145–151.
25. Chen, W. A Study of Feature Extraction from Semg Singal Based on Entropy. Ph.D. Thesis, Shanghai Jiaotong University, Shanghai, China, 2008.
26. Ding, J. A double impulsiveness measurement indices-bilaterally driven empirical wavelet transform and its application to wheelset-bearing-system compound fault detection. *Measurement* **2021**, *175*, 109135. [CrossRef]

Article

A Hybrid Recommender System Based on Autoencoder and Latent Feature Analysis

Shangzhi Guo [1], Xiaofeng Liao [1], Gang Li [1], Kaiyi Xian [1], Yuhang Li [2] and Cheng Liang [3,*]

1 College of Computer Science, Chongqing University, Chongqing 400044, China; 20211401018g@cqu.edu.cn (S.G.); xfliao@cqu.edu.cn (X.L.); 20211401019g@cqu.edu.cn (G.L.); 20211401021g@cqu.edu.cn (K.X.)
2 College of Computer and Information Science, Southwest University, Chongqing 400715, China; limouhang@email.swu.edu.cn
3 Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou 510006, China
* Correspondence: c_liang@e.gzhu.edu.cn

Abstract: A recommender system (RS) is highly efficient in extracting valuable information from a deluge of big data. The key issue of implementing an RS lies in uncovering users' latent preferences on different items. Latent Feature Analysis (LFA) and deep neural networks (DNNs) are two of the most popular and successful approaches to addressing this issue. However, both the LFA-based and the DNNs-based models have their own distinct advantages and disadvantages. Consequently, relying solely on either the LFA or DNN-based models cannot ensure optimal recommendation performance across diverse real-world application scenarios. To address this issue, this paper proposes a novel hybrid recommendation model that combines Autoencoder and LFA techniques, termed AutoLFA. The main idea of AutoLFA is two-fold: (1) It leverages an Autoencoder and an LFA model separately to construct two distinct recommendation models, each residing in a unique metric representation space with its own set of strengths; and (2) it integrates the Autoencoder and LFA model using a customized self-adaptive weighting strategy, thereby capitalizing on the merits of both approaches. To evaluate the proposed AutoLFA model, extensive experiments on five real recommendation datasets are conducted. The results demonstrate that AutoLFA achieves significantly better recommendation performance than the seven related state-of-the-art models.

Keywords: data science; deep neural network; Latent Feature Analysis; multi-metric recommender system; matrix representation

Citation: Guo, S.; Liao, X.; Li, G.; Xian, K.; Li, Y.; Liang, C. A Hybrid Recommender System Based on Autoencoder and Latent Feature Analysis. *Entropy* **2023**, *25*, 1062. https://doi.org/10.3390/e25071062

Academic Editor: Yanchun Liang

Received: 30 June 2023
Revised: 9 July 2023
Accepted: 10 July 2023
Published: 14 July 2023

1. Introduction

In the current era characterized by abundant information, individuals are confronted with a deluge of extensive data [1–4]. Notable examples include the colossal amount of data generated by Google, reaching the scale of petabytes, and Flickr, which produces terabytes of data on a daily basis [5,6]. The challenge at hand is to devise an intelligent system capable of extracting relevant information from these vast datasets [7–9]. One practical approach to tackle this challenge is the utilization of a recommender system (RS). RSs play crucial roles in enhancing online services, contributing to both business growth and improved user experiences [10]. Typically, a user-item rating matrix is employed to capture user preferences across various items such as news, short videos, music, movies, and commodities [11]. In this matrix, each row represents a specific user, each column corresponds to a specific item, and each entry signifies a user's preference for a particular item [3]. The key to implementing an RS lies in uncovering users' latent preferences for different items based on this user-item rating matrix [12,13].

Numerous approaches have been proposed for implementing an RS. Among them, the Latent Feature Analysis (LFA) model has gained significant popularity in industrial applications due to its efficiency and scalability [14]. When applied to a user-item rating matrix, the LFA model projects users and items onto a shared low-dimensional Latent Feature space [15]. By training two low-dimensional matrices using the observed entries only [16], the LFA model can estimate the missing entries by leveraging these trained matrices [17–20]. As a result, the LFA model offers advantages in terms of efficiency and scalability, particularly in industrial contexts. However, it should be noted that the LFA model is a linear model and may not effectively address complex non-linear relationships between users and items [21].

In recent times, the rapid advancement of deep learning has led to the widespread adoption of deep neural networks (DNNs) [22–24] in RSs [25,26]. DNNs have emerged as a promising approach for capturing complex non-linear relationships between users and items [27,28]. In the pursuit of implementing RSs, various DNN-based models have been proposed, with significant emphasis placed on devising sophisticated structures that can better accommodate user behavior data [29]. However, a notable difference between DNN-based models and the Latent Feature Analysis (LFA) model lies in their approaches to handling data [30–34]. While DNN-based models often operate on complete data, the observed entries within a user-item rating matrix, the reality is that RS-generated user-item rating matrices tend to exhibit low rating density [35–38]. This means that a significant portion of the matrix remains empty or contains missing ratings. Consequently, DNN-based models face challenges in effectively addressing the prevalent data sparsity issues in RSs [12,13,39,40].

Upon the aforementioned discussions, it becomes apparent that the LFA and DNN-based models have distinct advantages and disadvantages. Consequently, relying solely on either the LFA model or the DNN-based model cannot ensure optimal recommendation performance across diverse real-world application scenarios. To tackle this challenge, this study proposes a novel hybrid recommendation model called AutoLFA, which combines Autoencoder [41] and LFA techniques. The main concept behind AutoLFA is two-fold: (1) It leverages an Autoencoder and an LFA model separately to construct two distinct recommendation models, each residing in a unique metric representation space with its own set of strengths, and (2) it integrates the Autoencoder and LFA models using a customized self-adaptive weighting strategy, thereby capitalizing on the merits of both approaches. By incorporating elements from both the LFA model and DNN-based models, AutoLFA can deliver superior recommendation performance across various real-world application scenarios. This paper contributes to the field in the following ways:

1. It proposes an AutoLFA model that aggregates the merits of both the LFA model and the DNN-based model by a customized self-adaptive weighting strategy;
2. Theoretical analyses and model designs are provided for the proposed AutoLFA model;
3. Extensive experiments on five real recommendation datasets are conducted to evaluate the proposed AutoLFA model. The results demonstrate that AutoLFA achieves significantly better recommendation performance than the related state-of-the-art models.

2. Related Work

Collaborative Filtering (CF) stands as a popular and effective approach for implementing an RS [2]. Its fundamental principle involves utilizing historical user behavior data to uncover similarities between users and items, thereby predicting users' potential preferences for items. Matrix factorization serves as a prominent CF method, which typically maps the user-item rating matrix into two Latent matrices to explore the similarity between users and items [12]. Subsequently, the development of the LFA model introduced a notable distinction. Unlike matrix factorization, the LFA model exclusively trains the Latent Feature model using observed entries within the user-item rating matrix. As a result, LFA exhibits high efficiency and scalability, particularly in industrial applications [12,13]. Over time, several sophisticated LFA models have emerged, including those that consider data characteristics [42], incorporate non-negativity constraints [43], adopt generalized and fast-converging approaches [44], focus on smooth L_1-norm regularization [12], employ probabilistic methods [45], apply dual loss [13], utilize prediction sampling [46,47], prioritize confidence-driven techniques [48], incorporate posterior neighborhood regularization [49], employ ensemble approaches involving multiple spaces and norms [50], explore graph regularization [51], and embrace deep structured architectures [52]. However, it is essential to note that the LFA model is inherently shallow and linear in nature. Consequently, it faces challenges when attempting to capture the deep non-linear relationships between users and items embedded within complex user-item rating matrices [21].

In recent times, Deep Neural Networks (DNN) have gained significant traction in the development of Collaborative Filtering (CF)-based RSs due to their powerful non-linear learning capabilities derived from deep learning structures [53]. DNN-based models aim to reduce the user-item rating matrix into a low-dimensional space to capture the similarities between users and items. A comprehensive review of DNN-based RSs was conducted by Zhang et al. [29]. Various sophisticated DNN-based models have emerged, including hybrid Autoencoder-based approaches [54], Autoencoder-based methods [41], multi-task learning-oriented techniques [11], graph neural network (GNN)-based models [55], neural factorization-based approaches [56], Autoencoders combined with radial basis function-based methods [57], attentional factorization-based models [58], hybrid deep models [28], biased Autoencoder-based techniques [21], and convolutional matrix factorization approaches [59]. However, it is worth noting that DNN-based models face challenges in addressing data sparsity problems since they are trained on complete data rather than solely relying on the observed entries within a user-item rating matrix [13]. Unfortunately, user-item rating matrices generated by RSs often exhibit very low rating densities.

Notably, although many LFA-based and DNN-based models have been built to achieve commendable recommendation performance, each approach has its own set of advantages and disadvantages. In comparison, the proposed AutoLFA is a hybrid recommendation model that combines the strengths of both Autoencoder and LFA techniques. This combination is controlled by a customized self-adaptive weighting strategy, ensuring that AutoLFA leverages the merits of both the LFA and DNN-based models, ultimately leading to superior recommendation performance across various real-world application scenarios.

3. Preliminaries

Definition 1 (**user behavior data**): *Let M be a set of users, and N be a set of items. The matrix $X \in \mathbb{R}$ with $|M|$ rows and $|N|$ columns records the interactions between different users and items. Here, x_{mn} represents the specific interaction specification of user m on item n. The vector $\mathrm{x}^m = \{x_{m1}, \cdots x_{m|N|}\}$ denotes the behavioral data of user m across all items, while each item n can be represented as a vector $\mathrm{x}^n = \{x_{1n}, \cdots x_{|M|n}\}$. A binary matrix $B \in \mathbb{R}$ with $|M|$ rows and $|N|$ columns distinguish the observed and unobserved interactions of X:*

$$b_{mn} = \begin{cases} 1 & \text{if } x_{mn} \text{ observed} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where b_{mn} denotes the specific entry on B.

Definition 2 (**problem**): *In recommender systems, two primary tasks exist: rating prediction and ranking prediction. Our proposed model is more suited to rating prediction, which aims to learn a parametric model denoted as $f(\cdot)$ using observed ratings of X in order to predict the unobserved ones. The prediction process can be represented as follows:*

$$f(M, N; \theta) \to X. \tag{2}$$

Here, θ represents the parameters of f(·). The objective function of f(·) is to minimize the empirical risk, expressed as:

$$L(f) = \sum_{m \in M, n \in M} \epsilon(f(m,n;\theta), x_{mn}). \tag{3}$$

In this Equation, ϵ(·) denotes the error function that measures the distance between the predicted output $\hat{x}_{mn}$ from f(m, n; θ) and the true rating x_{mn}.

4. The Proposed AutoLFA

As mentioned above, traditional approaches, such as Latent Feature Analysis (LFA), offer efficiency and scalability but may not capture complex non-linear relationships. On the other hand, deep neural networks (DNNs) show potential in capturing non-linear relationships but face challenges in dealing with data sparsity issues. Inspired by this finding, we propose AutoLFA with the aim of addressing both the challenge of LFA's inability to capture complex non-linear relationships and the difficulty faced by DNN-based models in handling data sparsity issues. Figure 1 depicts the architecture of our proposed model, which can be separated into three steps: (1) Feed the user behavior data into the LFA-based and Autoencoder-based models separately; (2) obtain the predictions of the unobserved value from these two models; (3) aggregate the predictions of two models with a self-adaptive ensemble method to obtain final prediction $\hat{X}$. To illustrate the principle of Auto-LFA, we provide an example of predicting x_{22} in Figure 1. The predicted values from the two predictors differ by 3.5 in the LFA-based model and 2 in the Autoencoder-based model. These predictions are then weighted to derive the final prediction of 2.9. Next, we will provide a detailed description of AutoLFA.

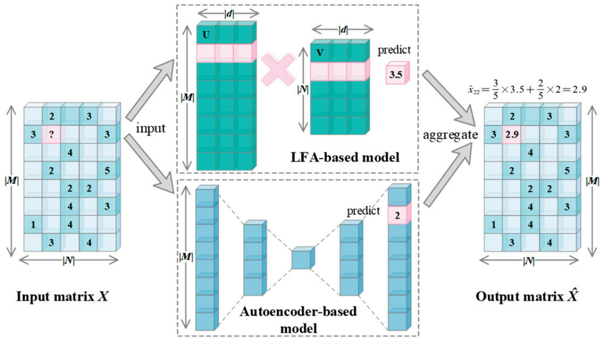


Figure 1. The architecture of the proposed AutoLFA model.

4.1. The Latent Feature Analysis-Based (LFA-Based) Model

Given a user behavior matrix X, an LFA-based predictor aims to train two Latent Feature matrices U of size $|M| \times d$ and V of size $d \times |N|$ to generate the rank-d approximation $\hat{X}$ of X is based on the known entry of X, in which d is much smaller than $min\{|M|, |N|\}$. In this context, the row vectors of U represent user characteristics, while the column vectors of V represent item characteristics in the Latent Feature space.

We utilize the inner product space with an L_2-norm $||\cdot||_{L_2}^2$ as the *Loss* function in the LFA-based model to measure the distance between X and $\hat{X}$, as demonstrated below:

$$L(U,V) = \frac{1}{2} \parallel B \odot (X - \hat{X})_{L_2}^2 = \frac{1}{2} \parallel B \odot (M - UV)_{L_2}^2 \tag{4}$$

where $\odot$ denotes the Hadamard product. According to [12,13], regularization is crucial in preventing overfitting. By incorporating Tikhonov regularization into Equation (4), we obtain:

$$L(U,V) = \frac{1}{2} \parallel B \odot (X - UV) \parallel_{L_2}^2 + \frac{\lambda_1}{2}(\|U\|_{L_2}^2 + \|V\|_{L_2}^2). \tag{5}$$

Here, λ_1 is a hyperparameter that controls the intensity of its regularization penalty. It is worth noting that since the user cannot fully access all items leading X to be sparse, it is necessary to expand Equation (5) into a density-oriented form to improve efficiency, as follows [12,13]:

$$L(U,V) = \frac{1}{2} \sum_{x_{mn} \in X_o} (x_{mn} - \sum_{k=1}^{d} u_{mk} v_{kn})^2 + \frac{\lambda}{2} \sum_{x_{mn} \in X_o} (\sum_{k=1}^{d} u_{mk}^2 + \sum_{k=1}^{d} v_{kn}^2). \tag{6}$$

Here, u_{mk} represents the entry at the u-th row and k-th column of U, and v_{kn} represents the entry at the k-th row, n-th column of V, and X_o is the observed entries of X. We train the matrices U and V with the Adam optimizer [16] to obtain better prediction results.

4.2. The Autoencoder-Based Model

We chose the representative I-AutoRec [41] as the Autoencoder-based model. Formally, when given a user behavior data matrix X, I-AutoRec aims to solve the same problem as defined in Equation (3). The objective is to minimize the following loss function:

$$L(f) = \sum_{\mathbf{x}^n \in M} \|(\mathbf{x}^n - f(\mathbf{x}^n;\theta)) \odot \mathbf{b}^n\|_{L_2}^2 + \frac{\lambda_2}{2} \cdot (\|w_1\|_{L_2}^2 + \cdots + \|w_K\|_{L_2}^2), \tag{7}$$

where $\lambda_2 > 0$ represents the regularization factor to prevent I-AutoRec from overfitting. The parameter set $\theta = \{w_1, \ldots, w_k, b_1, \ldots, b_k\}$ includes the weighted terms w_k and the intercept terms b_k of the hidden layers, where $k \in \{1, 2, \ldots, K\}$, $\mathbf{b}^n$ represents the n-th column of the index matrix B, and $\mathbf{x}^n$ corresponds to the item vector $\mathbf{x}^n = \{x_{1n}, \ldots, x_{|M|n}\}$.

4.3. Self-Adaptive Aggregation

Ensemble learning is a practical approach to combining multiple models. It is essential for the base models to exhibit diversity and accuracy [13]. To ensure diversity, we employ different types of models. Additionally, the representative LFA-based model and Autoencoder-based I-AutoRec ensure accuracy. As a result, the base models fulfil the two requirements for ensemble learning. To aggregate the models, we adopt a self-adaptive aggregation method based on their loss values on the validation set. The underlying principle is to increase the weight of the t-th base model if its loss decreases in the i-th training iteration or otherwise decreases. To comprehensively understand this idea, we will introduce relevant definitions to facilitate theoretical analysis.

Definition 3 (**Fractional *Loss* of Base Models):** *The fractional loss of the t-th base model at the i-th iteration, denoted as $Fl^t(i)$, is computed as follows:*

$$Fl^t(i) = \sqrt{\sum_{m \in M, n \in N, (M, N \in \Gamma)} ((x_{mn} - \hat{x}_{mn}^t) \times m_{mn})^2 / \|T\|_0}$$
$$\hat{x}_{mn}^t = \begin{cases} \sum_{k=1}^{d} u_{mk} v_{kn} & \text{if } t = 1 \\ f(m,n;\theta) & \text{if } t = 2 \end{cases}, \tag{8}$$

where $||\cdot||_0$ represents the L_0-norm of a matrix which calculates the number of non-zero elements of it, and Γ is the validation subset of X.

Definition 4 (Cumulative *Loss* of Base Models): *We let $Cl^t(i)$ be the cumulative loss of Fl^t until the i-th training iteration and calculate as follows:*

$$Cl^t(i) = \sum_{j=1}^{i} Sl^t(j). \tag{9}$$

Definition 5 (Ensemble Weights): *The ensemble weight Ew^t for the t-th base model can be computed using the following formula:*

$$Ew^t(i) = \frac{e^{-\delta Cl^t(i)}}{\sum\limits_{l=1}^{2} e^{-\delta Cl^t(i)}}. \tag{10}$$

Here, δ represents the equilibrium factor that controls the ensemble weights of the aggregation during the training process. Considering Definitions 3 to 5, the final prediction of AutoLFA in the i-th training iteration can be denoted as:

$$\hat{x}_{mn} = \sum_{t=1}^{2} El^t(i) \cdot \hat{x}_{mn}^t. \tag{11}$$

4.4. Theoretical Analysis

The loss of the AutoLFA model at the *i*-th training iteration is represented as *Fl*(*i*) and computed as follows:

$$Fl(n) = \sqrt{\sum_{m\in M, n\in N, (M,N\in\Gamma)} ((x_{mn} - \hat{x}_{mn}) \times b_{mn})^2 / \parallel \Gamma \parallel_0}, \tag{12}$$

where $\hat{x}_{mn}$ is calculated using align (11).

Definition 6 (Cumulative *Loss* of AutoLFA): *The cumulative loss of the AutoLFA model is represented as Cl(i) and can be expressed as:*

$$Cl(i) = \sum_{j=1}^{i} Fl(j). \tag{13}$$

Theorem 1. *For an AutoLFA model, assuming the $Cl^t(i)$ of the base models lies between [0, 1], and if $Ew^t(i)$ is set according to align (10) during training, the following alignment holds:*

$$Cl(I) \leq \min\{Cl^t(I) \mid t = 1,2\} + \frac{\ln 4}{\delta} + \frac{\delta I}{8}, \tag{14}$$

where I is the maximum iteration.

By setting $\delta = \sqrt{1/\ln I}$ in Theorem 1, the upper bound becomes:

$$Cl(I) \leq \min\{Cl^t(I)|t = 1,2\} + \ln 2\sqrt{\ln I} + \frac{I}{8\sqrt{\ln I}}, \tag{15}$$

where $\ln 2\sqrt{\ln I} + \frac{I}{8\sqrt{\ln I}}$ is bound by *I* linearly. This leads us to the following proposition.

Proposition 1. *With $\delta = \sqrt{1/\ln I}$, the inequality holds:*

$$Cl(I) \leq \min\{Cl^t(I)|t = 1, 2\} + const, \tag{16}$$

where the limit as I approaches infinity, const = 19.45.

Remark 1. *Proposition 1 indicates that Cl(I) is constrained by min{Cl^t(I) ∣ t = 1, 2} + const, with $\delta = \sqrt{1/\ln I}$. Remarkably, each base variant with a different foundation allows them to exist in separate metric spaces. The ensemble weight in align (10) ensures that the AutoLFA model's loss is always lower than the base models and benefits from the capabilities derived from the LFA and DNN-based models. Additionally, Proposition 1 is not intended to demonstrate the accuracy improvement of AutoLFA on the test set but rather to establish that the model possesses the advantages of the basic models. By showing that the proposed model achieves a smaller loss compared to each basic model used separately, it indicates that the model retains the respective strengths of the basic models without compromising its ability to fit the data.*

5. Experiments

In this section, we aim to address the following research questions (RQs) through subsequent experiments:

- RQ 1: Does the proposed AutoLFA model outperform state-of-the-art models in accurately predicting user behavior data?
- RQ 2: How does the AutoLFA model self-adaptively control the ensemble weights of its base models during the training process to ensure optimal performance?
- RQ 3: Are the base models of AutoLFA diversified in their ability to represent the same user behavior data matrix, thereby enhancing the performance of AutoLFA?
- RQ 4: What is the impact of the number of Latent Features and hidden units in the base models on the accuracy of AutoLFA?

5.1. General Settings

Datasets: For our experiments, we utilize five commonly used user-item datasets, as summarized in Table 1 These datasets include MovieLens_1M, MovieLens_100k, and MovieLens_HetRec from the MovieLens website, the Yahoo dataset from the Yahoo website, and the Douban dataset obtained from an open-access code. Table 1 summarizes the details of these datasets. The datasets are divided into train–validate–test sets using a ratio of 70%–10%–20%.

Table 1. Properties of all the datasets.

No.	Name	∣M∣	∣N∣	∣HO∣	Density *
D1	MovieLens_1M	6040	3952	1,000,209	4.19%
D2	MovieLens_100k	943	1682	100,000	6.30%
D3	MovieLens_HetRec	2113	10,109	855,598	4.01%
D4	Yahoo	15,400	1000	365,704	2.37%
D5	Douban	3000	3000	136,891	1.52%

* Density denotes the percentage of observed entries in the user-item matrix.

Evaluation Metrics: The primary objective of representing the user-item matrix is to predict missing ratings accurately. To assess the prediction accuracy of the tested models, we employ two evaluation metrics: root mean square error (RMSE) and mean absolute error (MAE), which are calculated according to [52].

Baselines: Our proposed MMA model is compared against seven state-of-the-art models: AutoRec (an original model), MF, and FML (Latent Feature Analysis-based models), and NRR, SparseFC, IGMC, and GLocal-K (deep-learning models). A brief description of these competing models is provided in Table 2.

Table 2. Descriptions of all the contrasting models.

Model	Description
MF [10]	A representative LFA-based model for factorizing user-item matrix data in recommender systems. *Computer 2009.*
AutoRec [41]	A notable DNNs-based model for representing user-item data in recommender systems. *WWW 2015.*
NRR [11]	A DNNs-based multi-task learning framework for rating prediction in recommender systems. *SIGIR 2017.*
SparseFC [27]	A DNNs-based model that reparametrizes weight matrices into low-dimensional vectors to capture important features. *ICML 2018.*
IGMC [55]	A GNNs-based model for inductive matrix completion without using side information. *ICLR 2019.*
FML [9]	An LFA-based model that combines metric learning (distance space) and collaborative filtering. *IEEE TII 2020.*
GLocal-K [57]	A DNNs-based model for generalizing and representing user-item data in a low-dimensional space with important features. *CIKM 2021.*

Implementation Details: For all datasets, we set the learning rate to 0.001 for two models. We set the number of hidden units for the Autoencoder to 500 and the number of latent factors for the LFA model 30 to achieve better performance. The final testing results are obtained from the best-performing model, which exhibits the lowest prediction error on the validation set during training. The training process terminates when the preset threshold for training iterations is reached. All experiments are conducted on a GPU server with two 2.4 GHz Xeon Gold 6240 R CPUs, 376.40 GB RAM, and 4 Tesla V100 GPUs.

5.2. Performance Comparison (RQ. 1)

5.2.1. Comparison of Prediction Accuracy

Table 3 presents the prediction accuracies of all models from D1 to D5. Statistical tests, including loss/tie/win analysis, the Wilcoxon signed-ranks test [60], and the Friedman test [21], are performed to analyze these results. The loss/tie/win analysis identifies cases where AutoLFA's RMSE/MAE is higher/same/lower than other competitors. The Wilcoxon signed-ranks test is a non-parametric pairwise comparison method that determines if AutoLFA's prediction accuracy is significantly higher than each comparison model based on *p*-values. The Friedman test compares the performance of multiple models across multiple datasets using F-rank values, with lower values indicating higher prediction accuracy. The comparative experiment results are normalized for better interpretation before conducting the Wilcoxon signed-ranks test and the Friedman test. The statistical analysis results of loss/tie/win, the Wilcoxon signed-ranks test, and the Friedman test are presented in the third-to-last, second-to-last, and last rows of Table 3. Key observations from Table 3 are as follows:

- AutoLFA achieves the lowest RMSE/MAE in most cases, with only ten cases of loss and one case of a tie in comparison. The total count of loss/tie/win cases is 7/1/62.
- All *p*-values are below the significance level of 0.1, indicating that AutoLFA outperforms all competitors in terms of prediction accuracy.
- AutoLFA obtains the lowest F-rank among all participants, confirming its highest accuracy across all datasets.

These observations highlight that AutoLFA achieves the highest prediction accuracy for predicting missing user data compared to other models.

Table 3. Performance comparison of AutoLFA and its competitors.

Dataset	Metric	MF	AutoRec	NRR	SparseFC	IGMC	FML	Glocal-K	AutoLFA
D1	RMSE	0.857 ●	0.847 ●	0.881 ●	0.839 ○	0.867 ●	0.849 ●	0.839 ○	0.842
	MAE	0.673 ●	0.667 ●	0.691 ●	0.656 ○	0.681 ●	0.667 ●	0.655 ○	0.664
D2	RMSE	0.913 ●	0.897 ●	0.923 ●	0.899 ●	0.915 ●	0.904 ●	0.892 ●	0.887
	MAE	0.719 ●	0.706 ●	0.725 ●	0.706 ●	0.722 ●	0.718 ●	0.697	0.699
D3	RMSE	0.757 ●	0.752 ●	0.774 ●	0.749 ●	0.769 ●	0.754 ●	0.756 ●	0.744
	MAE	0.572 ●	0.569 ●	0.583 ●	0.567 ●	0.582 ●	0.573 ●	0.573 ●	0.562
D4	RMSE	1.206 ●	1.172 ●	1.227 ●	1.203 ●	1.133 ○	1.176 ●	1.204 ●	1.167
	MAE	0.937 ●	0.900 ●	0.949 ●	0.915 ●	0.848 ○	0.937 ●	0.905 ●	0.895
D5	RMSE	0.738 ●	0.744 ●	0.726 ●	0.745 ●	0.751 ●	0.762 ●	0.737	0.737
	MAE	0.588 ●	0.588 ●	0.573 ●	0.587 ●	0.594 ●	0.598 ●	0.580 ○	0.584
Statistic	loss/tie/win	0/0/10	0/0/10	0/0/10	2/0/8	2/0/8	0/0/10	3/1/6	**7/1/62** *
	p-value	0.0039	0.0039	0.0039	0.039	0.0195	0.039	0.0977	-
	F-rank	5.7	3.75	6.6	3.5	5.9	5.45	3.05	**2.05**

* The total loss/tie/win cases of AutoLFA. ● The cases in which AutoLFA wins the other models in comparison. ○ The cases in which AutoLFA loses the comparison.

5.2.2. Comparison of Computational Efficiency

Figure 2 depicts the total time required for all participating models to reach the optimal RMSE on the validation dataset during training. The following observations can be made:

- LFA-based models generally exhibit higher computational efficiency compared to DNN-based models, as they are trained on observed user behavior data, unlike DNN-based models.
- Due to their complex data form and architecture, GNN-based models consume significant computational resources and time. From Figure 2, it is evident that IGMC surpasses 3000 s in time costs.
- Except for slightly longer time consumption on dataset D4, AutoLFA's time consumption falls between LFA-based and GNN-based models. It is slightly higher than the original Autoencoder-based model but faster than other DNN-based models in most cases.

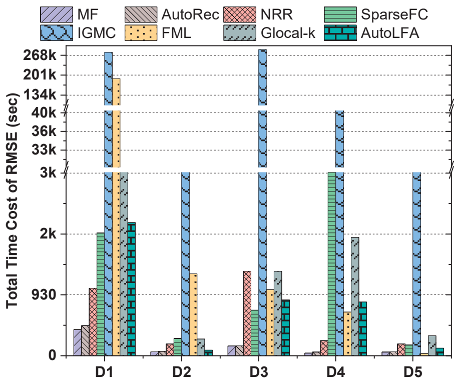


Figure 2. The histogram graph of the total time cost to reach the optimal accuracy of all the participating models.

Based on these observations, we can conclude that the relatively simple structure of the base models in AutoLFA allows for acceptable total time costs after ensembling two base models.

5.3. The Self-Ensembling of MMA (RQ. 2)

To discuss the self-adaptive control of AutoLFA in ensembling different variant models and ensuring its performance, we monitor the variations of ensemble weights between its base models.

Monitoring ensemble weight variations: Figure 3 illustrates the changes in ensemble weights from D1 to D5, yielding the following observations:

- In most cases (e.g., Figure 3a–d), the ensemble weights of the Autoencoder-based model gradually increase and surpass the LFA-based model as the training progresses until the base models are fitted.
- In some instances, the LFA-based model's weight may exceed that of the Autoencoder-based model. For example, in Figure 3e, the ensemble weight of the LFA-based model is greater than those of the Autoencoder-based model due to their faster convergence.

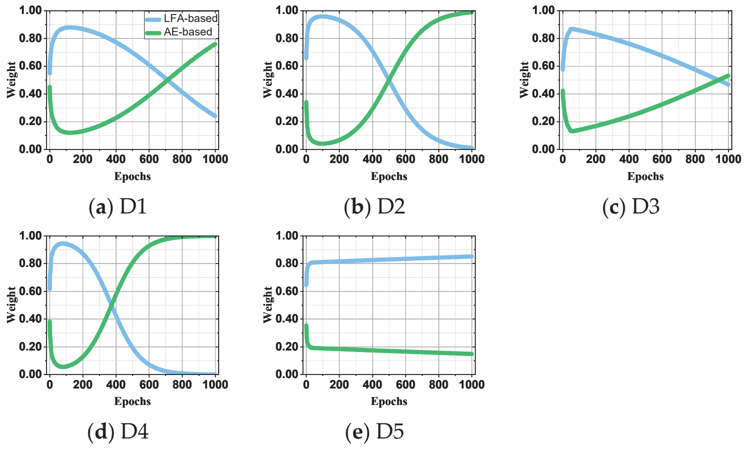


Figure 3. The changes in ensemble weights during the training process.

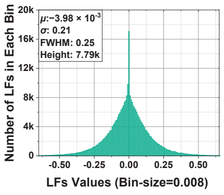
In conclusion, based on the experimental results and observations above, we can infer that AutoLFA effectively leverages different types of models. By aggregating these models in the ensemble stage, AutoLFA surpasses other state-of-the-art models in predicting missing ratings with only minor sacrifices in computational resources.

5.4. Distribution of Latent Features of Base Models (RQ. 3)

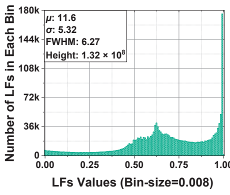
In order to investigate the diversity of the base models of AutoLFA and their abilities to predict the user behavior data matrix, we visually analyze the encoder output of the Autoencoder-based model, which represents the Latent Features of an Autoencoder model, and the Latent Features of the LFA-based model. The distribution of these Latent Features for the base models across all datasets is depicted in Figure 4. To analyze the distribution, we employ a Gaussian function and examine factors such as expectation (μ) and standard deviation (σ). The measurements of the full width at half maximum (FWHM) and the height of the Gaussian curve are also presented. From Figure 4, the following observations emerge:

- The distribution of Latent Features in the Autoencoder-based model tends to have more values concentrated at the extremes (i.e., 0 or 1), as shown in Figure 4f,h,i, while in the LFA-based model, the distribution tends to follow a normal distribution.
- After encoding, the Autoencoder-based model's Latent Features are more likely to exhibit unusually high values within specific ranges. In contrast, in the LFA-based model, there are no extreme values, as depicted in Figure 4a,c,d.

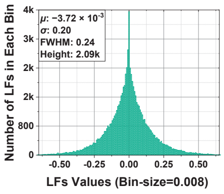
- In some cases, the distribution of Latent Features in the Autoencoder-based model appears to be slightly more uniform compared to the LFA-based model, as illustrated in Figure 4e,j.



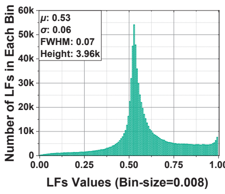
(**a**) LFA-based model on D1



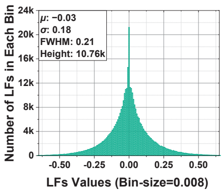
(**f**) AE-based model on D1



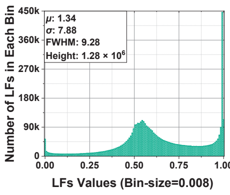
(**b**) LFA-based model on D2



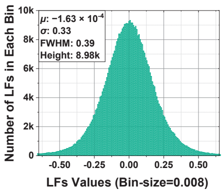
(**g**) AE-based model on D2



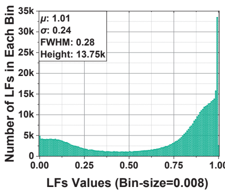
(**c**) LFA-based model on D3



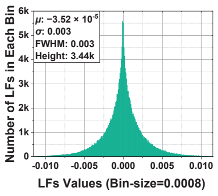
(**h**) AE-based model on D3



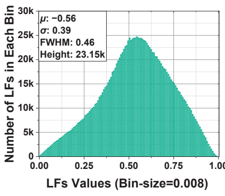
(**d**) LFA-based model on D4



(**i**) AE-based model on D4



(**e**) LFA-based model on D5



(**j**) AE-based model on D5

Figure 4. The distribution histogram of LFs of LFA-based and Autoencoder-based models from D1 to D5.

The observed information above indicates that the Autoencoder-based and LFA-based models have distinct representation characteristics, allowing AutoLFA to benefit from their different representation abilities. Consequently, AutoLFA ensures accurate prediction of missing ratings.

5.5. Influence of Numbers of Latent Features and Hidden Units to Base Models (RQ. 4)

We further investigate the impact of the number of Latent Features and hidden units in the base models on AutoLFA. Figure 5 illustrates the RMSE and MAE of AutoLFA as the number of Latent Features and hidden units varies simultaneously across D1 to D5. The following observations can be made from Figure 5:

- Increasing the number of Latent Features/hidden units from 2/20 to 20/300 results in a rapid improvement in the accuracy of AutoLFA. During this range, AutoLFA substantially increases accuracy without incurring significant computational costs.
- Once the number of Latent Features/hidden units reaches 25/400, the rate of accuracy improvement becomes less prominent in Figure 5b–d.

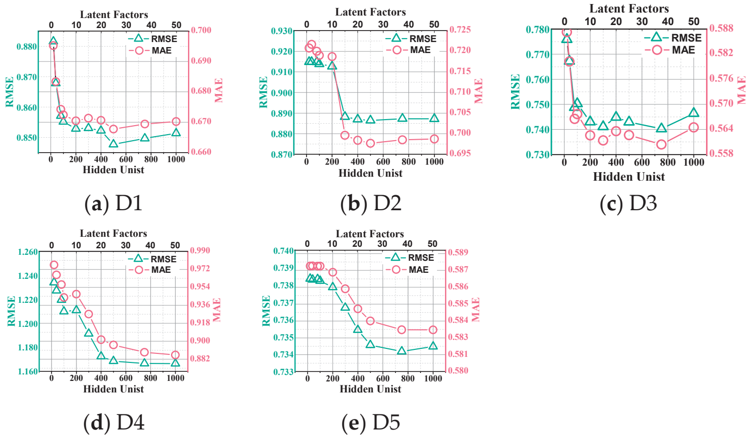


Figure 5. The line graphs of RMSE and MAE of AutoLFA from D1 to D5 as the number of Hidden Units and Latent Factors vary.

These observations suggest that setting the number of Latent Features/hidden units as 30/500 allows AutoLFA to achieve optimal accuracy in most cases without imposing significant computational resource demands. Although this setting may not yield the highest accuracy in certain cases, it remains relatively close to the optimal value.

6. Conclusions

This paper proposes a novel hybrid recommendation model by combining Autoencoder and LFA models, termed AutoLFA. Its main idea is two-fold: (1) It leverages an Autoencoder and a Latent Feature Analysis (LFA) model separately to construct two distinct recommendation models, each residing in a unique metric representation space with its own set of strengths, and (2) it integrates the Autoencoder and LFA models using a customized self-adaptive weighting strategy. As such, the merits of the LFA and DNN-based models are combined into the AutoLFA model, making it achieve superior recommendation performance under various real-world applications. The experiments investigate four research questions on five real recommendation datasets. The results verify that the proposed AutoLFA outperforms several state-of-the-art models. In the future, we plan to aggregate more variants of LFA-based and deep neural networks (DNNs)-based models to achieve better recommendation performance.

Author Contributions: Formal analysis, X.L.; investigation, Y.L.; methodology, S.G.; software, K.X.; validation, G.L.; writing—original draft, S.G.; writing—review and editing, C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All data supporting the reported results in our study are publicly available and can be accessed through the following link: https://grouplens.org/datasets/movielens/, https://webscope.sandbox.yahoo.com/catalog.php?datatype=r, https://github.com/fmonti/mgcnn.

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

References

1. Deng, S.; Zhai, Y.; Wu, D.; Yue, D.; Fu, X.; He, Y. A Lightweight Dynamic Storage Algorithm With Adaptive Encoding for Energy Internet. *IEEE Trans. Serv. Comput.* **2023**, 1–14. [CrossRef]
2. He, Y.; Wu, B.; Wu, D.; Beyazit, E.; Chen, S.; Wu, X. Toward Mining Capricious Data Streams: A Generative Approach. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 1228–1240. [CrossRef] [PubMed]
3. Wang, D.; Liang, Y.; Xu, D.; Feng, X.; Guan, R. A content-based recommender system for computer science publications. *Knowl.-Based Syst.* **2018**, *157*, 1–9. [CrossRef]
4. Bai, X.; Huang, M.; Xu, M.; Liu, J. Reconfiguration Optimization of Relative Motion Between Elliptical Orbits Using Lyapunov-Floquet Transformation. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, *59*, 923–936. [CrossRef]
5. You, D.; Niu, S.; Dong, S.; Yan, H.; Chen, Z.; Wu, D.; Shen, L.; Wu, X. Counterfactual explanation generation with minimal feature boundary. *Inf. Sci.* **2023**, *625*, 342–366. [CrossRef]
6. You, D.; Xiao, J.; Wang, Y.; Yan, H.; Wu, D.; Chen, Z.; Shen, L.; Wu, X. Online Learning From Incomplete and Imbalanced Data Streams. *IEEE Trans. Knowl. Data Eng.* **2023**, 1–14. [CrossRef]
7. Chen, J.; Wang, Q.; Peng, W.; Xu, H.; Li, X.; Xu, W. Disparity-based multiscale fusion network for transportation detection. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 18855–18863. [CrossRef]
8. Cao, B.; Zhao, J.; Lv, Z.; Yang, P. Diversified personalized recommendation optimization based on mobile data. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 2133–2139. [CrossRef]
9. Zhang, S.; Yao, L.; Wu, B.; Xu, X.; Zhang, X.; Zhu, L. Unraveling metric vector spaces with factorization for recommendation. *IEEE Trans. Ind. Inform.* **2019**, *16*, 732–742. [CrossRef]
10. Koren, Y.; Bell, R.; Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer* **2009**, *42*, 30–37.
11. Li, P.; Wang, Z.; Ren, Z.; Bing, L.; Lam, W. Neural rating regression with abstractive tips generation for recommendation. In Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, 7–11 August 2017; pp. 345–354.
12. Wu, D.; Luo, X. Robust Latent Factor Analysis for Precise Representation of High-Dimensional and Sparse Data. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 796–805. [CrossRef]
13. Wu, D.; Shang, M.; Luo, X.; Wang, Z. An L1-and-L2-Norm-Oriented Latent Factor Model for Recommender Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 5775–5788. [CrossRef]
14. Luo, X.; Wu, H.; Li, Z. Neulft: A Novel Approach to Nonlinear Canonical Polyadic Decomposition on High-Dimensional Incomplete Tensors. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 6148–6166. [CrossRef]
15. Wu, D.; He, Y.; Luo, X.; Zhou, M. A Latent Factor Analysis-Based Approach to Online Sparse Streaming Feature Selection. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 6744–6758. [CrossRef]
16. Wu, D. *Robust Latent Feature Learning for Incomplete Big Data*; Springer Nature: Berlin, Germany, 2022.
17. Liu, H.; Yuan, H.; Hou, J.; Hamzaoui, R.; Gao, W. Pufa-gan: A frequency-aware generative adversarial network for 3d point cloud upsampling. *IEEE Trans. Image Process.* **2022**, *31*, 7389–7402. [CrossRef] [PubMed]
18. Liu, Q.; Yuan, H.; Hamzaoui, R.; Su, H.; Hou, J.; Yang, H. Reduced reference perceptual quality model with application to rate control for video-based point cloud compression. *IEEE Trans. Image Process.* **2021**, *30*, 6623–6636. [CrossRef]
19. Liu, X.; He, J.; Liu, M.; Yin, Z.; Yin, L.; Zheng, W. A Scenario-Generic Neural Machine Translation Data Augmentation Method. *Electronics* **2023**, *12*, 2320. [CrossRef]
20. Liu, X.; Zhao, J.; Li, J.; Cao, B.; Lv, Z. Federated neural architecture search for medical data security. *IEEE Trans. Ind. Inform.* **2022**, *18*, 5628–5636. [CrossRef]
21. Huang, T.; Liang, C.; Wu, D.; He, Y. A Debiasing Autoencoder for Recommender System. *IEEE Trans. Consum. Electron.* **2023**. [CrossRef]
22. Zenggang, X.; Mingyang, Z.; Xuemin, Z.; Sanyuan, Z.; Fang, X.; Xiaochao, Z.; Yunyun, W.; Xiang, L. Social similarity routing algorithm based on socially aware networks in the big data environment. *J. Signal Process. Syst.* **2022**, *94*, 1253–1267. [CrossRef]
23. Xiong, Z.; Li, X.; Zhang, X.; Deng, M.; Xu, F.; Zhou, B.; Zeng, M. A Comprehensive Confirmation-based Selfish Node Detection Algorithm for Socially Aware Networks. *J. Signal Process. Syst.* **2023**, 1–19. [CrossRef]
24. Xie, L.; Zhu, Y.; Yin, M.; Wang, Z.; Ou, D.; Zheng, H.; Liu, H.; Yin, G. Self-feature-based point cloud registration method with a novel convolutional Siamese point net for optical measurement of blade profile. *Mech. Syst. Signal Process.* **2022**, *178*, 109243. [CrossRef]
25. Wu, D.; Sun, B.; Shang, M. Hyperparameter Learning for Deep Learning-based Recommender Systems. *IEEE Trans. Serv. Comput.* **2023**, 1–13. [CrossRef]
26. Chen, F.; Yin, G.; Dong, Y.; Li, G.; Zhang, W. KHGCN: Knowledge-Enhanced Recommendation with Hierarchical Graph Capsule Network. *Entropy* **2023**, *25*, 697. [CrossRef] [PubMed]

27. Muller, L.; Martel, J.; Indiveri, G. Kernelized synaptic weight matrices. In *International Conference on Machine Learning*; PMLR: London, UK, 2018; pp. 3654–3663.
28. Han, H.; Liang, Y.; Bella, G.; Giunchiglia, F.; Li, D. LFDNN: A Novel Hybrid Recommendation Model Based on DeepFM and LightGBM. *Entropy* **2023**, *25*, 638. [CrossRef]
29. Zhang, S.; Yao, L.; Sun, A.; Tay, Y. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.* **2019**, *52*, 1–38. [CrossRef]
30. Lu, S.; Liu, M.; Yin, L.; Yin, Z.; Liu, X.; Zheng, W. The multi-modal fusion in visual question answering: A review of attention mechanisms. *PeerJ Comput. Sci.* **2023**, *9*, e1400. [CrossRef]
31. Shen, X.; Jiang, H.; Liu, D.; Yang, K.; Deng, F.; Lui, J.C.; Liu, J.; Dustdar, S.; Luo, J. PupilRec: Leveraging Pupil Morphology for Recommending on Smartphones. *IEEE Internet Things J.* **2022**, *9*, 15538–15553. [CrossRef]
32. Shen, Y.; Ding, N.; Zheng, H.-T.; Li, Y.; Yang, M. Modeling relation paths for knowledge graph completion. *IEEE Trans. Knowl. Data Eng.* **2020**, *33*, 3607–3617. [CrossRef]
33. Wang, H.; Wang, B.; Luo, P.; Ma, F.; Zhou, Y.; Mohamed, M.A. State evaluation based on feature identification of measurement data: For resilient power system. *CSEE J. Power Energy Syst.* **2021**, *8*, 983–992.
34. Wang, Y.; Xu, N.; Liu, A.-A.; Li, W.; Zhang, Y. High-order interaction learning for image captioning. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *32*, 4417–4430. [CrossRef]
35. Luo, X.; Zhou, Y.; Liu, Z.; Zhou, M. Fast and Accurate Non-Negative Latent Factor Analysis of High-Dimensional and Sparse Matrices in Recommender Systems. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 3897–3911. [CrossRef]
36. Chen, J.; Xu, M.; Xu, W.; Li, D.; Peng, W.; Xu, H. A Flow Feedback Traffic Prediction Based on Visual Quantified Features. *IEEE Trans. Intell. Transp. Syst.* **2023**, 1–9. [CrossRef]
37. Deng, X.; Liu, E.; Li, S.; Duan, Y.; Xu, M. Interpretable Multi-modal Image Registration Network Based on Disentangled Convolutional Sparse Coding. *IEEE Trans. Image Process.* **2023**, *32*, 1078–1091. [CrossRef]
38. Fan, W.; Yang, L.; Bouguila, N. Unsupervised grouped axial data modeling via hierarchical Bayesian nonparametric models with Watson distributions. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 9654–9668. [CrossRef] [PubMed]
39. Guo, C.; Hu, J. Fixed-Time Stabilization of High-Order Uncertain Nonlinear Systems: Output Feedback Control Design and Settling Time Analysis. *J. Syst. Sci. Complex.* **2023**, 1–22. [CrossRef]
40. Huang, H.; Xue, C.; Zhang, W.; Guo, M. Torsion design of CFRP-CFST columns using a data-driven optimization approach. *Eng. Struct.* **2022**, *251*, 113479. [CrossRef]
41. Sedhain, S.; Menon, A.K.; Sanner, S.; Xie, L. Autorec: Autoencoders meet collaborative filtering. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 111–112.
42. Wu, D.; Luo, X.; Shang, M.; He, Y.; Wang, G.; Wu, X. A Data-Characteristic-Aware Latent Factor Model for Web Services QoS Prediction. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 2525–2538. [CrossRef]
43. Luo, X.; Zhou, M.; Li, S.; Wu, D.; Liu, Z.; Shang, M. Algorithms of Unconstrained Non-Negative Latent Factor Analysis for Recommender Systems. *IEEE Trans. Big Data* **2021**, *7*, 227–240. [CrossRef]
44. Yuan, Y.; Luo, X.; Shang, M.; Wu, D. A generalized and fast-converging non-negative latent factor model for predicting user preferences in recommender systems. In Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 498–507.
45. Ren, X.; Song, M.; Haihong, E.; Song, J. Context-aware probabilistic matrix factorization modeling for point-of-interest recommendation. *Neurocomputing* **2017**, *241*, 38–55. [CrossRef]
46. Wu, D.; Jin, L.; Luo, X. PMLF: Prediction-Sampling-Based Multilayer-Structured Latent Factor Analysis. In Proceedings of the 2020 IEEE International Conference on Data Mining (ICDM), Sorrento, Italy, 17–20 November 2020; pp. 671–680.
47. Wu, D.; Luo, X.; He, Y.; Zhou, M. A Prediction-Sampling-Based Multilayer-Structured Latent Factor Model for Accurate Representation to High-Dimensional and Sparse Data. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, 1–14. [CrossRef]
48. Wang, C.; Liu, Q.; Wu, R.; Chen, E.; Liu, C.; Huang, X.; Huang, Z. Confidence-aware matrix factorization for recommender systems. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 434–442.
49. Wu, D.; He, Q.; Luo, X.; Shang, M.; He, Y.; Wang, G. A Posterior-Neighborhood-Regularized Latent Factor Model for Highly Accurate Web Service QoS Prediction. *IEEE Trans. Serv. Comput.* **2022**, *15*, 793–805. [CrossRef]
50. Wu, D.; Zhang, P.; He, Y.; Luo, X. A Double-Space and Double-Norm Ensembled Latent Factor Model for Highly Accurate Web Service QoS Prediction. *IEEE Trans. Serv. Comput.* **2023**, *16*, 802–814. [CrossRef]
51. Leng, C.; Zhang, H.; Cai, G.; Cheng, I.; Basu, A. Graph regularized Lp smooth non-negative matrix factorization for data representation. *IEEE/CAA J. Autom. Sin.* **2019**, *6*, 584–595. [CrossRef]
52. Wu, D.; Luo, X.; Shang, M.; He, Y.; Wang, G.; Zhou, M. A Deep Latent Factor Model for High-Dimensional and Sparse Matrices in Recommender Systems. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 4285–4296. [CrossRef]
53. Sun, B.; Wu, D.; Shang, M.; He, Y. Toward auto-learning hyperparameters for deep learning-based recommender systems. In Proceedings of the International Conference on Database Systems for Advanced Applications: 27th International Conference, DASFAA 2022, Virtual Event, 11–14 April 2022; Proceedings, Part II. Springer: Cham, Switzerland, 2022; pp. 323–331.
54. Wang, Q.; Peng, B.; Shi, X.; Shang, T.; Shang, M. DCCR: Deep collaborative conjunctive recommender for rating prediction. *IEEE Access* **2019**, *7*, 60186–60198. [CrossRef]

55. Zhang, M.; Chen, Y. Inductive matrix completion based on graph neural networks. In Proceedings of the International Conference on Learning Representations, Formerly Addis Ababa, Ethiopia, 26 April–1 May 2020.
56. He, X.; Chua, T.-S. Neural factorization machines for sparse predictive analytics. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; pp. 355–364.
57. Han, S.C.; Lim, T.; Long, S.; Burgstaller, B.; Poon, J. GLocal-K: Global and Local Kernels for Recommender Systems. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Virtual Event, Australia, 1–5 November 2021; pp. 3063–3067.
58. Xiao, J.; Ye, H.; He, X.; Zhang, H.; Wu, F.; Chua, T.-S. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI, Melbourne, Australia, 19–25 August 2017; pp. 3119–3125.
59. Kim, D.H.; Park, C.; Oh, J.; Lee, S.; Yu, H. Convolutional matrix factorization for document context-aware recommendation. In Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016; pp. 233–240.
60. Wu, D.; Luo, X.; Wang, G.; Shang, M.; Yuan, Y.; Yan, H. A Highly Accurate Framework for Self-Labeled Semisupervised Classification in Industrial Applications. *IEEE Trans. Ind. Inform.* **2018**, *14*, 909–920. [CrossRef]

entropy

Article

Chinese Few-Shot Named Entity Recognition and Knowledge Graph Construction in Managed Pressure Drilling Domain

Siqing Wei [1], Yanchun Liang [1,2], Xiaoran Li [1], Xiaohui Weng [3], Jiasheng Fu [4] and Xiaosong Han [1,*]

1 Key Laboratory for Symbol Computation and Knowledge Engineering of National Education Ministry, College of Computer Science and Technology, Jilin University, Changchun 130012, China; weisq22@mails.jlu.edu.cn (S.W.); ycliang@jlu.edu.cn (Y.L.); xiaoran21@mails.jlu.edu.cn (X.L.)
2 Zhuhai Laboratory of Key Laboratory for Symbol Computation and Knowledge Engineering of Ministry of Education, Zhuhai College of Science and Technology, Zhuhai 519041, China
3 School of Mechanical and Aerospace Engineering, Jilin University, Changchun 130012, China; wengxiaohui@jlu.edu.cn
4 CNPC Engineering Technology R&D Company Limited, National Engineering Research Center of Oil & Gas Drilling and Completion Technology, Beijing 102206, China; fujsdr@cnpc.com.cn
* Correspondence: hanxiaosong@jlu.edu.cn

Abstract: Managed pressure drilling (MPD) is the most effective means to ensure drilling safety, and MPD is able to avoid further deterioration of complex working conditions through precise control of the wellhead back pressure. The key to the success of MPD is the well control strategy, which currently relies heavily on manual experience, hindering the automation and intelligence process of well control. In response to this issue, an MPD knowledge graph is constructed in this paper that extracts knowledge from published papers and drilling reports to guide well control. In order to improve the performance of entity extraction in the knowledge graph, a few-shot Chinese entity recognition model CEntLM-KL is extended from the EntLM model, in which the KL entropy is built to improve the accuracy of entity recognition. Through experiments on benchmark datasets, it has been shown that the proposed model has a significant improvement compared to the state-of-the-art methods. On the few-shot drilling datasets, the F-1 score of entity recognition reaches 33%. Finally, the knowledge graph is stored in Neo4J and applied for knowledge inference.

Keywords: MPD; knowledge graphs; entity extraction; relation extraction; few shot

Citation: Wei, S.; Liang, Y.; Li, X.; Weng, X.; Fu, J.; Han, X. Chinese Few-Shot Named Entity Recognition and Knowledge Graph Construction in Managed Pressure Drilling Domain. *Entropy* **2023**, *25*, 1097. https://doi.org/10.3390/e25071097

Academic Editor: Vladimir Frid

Received: 29 May 2023
Revised: 14 July 2023
Accepted: 20 July 2023
Published: 22 July 2023

1. Introduction

The demand for energy is increasing with global population growth and economic development. Oil and gas are major energy sources, crucial for national and regional economic development, energy security, technological innovation, employment opportunities, and enhancing a country's status and influence. Therefore, oil and gas developments are of utmost importance. Drilling is a critical activity in oil and gas development, and managed pressure drilling (MPD) plays a key role in ensuring a safe and reliable drilling process while improving production efficiency. MPD is essential for safe work, resource protection, production efficiency, cost control, and environmental preservation. Effective MPD measures enable efficient, safe, and sustainable oil and gas development. In the process of using MPD for oil and gas development, a large amount of MPD field data have been generated, but these data are often disorganized and underutilized. Constructing a knowledge graph of the MPD domain can improve the current state of disorganized data and enhance their accessibility. Hence, this paper focuses on constructing a knowledge graph based on the aforementioned study. The construction of a knowledge graph involves fundamental steps such as knowledge extraction, knowledge fusion, and quality control. Entity extraction and relationship extraction represent typical knowledge extraction tasks.

Named entity recognition (NER), also known as entity extraction, is primarily focused on identifying the text range of entities and categorizing them into predefined categories. Four common methods are employed for NER.

Rule-based methods: These methods utilize manually written rules and patterns to identify named entities. While simple and easy to implement, they require extensive manual effort and are not suitable for complex entity recognition tasks.

Dictionary-based methods: This approach utilizes preconstructed dictionaries or knowledge bases to match entities in text. It offers the advantage of speed and high customization but fails to recognize entities not present in the lexicon.

Machine learning-based methods: Machine learning algorithms, such as conditional random fields (CRF), support vector machines, maximum entropy Markov models, and deep learning models, are employed to learn and predict named entities in text. These methods require well-labeled training data for effective model training and have the capability to automatically learn features and patterns for various entity recognition tasks.

Deep learning-based methods: Deep learning-based models, such as long short-term memory networks (LSTM), have achieved significant advancements in NER. These models capture contextual information and semantic features in text. With large-scale labeled training data, they automatically learn and extract features, leading to successful NER. In recent years, deep learning-based methods have become mainstream approaches, exhibiting significant performance improvements in NER tasks. For example, Dai et al. [1] employed a BERT-BiLSTM-CRF architecture to extract named entities from Chinese electronic medical records.

Few-shot NER addresses the challenge of data scarcity in NER tasks. Traditional approaches require a large number of labeled samples to train the model effectively. However, practical applications often involve domain-specific or type-specific named entities with limited sample data, posing a challenge for model training. Few-shot NER aims to address this challenge [2–6]. Common methods include transfer learning, meta-learning, active learning, data augmentation, and distant supervision. Prompt-based learning has recently achieved significant success in few-shot classification tasks. The success of prompt-based learning depends on two factors: (1) reusing masked language model targets to bridge the gap between pre-training and fine-tuning targets [7–9]; and (2) designing complex templates and labeled words to help the language model adapt to task-specific answer distributions, thereby improving performance with fewer samples. Manual selection [2,3], gradient-based discrete search [10], language model generation [11], and continuous optimization [12] are popular approaches used to induce appropriate answers in corresponding tasks.

However, template-based prompt tuning is primarily designed for sentence-level tasks and is challenging to adapt to token-level classification tasks like NER. The search for suitable templates becomes more difficult as the search space expands in NER, and relying on a small number of annotated samples can lead to overfitting. To address this issue, the Chinese EntLM with new loss (CEntLM-KL) model inspired by EntLM for Chinese few-shot entity extraction (CEntLM) is presented in this paper. The proposed model formulates the NER task as a language model (LM) problem without templates and is further applied to entity extraction, combined with the results of entity extraction based on Trie trees [13].

Entity relationship extraction (ERE) aims to extract semantic relationships between entities from text. It is a crucial task in natural language processing and information extraction, facilitating the understanding of connections between entities in a text and supporting various applications. ERE methods include rule-based extraction, supervised learning, distant supervised learning, semi-supervised learning, and transfer learning. In this paper, we adopt a rule-based method for relationship extraction in the construction of the knowledge graph, as it offers higher accuracy and simplicity. Rule formulation and relationship extraction are performed based on subject terms, entity location relationships, and keywords.

The main contributions of this paper are as follows.

Firstly, a novel model, CEntLM-KL, is proposed for Chinese few-shot NER. Given the presence of domain-specific or type-specific named entities in practical applications with limited sample data, a novel model, CEntLM-KL, is proposed for Chinese few-shot NER. Inspired by the EntLM [14] model, the CEntLM-KL model formulates the NER task as an LM problem without relying on templates. The model demonstrates good performance on both private and public datasets, including the People's Daily-NER Dataset and the few-shot dataset.

Secondly, in this paper, we complete the structure of the MPD domain knowledge graph, defining 14 entity types. Based on an analysis of the specific requirements of the MPD domain, structure of the Knowledge Graph design is completed, defining 14 types of entities (e.g., well class, well type, drilling fluid type, complex working conditions, performance of complex working conditions, handling measures for complex working conditions, and possible causes of complex working conditions) and eleven types of relationships (e.g., well class—drilling fluid type).

Finally, a knowledge base is built for the MPD domain in this paper. A domain knowledge base is built by crawling the Internet and CNKI related literature as well as collating relevant owned MPD data, including a structured knowledge base, a manually constructed domain dictionary based on the acquired knowledge, and a literature base consisting of published papers and drilling reports. Entity extraction and relationship extraction are performed sequentially using knowledge extraction in the data layer construction. For entity extraction, the unstructured text is annotated with BIO by the constructed domain dictionary to generate a labeled entity dataset. The relationship extraction uses a rule-based approach to ensure high accuracy and simplicity. Rule formulation and relationship extraction are performed based on subject terms, entity location relationships and keywords.

2. Methods

The main goal of this paper is to improve the template free NER model EntLM, and establish the MPD knowledge graph. In the process of constructing the knowledge graph, the first step is to use the existing data to construct the knowledge graph schema layer to obtain the entity types and relationship types required to construct the MPD knowledge graph. Then, the improved model CEntLM-KL is trained using the existing data, and the trained deep learning model is combined with the Trie tree method to extract entities from the input data. A rule-based method is used to extract relationships from the input data. Finally, Neo4J is used to store the obtained entities and relationships and construct a visual Knowledge graph. The overall framework of the paper is shown in Figure 1.

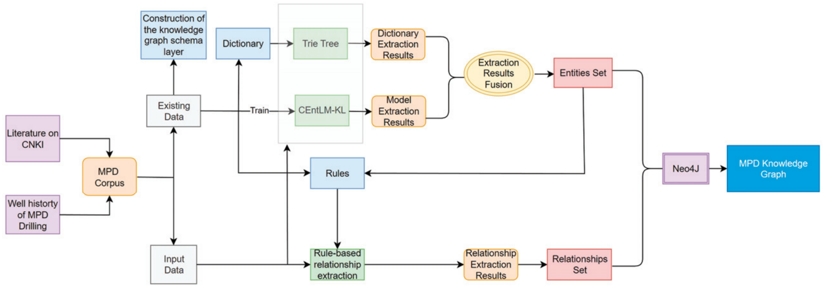

Construction of the knowledge graph schema layer
Dictionary
Trie Tree
Dictionary Extraction Results
Extraction Results Fusion
Entities Set
Existing Data
Train
CEntLM-KL
Model Extraction Results
Literature on CNKI
MPD Corpus
Well historty of MPD Drilling
Rules
Neo4J
MPD Knowledge Graph
Input Data
Rule-based relationship extraction
Relationship Extraction Results
Relationships Set

Figure 1. Pipeline of our work.

2.1. Template-Based Tuning

Template-based tuning is a technique used to optimize and fine-tune LMs by combining template design and fine-tuning approaches. In this technique, a template is designed with keywords and placeholders to guide the generation task. The LM is then fine tuned using input samples that contain these templates, allowing it to better understand and generate text that aligns with the template.

By employing template-based tuning, the LM can generate text that adheres to the template, resulting in improved quality and consistency of the generated output. This technique has potential applications in various natural language processing tasks, including text generation [15], dialogue systems [16], question-and-answer systems [17], recommendation system [18], and text clustering [19]. It can enhance the accuracy, coherence, and usefulness of the generated results, thereby improving the overall performance and robustness of the model.

However, when applied to NER, the template-based approach becomes more complex. In NER, the goal is to obtain a sequence of tokens corresponding to each character in the input. Therefore, an additional placeholder, denoted as $[S]$, needs to be added to the constructed template to accommodate the characters or character sequences. For example, the template for an LM model could be "$[X][S]$ is a $[Z]$ entity," and during training, the model predicts the labeled words at the $[Z]$ position.

During the decoding process, to obtain the label sequence for a given input, all possible spans of the input sequence must be enumerated and considered.

$$Y = \left\{ \begin{array}{c} argmax\ P\Big([Z] = M(Y)\Big|T_{prompt}\Big(X, s_j^i\Big)\Big), \\ s_j^i = Enumerate(\{x_i, \ldots\ldots, x_j\}, i, j \in \{1, n\}) \end{array} \right\} \tag{1}$$

where T_{prompt} is a function that converts input X into prompt input, and function M is a mapping function that connects the label word Y mapping to the label space.

As shown in Figure 2, let us consider an example with the phrase "Jenny is a teacher." In this case, the model would need to enumerate 15 times to obtain the desired label sequence. This decoding process is time consuming, and the decoding time varies depending on the length of the input sequence. Consequently, while the template-based cue fine-tuning method is highly effective for small samples, it is not suitable for the NER task due to its inefficiency.

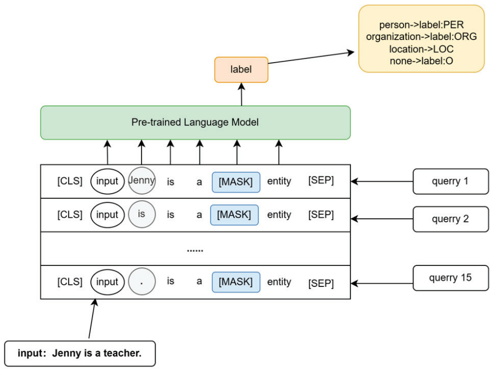
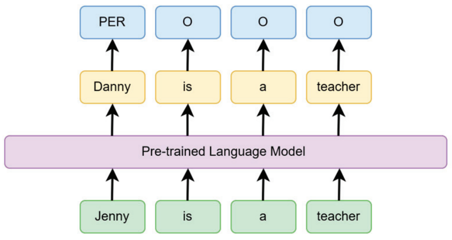

person->label:PER
organization->label:ORG
location->LOC
none->label:O
label
Pre-trained Language Model
[CLS] input Jenny is a [MASK] entity [SEP]
[CLS] input is is a [MASK] entity [SEP]
......
[CLS] input . is a [MASK] entity [SEP]
querry 1
querry 2
querry 15
input: Jenny is a teacher.

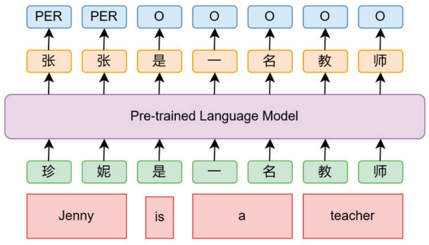
Figure 2. Example of NER's template-based prompt method.

2.2. *Chinese Entity-Oriented LM Fine Tuning*

The EntLM method introduces an innovative approach to template-free NER while leveraging the benefits of prompt tuning. Instead of using templates to guide the LM during fine tuning, this work proposes a novel objective called the entity-oriented LM goal, the flow is depicted in Figure 3a. During the prediction of the label sequence "Jenny is a teacher.", the LM is trained to predict "Jenny" as "Danny". Here, "Danny" serves as an indicator for the "PER" label, and indicators for other labels are determined using the label selection algorithm described below. Non-entity words are predicted as the original words. In this paper, we focus on Chinese entity-oriented LM fine tuning, and the process is illustrated in Figure 3b. For an input sentence "珍妮是一名教师", which means "Jenny is a teacher", the target sentence is obtained by replacing the corresponding position of the entities in the input sentence with the label word, while the other positions remain unchanged. Here "张" represents the "PER" label. It is a very common surname in China. The corresponding position character remains unchanged to represent the "O" label.



(**a**) EntLM prediction label process



(**b**) Chinese EntLM prediction label process

Figure 3. Comparison of the flow of Chinese and English EntLM prediction labels.

2.3. *Label Word Generation*

For the labeled data, there are three methods of label word selection.

2.3.1. Searching with Data Distribution

To identify the labeled words in a specific class, we employed a frequency-based approach on the corpus. Specifically, we calculated the occurrence frequency of each word 'w' within Class C individually using $\phi(x = w, y^* = C)$. Then, we ranked the words based on their frequencies and selected the most frequently occurring words as the labeled words for that particular class.

$$M(C) = \underset{w}{argmax}\phi(x = w, y^* = C) \tag{2}$$

2.3.2. Searching Using LM Output Distribution

In this method, a pre-trained LM is employed to search for label words. First, each sample (X, Y^*) is input to LM to obtain the probability distribution $p(\hat{x}_i = w|X)$ of each word $w \in V$ (V is the vocabulary) at each position. Suppose $I_{top_k}(\hat{x}_i = w|X, Y^*) \rightarrow \{0, 1\}$ is used to determine whether w belongs to the top-k prediction of the sample. Finally, the labeled words of Class C can be obtained by Formula (3).

$$M(C) = \underset{w}{argmax}\sum_{i}^{|X|}\phi_{top\_k}\left(\overset{\wedge}{x_i} = w, y^* = c\right) \tag{3}$$

where $\phi_{top\_k}\left(\overset{\wedge}{x_i} = w, y^* = C\right) = I_{top\_k}(\overset{\wedge}{x_i} = w\Big|X, Y^*)\cdot I\left(y_i^* = C\right)$ denotes the frequency of w appearing in the top-k predictions of Class C.

2.3.3. Search Using Both Data and LM Output Distributions

Considering both data and LM output to select label words, the label words of Class C are obtained by the following equation.

$$M(C) = \underset{w}{argmax}\left\{\sum_{(X,Y^*)\in D}\sum_{i}^{|X|}\phi\left(\overset{\wedge}{x_i} = w, y_i^* = C\right)\cdot\sum_{(X,Y^*)\in D}\sum_{i}^{|X|}\phi\left(\overset{\wedge}{x_i} = w, y_i^* = C\right)\right\} \tag{4}$$

2.3.4. Removing Conflicting Label Words

Because the high-frequency label words selected during the label word selection process may overlap across different categories, using them directly in the final training may lead to conflicts. To address this issue, a step is taken to remove conflicting label words for Class C. Th is the manually set threshold value. w is used as a label for category C only if the ratio of the frequency of the word w in category C to the frequency of the w in all categories is greater than Th. The specific procedure for removing conflicting label words is as follows:

$$w = M(C), if \frac{\phi(x = w, y^* = C)}{\sum_k \phi(x = w, y^* = k)} > Th \tag{5}$$

In order to utilize more information about the label words, a virtual label word approach can be used, where the virtual label word for each class uses the average vector of high frequency words for each class as a prototype.

2.4. CEntLM-KL Model

For a given input sentence $X = \{x_1, x_2, \ldots, x_n\}$ and its corresponding label sequence $Y = \{y_1, y_2, \ldots, y_n\}$, after the input sequence is processed by the fine-tuned BERT, the model utilizes two decoding methods: Softmax and CRF. The final target sentence is obtained by replacing the entity positions in the input sentence X with the label words, while keeping the tokens at the other positions unchanged, i.e., $X = \{x_1, x_2, \ldots, M(y_i), \ldots, x_n\}$. The overall architecture of the model is depicted in Figure 4, which illustrates the different components and their connections within the model. The BERT is the pre-trained Chinese-BERT.

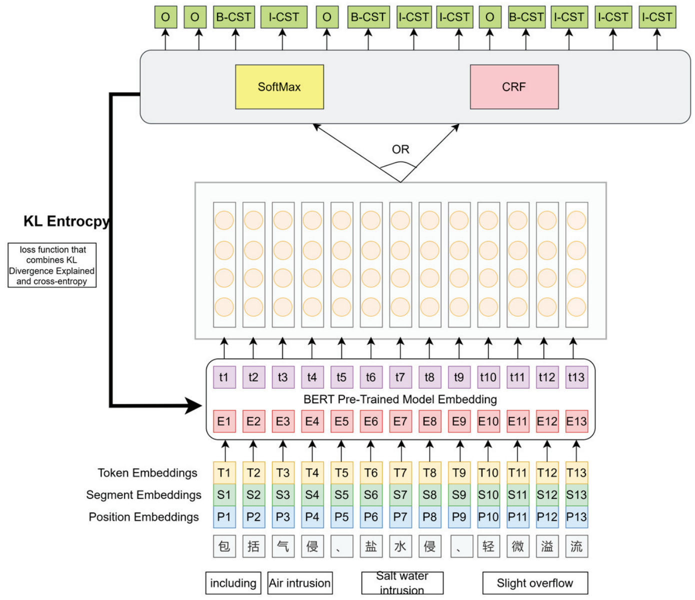

O
O
B-CST
I-CST
O
B-CST
I-CST
I-CST
O
B-CST
I-CST
I-CST
I-CST
SoftMax
CRF
OR
KL Entrocpy
loss function that combines KL Divergence Explained and cross-entropy
t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13
BERT Pre-Trained Model Embedding
E1 E2 E3 E4 E5 E6 E7 E8 E9 E10 E11 E12 E13
Token Embeddings
Segment Embeddings
Position Embeddings
T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11 T12 T13
S1 S2 S3 S4 S5 S6 S7 S8 S9 S10 S11 S12 S13
P1 P2 P3 P4 P5 P6 P7 P8 P9 P10 P11 P12 P13
包 括 气 侵 、 盐 水 侵 、 轻 微 溢 流
including
Air intrusion
Salt water intrusion
Slight overflow

Figure 4. Overall architecture of CEntLM-KL.

In general, the loss function for the task of NER is in the form of cross-entropy, as shown in Formula (7). In order to make the probability distribution obtained from the model prediction closer to the distribution of the real results, we choose to use a loss function that combines Kullback—Leibler Divergence Explained and cross-entropy and we named it the KL entropy. So, the loss function of the model is set as Formula (6).

$$L_{CEntLM-KL} = e^{-L_{EntLM}} \left(L_{EntLM} + a * log(1 - \left(e^{-L_{EntLM}} \right)^{\frac{1}{a}} \right) \right) \tag{6}$$

$$L_{EntLM} = -\sum_{i=1}^{n} logP\left(x_i = x_i^{Ent} \middle| X\right) \tag{7}$$

where a is a hyperparameter, which is set to 1944 in this model. $p\left(x_i = x_i^{Ent}\middle|X\right) = Softmax(W_{lm}.h_i)$. W_{lm} are also the parameters of the pre-trained LM head. No new parameters are introduced in the proposed approach and it avoids the need for complex template construction in the NER task. This allows us to maintain the capability of handling tasks with limited labeled samples, as observed in cue-based approaches. During testing, the test input X is fed directly into the model, and the probability of labeling characters with Class $y \in Y$ is modeled as Formula (8).

$$p(y_i = y|X) = p(x_i = M(y)|x) \tag{8}$$

Only one decoding process is needed to obtain all tokens for each sentence, which is more efficient than the template-based prompt task.

2.5. Construction of the Knowledge Graph Schema Layer

Before constructing the knowledge graph, it is crucial to understand the specific requirements of the drilling well control domain. Through careful analysis and investigation, a comprehensive list of 14 major categories of entities and 11 categories of relationships is identified. These entities and relationships are presented in Table 1 and Figure 5, respectively, providing a clear overview of the domain-specific entities and their interconnections. Based on these relationships, it can be seen that there are more relationships related to the types of wells (WTY) and complex working conditions in the drilling (CST), indicating that WTY and CST are two very important entity types in the process of MPD.

Table 1. Types of entities.

Entity	Abbreviation	Example
Complex working conditions in the drilling	CST	Overflow, well leak
Diagnosis methods for abnormal drilling conditions	DAM	Human judgment, expert system
Drilling fluid type	DFT	Water-in-oil emulsion drilling fluid
Method of drilling a well	DLM	Underbalanced drilling
Stages of the drilling process	DST	Drilling, well cementing
Number of drillings starts during drilling	NOS	First drilling, second drilling
Abnormal conditions processing measures	PMC	Forced drilling start, reduce stress agitation
Types of pumps	PTP	Filling pumps
Drilling fluid rheological model	RMD	Power law, Newton
Reasons for complex conditions	ROC	Personnel errors, well wall instability
Treatment for abnormal conditions	TOC	Plugging of leaks, stop drilling and circulation
The shape of the well	WSH	Horizontal wells, straight wells
Types of wells	WTY	Development Wells, evaluation wells
Phenomena of abnormal conditions	POC	Pump pressure rising, torque increases

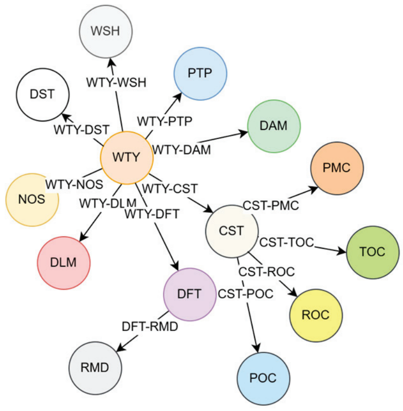


Figure 5. Various types of relationships between entities.

2.6. *Entity Extraction Based on Fusion Model*

This paper proposes an entity extraction framework that combines the Trie tree method and the CEntLM-KL model. The framework, depicted in Figure 6, involves two rounds of entity extraction.

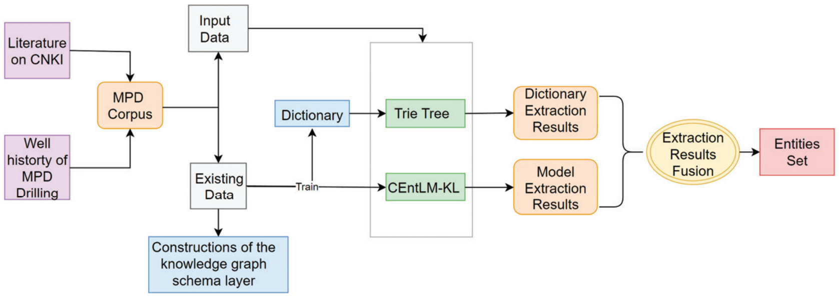

Literature on CNKI
Input Data
MPD Corpus
Well historty of MPD Drilling
Dictionary
Trie Tree
Dictionary Extraction Results
Extraction Results Fusion
Entities Set
Existing Data
Train
CEntLM-KL
Model Extraction Results
Constructions of the knowledge graph schema layer

Figure 6. Fusion entity extraction process.

In the first round, Trie trees are constructed based on a manually constructed dictionary derived from existing data. Figure 7 provides an example of a Trie tree. This dictionary primarily captures the known entities in the domain. Once the Trie tree is obtained, the next step is to extract entities and annotate the data using the BIO scheme. For documents such as literature, the following steps are performed after sentence segmentation.

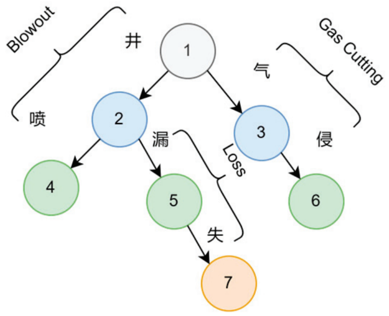


Figure 7. Trie tree example.

Candidate Word Generation: Algorithm 1 is executed to generate candidate words from the input text. This algorithm identifies potential entities by considering different combinations of consecutive words within each sentence.

Trie Tree Matching: The generated candidate words are then matched against the Trie tree. If a candidate word successfully matches an entity in the Trie tree, it is annotated with the corresponding BI (Beginning or Inside tag) to indicate the entity boundary. If a candidate word does not match any entity in the Trie tree, it is annotated with the O tag (Outside) to indicate that it is not part of any entity.

By annotating the data using the BIO scheme, each word in the input text is assigned a specific tag indicating its role in the entity extraction task. This annotation process helps in training and evaluating the performance of entity extraction models. The Trie tree method is then applied to extract entities from the input data and to generate a dataset for the subsequent deep learning model.

The second round of entity extraction utilizes a trained deep learning model. The model is trained on the dataset obtained from the first round and focuses on extracting potential entities that were not discovered by the Trie tree method. The extraction results from this round are subsequently merged with the first round's results to obtain a comprehensive entity set.

Algorithm 1: Find Entity

```
Input: root: Node, sentence: String
Output: entities: List
1:  curNode ← root; index ← 0; maxLength ← 10; entities ← [ ]
2:  while index < len(sentence) do:
3:      j ← maxLength
4:      while j ! = 0 then
5:          word ← sentence [index: index + j]
6:          if SeachWord(root, word) = True then
7:              index ← index + j − 1; entities.add(word);
8:          end if
9:          j ← j − 1
10:     end while
11:     index ← index + 1
12: end while
13: return entities
```

2.7. Rule-Based Relationship Extraction

In this paper, a rule-based method is employed for relationship extraction. The method leverages the analysis of a significant amount of text data to identify patterns and expressions commonly used to indicate relationships. Based on this analysis, matching rules are formulated to extract relationships from the text. The main methods for rule formulation are as follows.

1. Entities Word-Based Extraction: This method focuses on identifying relationships based on the subject words involved. By examining the subject words and their corresponding contexts, relationships between entities can be inferred. Matching rules are defined to capture specific patterns and linguistic cues indicating relationships.
2. Keyword-Based Extraction: This method relies on the presence of certain keywords or key phrases that frequently co-occur with specific relationships. By identifying these keywords and using them as indicators, relationships can be extracted. Matching rules are designed to detect the occurrence of these keywords in the text and associate them with the appropriate relationships.

In this paper, two methods, namely Entities Word-Based Extraction and Keyword-Based Extraction, are selected for relationship extraction. Table 2 shows the types of relationships based on entity word extraction and the entity words required to extract the relationships. Table 3 shows the types of relationships based on keyword extraction and the keywords required for extracting the relationships.

Table 2. Relationships based on entity word extraction.

Entity Word 1	Entity Word 2	Relationship
WTY	WSH	WTY-WSH
WTY	DST	WTY-DST
WTY	CST	WTY-CST
WTY	NOS	WTY-NOS
WTY	DAM	WTY-DAM
WTY	DLM	WTY-DLM
DFT	RMD	DFT-RMD

Table 3. Relationships based on keyword extraction.

Keyword	Relationship
‘特征’ (Features)	CST-POC
‘发生’ (Happen)	CST-POC
‘发现’ (Discover)	CST-POC
‘由于’ (Due to)	CST-ROC
‘造成’ (Causing)	CST-ROC
‘引起’ (Cause)	CST-ROC
‘采取’ (Take)	CST-TOC
‘应’ (Should)	CST-TOC
‘提高’ (Increase)	CST-TOC
‘减小’ (Decrease)	CST-TOC
‘进行’ (Performing)	CST-TOC
‘采用’ (Adopted)	CST-TOC
‘采用’ (Adopted)	CST-PMC
‘采取’ (Take)	CST-PMC
‘应’ (Should)	CST-PMC
‘提高’ (Increase)	CST-PMC
‘减小’ (Decrease)	CST-PMC
‘进行’ (Performing)	CST-PMS

3. Experiment and Results

3.1. Datasets

There are three labeled datasets for NER, all using BIO for labeling. Figure 8 shows an example of BIO annotation for ‘海钓比赛地点在厦门和金门之间’ which means ‘The sea fishing competition is between Xiamen and Jinmen.’ The first one is the People’s Daily-NER Dataset. This NER dataset contains 3 common entity types of Person Name (PER), Place Name (LOC), and Organization Name (ORG). The second one is the Few-shot-NER Dataset. A 10-shot dataset extracted manually from the above People’s Daily-NER dataset. The numbers of entities in this dataset labeled as PER, LOC, and ORG types are all 10. The final dataset is the MPD Dataset which is a domain knowledge dataset that was established by collecting 150+ documents obtained by crawling the Internet and CNKI related materials, as well as collating our own, relevant, 20+ MPD well history materials. The dataset contains 14 types of entities as shown in Table 1 above. The composition of the datasets is shown in Table 4.

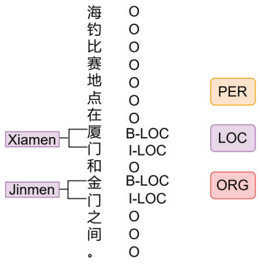


Figure 8. Example of BIO labeling.

Table 4. Composition of the People's Daily-NER Dataset.

Datasets	Type	Number
People's Daily-NER Dataset	PER	1W+
	LOC	2W+
	ORG	1W+
MPD Dataset	CST	780
	DAM	7
	DFT	13
	DLM	5
	DST	480
	NOS	92
	PMC	51
	PTP	48
	RMD	5
	ROC	62
	TOC	278
	WSH	12
	WTY	10
	POC	198

3.2. Comparison Experiment

Experiments were conducted using the three datasets described above.

To establish a baseline for comparison, four baseline models were selected. These baseline models represent existing approaches or techniques in the field of entity extraction and relationship extraction. The results obtained by these baseline models are presented in Table 5.

Table 5. F-1 scores of different models on three datasets.

Methods	People's Daily-NER Dataset	Few-Shot-NER Dataset	MPD Dataset
Two-tower	0.9102	0.2893	0.2400
BERT-CRF	0.9125	0.2015	0.1825
TemplateNER	0.8942	0.2945	0.2545
EntLM	0.9041	0.3373	0.2999
EntLM+CRF	0.9039	0.3542	0.3022
CEntLM-KL(ours)	0.9040	0.3935	0.3215
CEntLM-KL+CRF (ours)	**0.9127**	**0.4100**	**0.3342**

Table 5 showcases the comparative performance of the proposed model against the baseline models.

The results presented in Table 5 serve to highlight the strengths and improvements of the proposed model in comparison to existing approaches. They provide quantitative evidence of the model's effectiveness and its potential contributions in the context of the three datasets described above.

Two-tower model The two-tower model [20] used two BERT Encoders, one Encoder encodes the representation of each token, and the other Encoder encodes the natural language form of the BIO tag of the label (or describes the label with other text) to obtain the label representation, and then finds the similarity between each token in the text to be predicted. The similarity between each token and all the label representations in the text to be predicted is then found, and the label with the highest similarity is found.

BERT-CRF The BERT-CRF end-to-end deep learning model does not require handmotion features, and the word embedding is obtained by pre-training of BERT + fine-tune. The CRF layer only borrows the concept of transfer matrix from traditional CRF, which is completely different from traditional CRF.

TemplateNER TemplateNER [2] a template-based prompt method. By constructing a template for each class, it allows querying of each span of each class separately. The score of each query is obtained by generating a pre-trained LM and BART calculates the generalization probability of the query statement.

EntLM EntLM [14] discards the template and uses NER as a language modeling task, with the location of entities predicted as label word, and the non-entity location predicted as the original word, which is faster.

The comparison of the results between the models in this paper and the baseline models clearly demonstrates the superior performance of the proposed model, particularly when combined with the CRF decoder. The results show that the model in this paper outperforms the baseline models on both small and multiple sample datasets.

Furthermore, the stability of the performance of the proposed model on multiple sample datasets indicates its robustness and reliability. It demonstrates that the model is able to consistently achieve high performance across different datasets, suggesting its capability to generalize well and handle diverse data distributions.

Overall, the results clearly demonstrate that the model in this paper, especially when combined with the CRF decoder, exhibits superior performance on both small and multiple sample datasets. This highlights the effectiveness and stability of the proposed model for entity recognition tasks, further reinforcing its value and potential for practical applications. It can be found that the models with the CRF decoder added perform better than the model without it. We speculate that this is because our model experimented on Chinese datasets, which will have more cases of multiple tokens representing the same word compared to English data, but the model focuses more on the label of each token, and CRF alleviates this problem by focusing on the connection between tokens.

Meanwhile, the F-1 values for different types of entities are shown in Figure 9. We can find that CST, DST, NOS, WTY, WSH-type entities have higher F-1 values. For the CST-, DST-type entities, there are more types and numbers of entities, so the training effect is better. However, for the same type and number of DST-type entities, the performance is not good because the DST-type entities are generally longer and there is no way to achieve a good result with a small amount of training data. For NOS, WTY, and WSH-type data, there are fewer entity types, and all entities are covered in the training data, so the training results are better. However, for PTP-type entities, although the entity types are also relatively few, the types involved in the training dataset are limited, so that the F-1 value is relatively low.

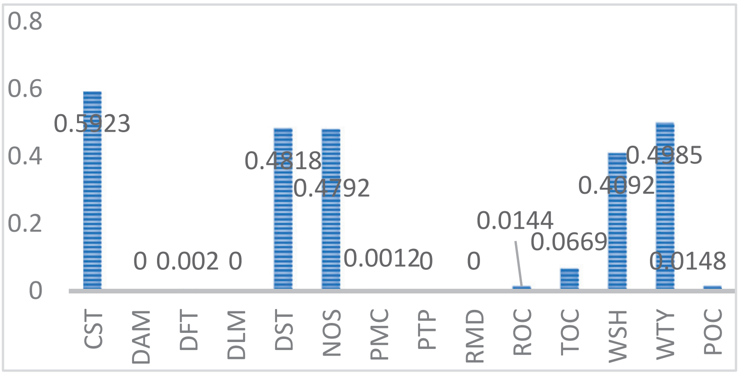


Figure 9. F-1 values for different types of entities.

In addition, we also compare the entity extraction results of CEntLM and CEntLM-KL, as shown in Figure 10. The comparison results in Figure 10 show that the number of entities extracted using the CEntLM-KL model is more than the number of entities extracted using the CEntLM model. By comparing the experimental results of the two models, we further proved the effectiveness of the CEntLM-KL model in practical applications.

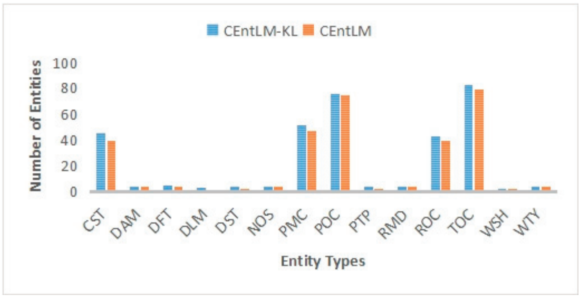


Figure 10. Comparison of Entity Extraction Results between CEntLM and CEntLM-KL Models.

3.3. MPD Knowledge Graph Construction

The data used to construct the knowledge graph in this study were obtained through a comprehensive collection and organization process. Various sources were utilized, including relevant literature on the Internet, information from reputable websites, and drilling data. The data are unlabeled data that are different from the training data. The aim was to collect a comprehensive set of data related to the drilling control domain.

Considering the limited availability of data, a small-scale knowledge graph was created by extracting entities and relationships. This knowledge graph includes 332 entities and 507 relationships, these are shown in Figures 10 and 11. The entities represent the various elements and concepts in the drilling control domain, while the relationships are the connections and associations between these entities.

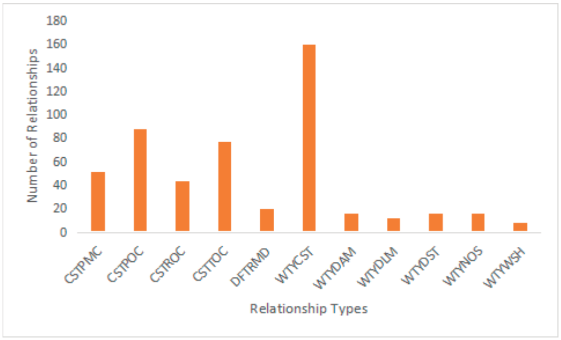


Figure 11. Number of Relationships.

The construction of the knowledge graph involved a meticulous process of data extraction, integration, and organization. The collected data were carefully analyzed and structured to ensure the accuracy and consistency of the knowledge map. The resulting knowledge map becomes a valuable resource for understanding and exploring the field of drilling control.

Firstly, entities are extracted from the data using an entity extraction framework that combines the Trie tree method and the CEntLM-KL model. Then, rule-based methods are used for relationship extraction, with two main rules developed: Entities Word-Based Extraction and Keyword-Based Extraction. After extracting entities and relationships, the MPD knowledge graph was stored and visualized in Neo4J.

Although the knowledge graph in this study is relatively small in scale, it still provides a solid foundation for further research and analysis. It can be used as a reference and starting point for future knowledge map expansion and refinement. In addition, the insights and findings derived from this small knowledge map can contribute to a deeper understanding of the drilling control field and facilitate decision-making in related industries.

3.4. *Knowledge Graph Reasoning*

The knowledge graph in the MPD domain can give possible reasons or solutions through knowledge graph reasoning when complex working conditions occur. Therefore, it is of great practical significance to construct the MPD domain knowledge graph and use it for reasoning. Timely diagnosis and prompt handling of complex working conditions can greatly reduce losses. As a result, the reasoning of MPD knowledge graph is of great significance for handling of complex working conditions. It is crucial to understand the phenomenon, cause, and treatment measures of a complex working condition. In this paper, we provide a one-hop reasoning example of the possible phenomenon of "dry drilling sticking" in reasoning as shown in Figure 12. Besides, promptly diagnosing of complex operating conditions and conducting well control are very important when an abnormal phenomenon occurs. A multi-hop reasoning example is provided in this paper for obtaining measures for dealing with "pump pressure rising" as shown in Figure 13.

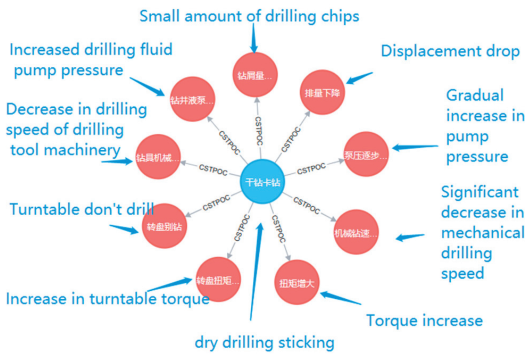

Small amount of drilling chips
Increased drilling fluid pump pressure
Displacement drop
Decrease in drilling speed of drilling tool machinery
Gradual increase in pump pressure
Turntable don't drill
Significant decrease in mechanical drilling speed
Increase in turntable torque
Torque increase
dry drilling sticking
钻屑量...
钻井液泵...
排量下降
钻具机械...
泵压逐步...
干钻卡钻
转盘别钻
机械钻速...
转盘扭矩...
扭矩增大
CSTPOC

Figure 12. One-hop Neo4J query results.

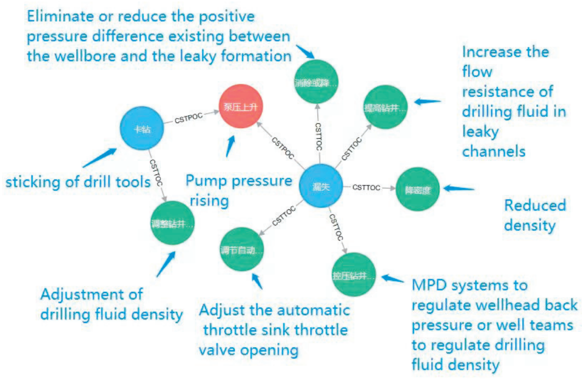

Eliminate or reduce the positive pressure difference existing between the wellbore and the leaky formation
Increase the flow resistance of drilling fluid in leaky channels
sticking of drill tools
Pump pressure rising
Reduced density
Adjustment of drilling fluid density
Adjust the automatic throttle sink throttle valve opening
MPD systems to regulate wellhead back pressure or well teams to regulate drilling fluid density
卡钻
泵压上升
消除或降...
提高钻井...
漏失
降密度
调整钻井...
调节自动...
控压钻井...
CSTPOC
CSTTOC

Figure 13. Multi-hop Neo4J query results.

3.4.1. One-Hop Reasoning

Figure 12 shows the results of a query using the Neo4j query language to retrieve all nodes that have a "CST-POC" relationship with a CST-type node named "dry drilling sticking" ("干钻卡钻"). Corresponding translations are shown in Table 6. Using the query command "MATCH (n:CST)-[r:CSTPOC]->(nn:POC) WHERE n.name = '干钻卡钻' RETURN nn" yields results that allow us to explore the potential outcomes or impacts when dry drill jams occur.

Table 6. The phenomenon of dry drill jamming.

Entity	Phenomenon
‘干钻卡钻’ (Dry drilling sticking)	‘钻屑量小’ (Small amount of drilling chips) ‘钻具机械钻速下降’ (Decrease in drilling speed of drilling tool machinery) ‘排量下降’ (Displacement drop) ‘泵压逐步上升’ (Gradual increase in pump pressure) ‘转盘别钻’ (Turntable don’t drill) ‘钻井液泵压增大’ (Increased drilling fluid pump pressure) ‘机械钻速下降明显’ (Significant decrease in mechanical drilling speed) ‘扭矩增大’ (Torque increase) ‘转盘扭矩增加’ (Increase in turntable torque)

The results shown in Table 4 describe the nodes associated with the “CST-POC” relationship and the identified anomalies (POC) associated with the CST node “dry drill jam”. These results provide insight into specific anomalies that may arise due to dry drill jamming, allowing researchers and domain experts to understand the potential challenges and impacts associated with this particular scenario.

Targeted exploration and analysis of the knowledge graph is possible using Neo4j’s query language, with the ability to retrieve specific nodes based on defined relationships and node attributes. This capability enhances the ability to discover valuable insights and patterns in the knowledge graph, facilitating informed decision making and proactive management of drilling operations.

In summary, the Neo4j query language provides a powerful tool for querying nodes that have a specific relationship to a given node. This knowledge in the context of this example query helps to understand the potential consequences and impacts of the queried events and supports the development of effective strategies to address and mitigate these challenges in drilling operations.

3.4.2. Multi-Hop Reasoning

Figure 13 shows the results of retrieving all nodes that have a “CST-POC” relationship with a node of type POC named “泵压上升” (pump pressure rising) and the results of nodes with “CST-TOC” relationship with the results obtained from the previous query are further queried. using the Neo4j query language. The corresponding translation results are shown in Table 7. In other words, we query the abnormal operating conditions that may occur when the pump pressure rises and the measures to deal with them. Using the query command “match (na:CST)-[re:CSTPOC]->(nb:POC) where nb.name = ‘泵压上升’ WITH na,re,nb match (na:CST)-[re2:CSTTOC]->(nc:TOC) return na,re,nb,re2,nc“, the results obtained allow us to explore what should be done to avoid losses to the greatest extent possible when there is a rise in pump pressure.

Table 7. Measures to deal with the abnormal operating conditions that may occur.

Phenomenon	One-Hop Abnormal Conditions	Two-Hop Measure
‘泵压上升’ (Pump pressure rising)	‘卡钻’ (sticking of drill tools)	‘调整钻井液密度’ (Adjustment of drilling fluid density)
	‘漏失’ (Loss)	‘消除或降低井筒与漏层之间存在的正压差’ (Eliminate or reduce the positive pressure difference existing between the wellbore and the leaky formation)
		‘提高钻井液在漏失通道中的流通阻力’ (Increase the flow resistance of drilling fluid in leaky channels)
		‘降密度’ (Reduced density)
		‘控压钻井系统调节井口回压或井队调节钻井液密度’ (MPD systems to regulate wellhead back pressure or well teams to regulate drilling fluid density)
		‘调节自动节流管汇节流阀开度’ (Adjust the automatic throttle sink throttle valve opening)

3.5. Further Analysis of Experimental Results

According to Table 4 and Figure 9, we can find that the number of various entities in the MPD Dataset are limited, mainly due to the fact that the MPD Dataset used in this paper is built manually, mainly coming from some drilling data and Internet data, and some types of entities are indeed relatively small in number, such as NOS and RMD, etc. The collected content is currently relatively small, and future improvements to the database will also be considered.

Figure 9 shows the F-1 values for different types of entities. It can be observed that there is a significant difference in F-1 values among different entities, and some entities have very low F-1 values. Through analysis, we believe that there are two main reasons.

(1) The number of entities of this type in the training set is too small, and the number of occurrences is too small to achieve good training results. For example, DAM, DFT, DLM, etc.
(2) This type of entity is generally longer and has poor training effectiveness. For example, ROC, POC, etc. For example, the ROC-type entity" 井内泥浆静止时间过长, 触变性很大, 下钻时又不分段循环, 破坏泥浆的结构", which means" The mud in the well is stationary for too long and thixotropic, and is not circulated in sections when drilling, which destroys the structure of the mud".

4. Conclusions

MPD is the most effective means to ensure safe drilling, and MPD can avoid further deterioration of complex conditions through fine control of wellhead back pressure. The key to successful MPD is the control strategy, but the current well control strategy relies strongly on manual experience, which hinders the automation and intelligence of well control. To address this problem, this paper constructs an MPD knowledge graph, which extracts knowledge from published papers and drilling reports to guide well control. To improve the performance of entity extraction in this knowledge graph, this paper extends the EntLM model to a few-shot Chinese entity recognition model CEntLM-KL and constructs KL entropy to enhance the entity recognition accuracy. By experimenting on the standard dataset, it is shown that the method proposed in this paper has significantly improved compared with the SOTA method, and the entity recognition F-1 value reaches 33% on the drill-down dataset with small samples. Finally, this paper stores the knowledge graph into Neo4J and performs knowledge inference applications. Due to the manual construction of the domain database used in this article, the data source and volume still need to be improved. In the future, we will consider improving the MPD dataset. At present, the source of MPD knowledge is mainly the Internet and CNKI related literature as well as relevant owned MPD data, and the fusion of knowledge graph has not been considered. Our next step will be to consider fusing of the relevant knowledge of known graphs.

Author Contributions: Conceptualization, X.W. and J.F.; Data curation, X.L. and J.F.; Funding acquisition, Y.L. and X.H.; Methodology, S.W., Y.L., X.L. and X.H.; Project administration, Y.L. and X.W.; Software, S.W.; Supervision, X.H.; Validation, X.L. and J.F.; Writing—original draft, S.W.; Writing—review & editing, X.H. All authors have read and agreed to the published version of the manuscript.

Funding: The authors are grateful for the support of the National Natural Science Foundation of China Major Program No. 2021ZD0112500, the National Natural Science Foundation of China No. 61972174 and No. 62172187, the Science and Technology Planning Project of Jilin Province No. 20220201145GX, No. 20200708112YY and No. 20220601112FG, Guangdong Universities' Innovation Team Project No. 2021KCXTD015, Guangdong Key Disciplines Project No. 2021ZDJS138, and Projects of CNPC No. 2021DQ0503 and No. 2020B-4019.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Dai, Z.; Wang, X.; Ni, P.; Li, Y.; Li, G.; Bai, X. Named entity recognition using BERT BiLSTM CRF for Chinese electronic health records. In Proceedings of the 2019 12th International Congress on Image and Signal Processing, Biomedical Engineering and Informatics (CISP-BMEI), Suzhou, China, 19–21 October 2019; IEEE: New York, NY, USA, 2019; pp. 1–5.
2. Cui, L.; Wu, Y.; Liu, J.; Yang, S.; Zhang, Y. Template-Based Named Entity Recognition Using BART. In Proceedings of the Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, Online, 1–6 August 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 1835–1845.
3. Fritzler, A.; Logacheva, V.; Kretov, M. Few-shot classification in named entity recognition task. In Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Limassol, Cyprus, 8–12 April 2019; pp. 993–1000.
4. Li, J.; Chiu, B.; Feng, S.; Wang, H. Few-Shot Named Entity Recognition via Meta-Learning. *IEEE Trans. Knowl. Data Eng.* **2022**, *34*, 4245–4256. [CrossRef]
5. Ma, T.; Jiang, H.; Wu, Q.; Zhao, T.; Lin, C.-Y. Decomposed Meta-Learning for Few-Shot Named Entity Recognition. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, 22–27 May 2022; Association for Computational Linguistics: Stroudsburg, PA, USA, 2022; pp. 1584–1596.
6. Chen, J.; Liu, Q.; Lin, H.; Han, X.; Sun, L. Few-shot Named Entity Recognition with Self-describing Networks. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, 22–27 May 2022; Association for Computational Linguistics: Stroudsburg, PA, USA, 2022; pp. 5711–5722.
7. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
8. Schick, T.; Schütze, H. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, Online, 19–23 April 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021. Main Volume. pp. 255–269.
9. Schick, T.; Schütze, H. It's Not Just Size That Matters: Small Language Models Are Also Few-Shot Learners. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online, 6–11 June 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 2339–2352.
10. Shin, T.; Razeghi, Y.; Logan, R.L., IV; Wallace, E.; Singh, S. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 4222–4235.
11. Gao, T.; Fisch, A.; Chen, D. Making Pre-trained Language Models Better Few-shot Learners. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Online, 1–6 August 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 3816–3830.
12. Liu, X.; Zheng, Y.; Du, Z.; Ding, M.; Qian, Y.; Yang, Z.; Tang, J. GPT understands, too. *arXiv* **2021**, *preprint*. arXiv:2103.10385.
13. Quimbaya, A.P.; Múnera, A.S.; Rivera, R.A.G.; Rodríguez, J.C.D.; Velandia, O.M.M.; Peña, A.A.G.; Labbé, C. Named entity recognition over electronic health records through a combined dictionary-based approach. *Procedia Comput. Sci.* **2016**, *100*, 55–61. [CrossRef]
14. Ma, R.; Zhou, X.; Gui, T.; Tan, Y.; Li, L.; Zhang, Q.; Huang, X. Template-free prompt tuning for few-shot NER. *arXiv* **2021**, *preprint*. arXiv:2109.13532.
15. Wiseman, S.; Shieber, S.; Rush, A. Learning Neural Templates for Text Generation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; Association for Computational Linguistics: Stroudsburg, PA, USA, 2018; pp. 3174–3187.
16. Wang, D.; Chen, Z.; He, W.; Zhong, L.; Tao, Y.; Yang, M. A Template-guided Hybrid Pointer Network for Knowledge-based Task-oriented Dialogue Systems. In Proceedings of the 1st Workshop on Document-Grounded Dialogue and Conversational Question Answering (DialDoc 2021), Online, 5 August 2021; Association for Computational Linguistics: Stroudsburg, PA, USA, 2021; pp. 18–28.
17. Gomes, J., Jr.; de Mello, R.C.; Ströele, V.; de Souza, J.F. A hereditary attentive template-based approach for complex Knowledge Base Question Answering systems. *Expert Syst. Appl.* **2022**, *205*, 117725. [CrossRef]
18. Wang, D.; Liang, Y.; Xu, D.; Feng, X.; Guan, R. A content-based recommender system for computer science publications. *Knowl.-Based Syst.* **2018**, *157*, 1–9. [CrossRef]
19. Guan, R.; Zhang, H.; Liang, Y.; Giunchiglia, F.; Huang, L.; Feng, X. Deep feature-based text clustering and its explanation. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 3669–3680. [CrossRef]
20. Ma, J.; Ballesteros, M.; Doss, S.; Anubhai, R.; Mallya, S.; Al-Onaizan, Y.; Roth, D. Label Semantics for Few Shot Named Entity Recognition. In Proceedings of the Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, 22–27 May 2022; Association for Computational Linguistics: Stroudsburg, PA, USA, 2022; pp. 1956–1971.

Article

IBGJO: Improved Binary Golden Jackal Optimization with Chaotic Tent Map and Cosine Similarity for Feature Selection

Kunpeng Zhang [1], Yanheng Liu [1,2], Fang Mei [1,2,*], Geng Sun [1,2] and Jingyi Jin [1]

1 College of Computer Science and Technology, Jilin University, Changchun 130012, China; zhangkp18@mails.jlu.edu.cn (K.Z.); yhliu@jlu.edu.cn (Y.L.); sungeng@jlu.edu.cn (G.S.); jyjin20@mails.jlu.edu.cn (J.J.)
2 Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Jilin University, Changchun 130012, China
* Correspondence: meifang@jlu.edu.cn

Abstract: Feature selection is a crucial process in machine learning and data mining that identifies the most pertinent and valuable features in a dataset. It enhances the efficacy and precision of predictive models by efficiently reducing the number of features. This reduction improves classification accuracy, lessens the computational burden, and enhances overall performance. This study proposes the improved binary golden jackal optimization (IBGJO) algorithm, an extension of the conventional golden jackal optimization (GJO) algorithm. IBGJO serves as a search strategy for wrapper-based feature selection. It comprises three key factors: a population initialization process with a chaotic tent map (CTM) mechanism that enhances exploitation abilities and guarantees population diversity, an adaptive position update mechanism using cosine similarity to prevent premature convergence, and a binary mechanism well-suited for binary feature selection problems. We evaluated IBGJO on 28 classical datasets from the UC Irvine Machine Learning Repository. The results show that the CTM mechanism and the position update strategy based on cosine similarity proposed in IBGJO can significantly improve the Rate of convergence of the conventional GJO algorithm, and the accuracy is also significantly better than other algorithms. Additionally, we evaluate the effectiveness and performance of the enhanced factors. Our empirical results show that the proposed CTM mechanism and the position update strategy based on cosine similarity can help the conventional GJO algorithm converge faster.

Keywords: feature selection; machine learning; classification; chaotic; cosine similarity; golden jackal optimization

Citation: Zhang, K.; Liu, Y.; Mei, F.; Sun, G.; Jin, J. IBGJO: Improved Binary Golden Jackal Optimization with Chaotic Tent Map and Cosine Similarity for Feature Selection. *Entropy* **2023**, *25*, 1128. https://doi.org/10.3390/e25081128

Academic Editor: Gholamreza Anbarjafari

Received: 16 June 2023
Revised: 23 July 2023
Accepted: 26 July 2023
Published: 27 July 2023

1. Introduction

Machine learning and data mining have expanded in many fields, including active matter, molecular and materials science, nature language process (NLP) and biomedicine [1–3]. To create more complex machine learning models, many datasets with high-dimensional feature spaces are created [4,5]. However, as the dimensionality of data increases, there are more and more redundant features, and it becomes more difficult to train models with high generalization ability. Therefore, it is necessary to perform feature selection to solve these problems. Feature selection is a critical step in data mining and machine learning that involves identifying the most relevant and useful features within a dataset or set of characteristics. Predictive models can be more effective and precise by eliminating redundant or unnecessary features. This improves classification accuracy and helps algorithms generalize better to new data, prevent overfitting, and produce more accurate predictions. In addition to these benefits, feature selection can help uncover hidden relationships within the data and provide more insightful explanations for predictive models.

Unsupervised Feature Selection methods can be classified into three main approaches, similar to supervised and semi-supervised feature selection [6,7]. These approaches are determined by the feature selection strategy employed, including filter, wrapper, and embedded methods [8,9]. In particular, filter methods rank the features according to the calculated scores by using a statistical metric to assign each feature a meaningful score. However, they might use up more computational resources. In the wrapper method, the selection subset obtained by the selection algorithm is evaluated using a classifier, and feature selection is guided by the feedback of the classifier [10]. As a result, the accuracy of the wrapper method is greater than the filtering method, as indicated by [11]. Furthermore, feature selection is regarded as a component of the machine learning training phase, which makes embedding methods a particular case of packing methods [12,13].

Meanwhile, it is possible to think of feature selection as a global group optimization problem. In particular, some of the original dataset answers the optimization problem, which can be resolved using exhaustive and heuristic search techniques [14]. In contrast to heuristic search methods, exhaustive search methods typically have higher computational costs, especially for high-dimensional datasets [4]. Using meta-heuristic search techniques may be a more practical way to solve the feature selection problem [14]. Therefore, it is essential to choose features using efficient methods.

Evolutionary algorithms (EAs) have recently been used to solve feature selection challenges for the global search capacity of feature selection methods. Numerous researchers have used various population intelligence techniques to address feature selection issues, including the cuckoo search (CS) [15], genetic algorithm (GA) [11], particle swarm optimization (PSO) [12], whale optimization algorithm (WOA) [16], sparrow search algorithm (SSA) [17], harris hawks optimization (HHO) [18,19] and variants of these algorithms, for dealing with the feature selection problems. For instance, Hegazy et al. [20] attempt to enhance the basic SSA structure to increase the solution accuracy, reliability, and convergence speed. Additionally, Behrouz et al. propose an unsupervised probabilistic feature selection algorithm using ant colony optimization [21].

The golden jackal optimization (GJO) algorithm is one of the EAs, and research has demonstrated that it is both efficient and simple to apply [22]. Nevertheless, despite its extensive use, the conventional GJO algorithm may have certain drawbacks, such as insufficiently exploiting issue areas. In addition, the no free lunch (NFL) theory contends that no single algorithm is capable of solving every optimization problem [23]. The conventional GJO algorithm was created to solve continuous optimization issues. There might be better choices for feature selection tasks involving binary solution spaces. These circumstances drive us to improve the conventional GJO to make it better suited for feature selection tasks.

The main contributions of this paper are summarized as follows:

- We aim to simultaneously reduce the number of selected features and improve the classification accuracy. Specifically, we design a fitness function to achieve these optimization objectives jointly.
- We propose an improved binary golden jackal optimization algorithm (IBGJO) to solve the designed fitness function. First, IBGJO introduces a chaotic tent map to improve the exploitation capability of conventional GJO. Second, a new position-updating mechanism by cosine similarity is proposed to balance the exploitation and exploration capabilities of the algorithm. Finally, a binarization strategy is introduced to transfer the continuous solution space to the binary ones, making it suitable for dealing with feature selection solutions.
- We conduct various experiments to assess the performance of the proposed IBGJO with the comparative algorithms on 28 classical UC Irvine (UCI) Machine Learning Repository datasets in terms of average fitness value, average classification accuracy, average CPU running time and average number of selected features.

The rest of this paper is organized as follows. Section 2 gives a brief overview of the related work. Section 3 designs the formulated fitness function of feature selection. Section 4 gives the details of IBGJO. Section 5 presents the experimental results. Finally, Section 6 concludes this paper and suggests the future works.

2. Related Work

The significance of wrapper-based selection techniques in feature selection optimizations cannot be overlooked [24,25]. These methods operate on the premise of treating feature selection as a black box, and employ meta-heuristic algorithms and classifiers to obtain the optimal subset [26]. Numerous classical meta-heuristic algorithms have undergone modifications to tackle the feature selection problem, such as binary bat algorithm (BBA) [27], bare bones particle swarm optimization algorithm (BPSO) [12], binary gray wolf optimization algorithm (BGWO) [28], binary gravitational search algorithm (BGSA) [29], and so on.

In recent times, an increasing number of novel algorithms have been proposed to enhance the optimization of feature selection problems based on the wrapper approach, due to their vital significance. For instance, Al-Tashi et al. [30] examine binary optimization utilizing hybrid grey wolf optimization for feature selection in their paper. To resolve feature selection issues, a binary version of the hybrid grey wolf optimization (GWO) and PSO is suggested. In 2019 [31], binary variations of the butterfly optimization algorithm (BOA) are suggested and utilized to choose the best feature subset for classification purposes. A self-adaptive particle swarm optimization (SaPSO) approach is suggested by Xue et al., especially for large-scale feature selection [32]. The two-archive multi-objective artificial bee colony algorithm (TMABC-FS) is a multi-objective feature selection approach that Zhang et al. investigate to satisfy diverse decision-makers' criteria [33]. To increase the predictability of the hospitalization expense model, a novel method proposed based on the GA for feature selection and parameter optimization of the support vector machine (SVM) in 2019 [34]. Aimed at finding distinguishing characteristics across several class labels, Zhang et al. [35] offer an embedded multi-label feature selection approach with manifold regularization. To develop a more affordable computational model for voice analysis-based emotion categorization, Dey et al [36] offer a meta-heuristic feature selection (FS) method employing a hybrid of equilibrium optimization (EO) and golden ratio optimization (GRO) algorithms. Wang and Chen [37] propose an improved whale optimization algorithm (CMWOA) that integrates chaotic and multi-swarm techniques to accomplish parameter optimization and feature selection simultaneously for SVM in 2020. For feature selection issues in medical diagnosis, a hybrid crow search optimization method integrated with chaos theory and fuzzy c-means algorithm was proposed in 2020 [38].

Several leading-edge researchers have focused on GJO algorithms for optimizing feature selection. Initially designed to address continuous problems, GJO requires transfer functions to convert it into a binary form (BGJO) [39] that can effectively handle feature selection optimizations. While some studies have made strides in addressing feature selection challenges across a variety of contexts, it is important to note that the NFL [23] theorem holds that no method can solve every optimization problem. Furthermore, none of the aforementioned research has identified optimal subsets of variables across all datasets tested. Nonetheless, given the strong potential of conventional GJO in this area, our aim in this study is to incorporate several enhanced factors into conventional GJO with the objective of improving the efficiency of feature selection optimizations.

3. Problem Formulation

In this study, feature selection aims to minimize the number of chosen features while improving the classification accuracy, which can be defined as a multi-objective optimization problem [40]. To consider the two objectives of optimization, we constructed the following fitness function:

$$f_{Fitness} = \alpha \cdot E_r + \beta \cdot \frac{F_s}{F_a} \tag{1}$$

where F_s and F_a stand for the number of chosen features and the total number of features, respectively, and E_r is the classification error rate of a certain classifier. Additionally, the weights used to balance these two goals are α and β.

The formulated feature selection problem has a nonlinear discrete search space with numerous potential local minimum points. As a result, we suggest the binary IBGJO algorithm to address the feature selection problem.

4. Proposed Improved Golden Jackal Optimization Algorithm for Feature Selection

The following section provides a succinct overview of the conventional GJO algorithm and its key principles. Additionally, the conventional GJO algorithm is reviewed before delving into a comprehensive discussion of the proposed IBGJO algorithm's formulation.

4.1. Conventional Golden Jackal Optimization

The conventional GJO algorithm is inspired by the hunting behavior of golden jackal pairs and adopts a swarm-based approach [22]. Figure 1 shows the entire foraging process of the golden jackal pair. The whole foraging process includes searching for prey, tracking and surrounding of prey, attacking prey and capturing prey. This section delves into the mathematical modeling of the conventional GJO algorithm.

(**a**) Pair of golden jackals. (**b**) Golden jackals searching for prey.
(**c**) Tracking and surrounding of prey. (**d**) Launching an attack. (**e**) Capturing prey.

Figure 1. The stages of golden jackal pair hunting.

4.1.1. Search Space Formulation

The initial solution of the golden jackal optimization algorithm is also uniformly distributed on the search space, which is similar to other metaheuristic methods, and its distribution is as follows:

$$Y_0 = Y_{\min} + rand \times (ub - lb) \tag{2}$$

where Y_0 represents the initial randomized population, and ub and lb denote the upper and lower boundaries of the decision variables. Moreover, $rand$ is a random number that falls within the range of $[0, 1]$. The initialization procedure involves generating a foundational $Prey$ matrix, with the male and female jackals occupying the first and second positions, correspondingly. The composition of the $Prey$ is illustrated as follows:

$$Prey = \begin{bmatrix} Y_{1,1} & Y_{1,2} & \cdots & Y_{1,d} \\ Y_{2,1} & Y_{2,2} & \cdots & Y_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ Y_{n,1} & Y_{n,2} & \cdots & Y_{n,d} \end{bmatrix} \tag{3}$$

where Y_{ij} stands for the *i*-th prey's *j*-th dimension. There are *n* preys and *d* variables in total. The prey position can be regarded as an optimal solution. During optimization, an objective function is used to assess the fitness of each prey, with the resulting fitness values being compiled into a matrix:

$$F_{OA} = \begin{bmatrix} f(Y_{1,1}; Y_{1,2}; \cdots ; Y_{1,d}) \\ f(Y_{2,1}; Y_{2,2}; \cdots ; Y_{2,d}) \\ \vdots \\ f(Y_{n,1}; Y_{n,2}; \cdots ; Y_{n,d}) \end{bmatrix} \tag{4}$$

where f is the objective function, Y_{ij} displays the value of the *j*-th dimension of the *i*-th prey, and F_{OA} is the matrix for storing each prey's fitness. A male jackal (MJ) is the most suitable, and a female jackal (FMJ) is the second most suitable. The jackal couple finds the appropriate prey location.

4.1.2. Searching for Prey (Exploration Stage)

With their remarkable capability to detect and pursue prey, jackals can usually track down food successfully. Nevertheless, there are instances when their attempts fail, and the potential prey evades capture, prompting the jackals to give up and search for alternative sources of sustenance. During hunts, the MJ takes the lead, while the FMJ follows closely behind, and the mathematically modelled jackal pairs hunt as follows:

$$Y_1(t) = Y_M(t) - E.|Y_M(t) - rl.Prey(t)| \tag{5a}$$

$$Y_2(t) = Y_{FM}(t) - E.|Y_{FM}(t) - rl.Prey(t)| \tag{5b}$$

where t represents the current iteration, $Prey(t)$ is the vector indicating the prey's position. In contrast, $YM(t)$ and $YFM(t)$ are the positions of the MJ and FMJ, respectively. The revised positions of MJ and FMJ in relation to the prey are represented by $Y_1(t)$ and $Y_2(t)$. The prey's evasive energy E is computed as:

$$E = E_1 * E_0 \tag{6a}$$

$$E_0 = 2 * r - 1 \tag{6b}$$

$$E_1 = c_1 * (1 - (t/T)) \tag{6c}$$

E_0 depicts the beginning state of the prey's energy, while E_1 represents the prey's declining energy, where r is any random value between 0 and 1.

T stands for the max iteration number and c_1 is a constant value of 1.5. E_1 decreases linearly across iterations, from 1.5 to 0. In Equations (5a) and (5b), the distance between the jackal and the prey is calculated by $|Y(t) - rl.Prey(t)|$. Depending on how well the prey manages to evade the jackal, this distance is either added to or deducted from its present location. The vector of random numbers rl in Equations (5a) and (5b) represents the Lévy movement and is based on the Lévy flight. Prey is multiplied by rl to imitate Lévy-style prey movement, which is comparable to MPA [41] and is computed as follows:

$$rl = 0.05 * LF(y) \tag{7}$$

LF is the Lévy flight function, which is calculated as follows:

$$LF(y) = 0.01 \times (\mu \times \sigma) / \left(\left| v^{(1/\beta)} \right| \right); \sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times \left(2^{\frac{\beta-1}{2}}\right)} \right)^{1/\beta} \tag{8}$$

where β is constant set to 1.5 and u, v are random values inside of $(0, 1)$. The jackal positions are updated by averaging Equations (5a) and (5b), which results in the following:

$$Y(t+1) = \frac{Y_1(t) + Y_2(t)}{2} \tag{9}$$

4.1.3. Tracking and Pouncing the Prey (Exploitation Stage)

As prey are pursued by jackals, their evasive energy declines, leading to the eventual encirclement of the prey by a pair of jackals identified in an earlier phase. Once encircled, the prey is attacked and consumed by the jackals. The following mathematical model is a representation of the hunting behaviour of male and female jackals that hunt in pairs, which is as follows:

$$Y_1(t) = Y_M(t) - E.|rl.Y_M(t) - Prey(t)| \tag{10a}$$

$$Y_2(t) = Y_{FM}(t) - E.|rl.Y_{FM}(t) - Prey(t)| \tag{10b}$$

where $Prey(t)$ is the position vector of the prey during the current iteration t, and $Y_M(t)$ and $Y_FM(t)$ indicate the position of the MJ and FMJ. The updated MJ and FMJ positions in relation to the prey are represented by $Y_1(t)$ and $Y_2(t)$. Equation (6a) determines the prey's evading energy, or E. The jackal positions are updated in accordance with Equation (9).

The purpose of rl in Equations (10a) and (10b) is to allow for arbitrary behavior in the exploitation stage, favoring exploration and avoiding local optima. Equation (7) is used to determine rl. In the final iterations, this component aids in avoiding local optima sluggishness.

As a result of jackals moving closer to the prey, the factor can be carefully considered. Typically, natural obstacles stand in the way of jackals' proper and swift movement toward their prey. This is the goal of rl during the exploitation stage.

4.1.4. Switching from Exploration to Exploitation

The escape energy of the prey is utilized in the conventional GJO algorithm to transition from exploration to exploitation. Throughout avoiding behavior, prey energy significantly decreases. In light of this, Equation (6a) is used to represent the evasive energy. Every repetition, the initial energy E_0 deviates arbitrarily from the range of -1 to 1. The prey is physically waning when E_0 value decreases from 0 to -1, but when E_0 value increases from 0 to 1, it indicates an improvement in the strength of prey.

According to Figure 2, the altering avoiding energy E decreases over the iteration procedure. When $|E| > 1$, jackal partners hunt for prey that is exploring in different areas, and when $|E| < 1$, the jackal attacks the prey and engages in predation, as depicted in Figure 1.

To sum up, the conventional GJO search procedure starts with the random generation of a population of prey (possible solutions). MJ and FMJ hunting couples calculate the location of the prey during iterations. Each prospective member of the population updates their separation from the jackal pair. To emphasize exploration and exploitation, the E_1 parameter is decreased from 1.5 to 0, accordingly. When $E > 1$, the hunting pair of golden jackals strays from their victim, and when $E < 1$, it gathers at the prey. The conventional GJO algorithm is finally completed by satisfying an end criterion. Algorithm 1 presents the conventional GJO algorithm's pseudo-code.

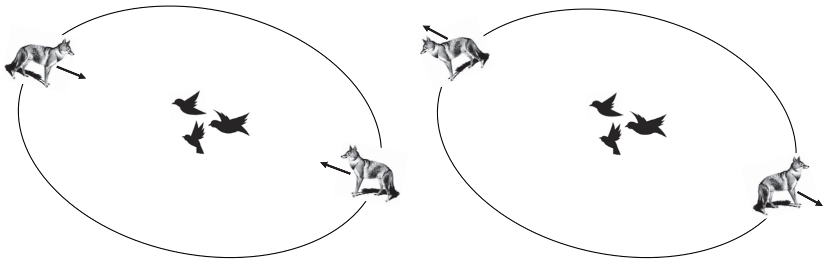

(**a**) $if(E < 1)$ (**b**) $if(E > 1)$

Figure 2. Attacking and searching for prey.

Algorithm 1 Conventional Golden Jackal Optimization

```
Require: The size of population $N_{pop}$, solution dimension $N_{dim}$, the max number of iterations $T_{max}$, lower and upper bounds $L_b$, $U_b$, the fitness function, the golden jackal $GJ$, $prey$, etc.
Ensure: The best solution found in the search process
 1: Initializing the population through random mechanism
 2: While $(t < T_{max})$
 3:   Calculate the fitness values of preys
 4:   $Y_1$ = best prey (Male Jackal position)
 5:   $Y_2$ = second best prey (Female Jackal Position)
 6:   for (each prey)
 7:     Update the evading energy $E$ using Equations (6a), (6b) and (6c)
 8:     Update $rl$ using Equations (7) and (8)
 9:     if $(|E| >= 1)$  // (Exploration phase)
10:       Update the prey position using Equations (5b), (5a) and (9)
11:     if $(|E| < 1)$  //(Exploitation phase)
12:       Update the prey position using Equations (10a), (10b) and (9)
13:   end for
14:   $t = t + 1$
15: end While
16: return $Y_1$
```

4.2. *The Proposed IBGJO*

This section introduces the enhancement factors in the proposed IBGJO algorithm, including the random population initialization strategy based on the Chaotic Tent map, the optimal location update mechanism based on cosine similarity, and the sigmoid function used to discretization the continuous solution space problem. Finally, the complexity of the IBGJO algorithm was analyzed.

4.2.1. Chaotic Tent Map for Initiate Population

In the conventional GJO algorithm, initial population information is generated randomly, which can pose difficulties in retaining population diversity and hinder the algorithm's effectiveness in achieving the optimal solution. In contrast, the chaotic tent map (CTM) mechanism is characterized by randomness, ergodicity, and regularity. It can be used either to generate the initial population or as a perturbation during the optimization process [37,42]. This approach overcomes the limitation of the algorithm becoming trapped in a suboptimal local solution, thereby improving its search efficiency compared to the original algorithm. The CTM mechanism is described as Algorithm 2.

Algorithm 2 Chaotic Tent Map (CTM) Mechanism

Define and initialize the related parameters: the size of population N_{pop}, solution dimension N_{dim}, chaotic tent map threshold a, low boundaries lb and up boundaries ub, respectively.

```
1:  For $i = 1$ to $N_{pop}$
2:      For $j = 1$ to $N_{dim}$
3:          If $rand < a$
4:              $x_{i,j} = \frac{rand}{a}$
5:          Else
6:              $x_{i,j} = a \cdot (1 - rand)$
7:      $x_i = lb + x_i \cdot (ub - lb)$
8:  return $mean$ of $x$
9:  For $i = 1$ to $N_{pop}$
10:     For $j = 1$ to $N_{dim}$
11:         If $x_{i,j} < mean$
12:             $x_{i,j} = 0$
13:         Else
14:             $x_{i,j} = 1$
15: return $x$
```

where a is the tent map's call threshold, generally set as 0.5. In IBGJO, we use a CTM as the initialization mechanism. Considering the different dimensions of datasets, we provide Hillvalley in 28 datasets as an example of population initialization. As shown in Figure 3, the number of golden jackals in the population is 20, and the dimension is 100. And in Figure 3, the points labelled as random population initialization are denoted in red, while the points labelled as CTM population initialization are represented in blue. As can be seen, compared with the random mechanism, the CTM mechanism has good distribution and randomness. Therefore, the initialized population is more evenly distributed in the search space, which is more conducive to the algorithm's optimization efficiency and solution accuracy.

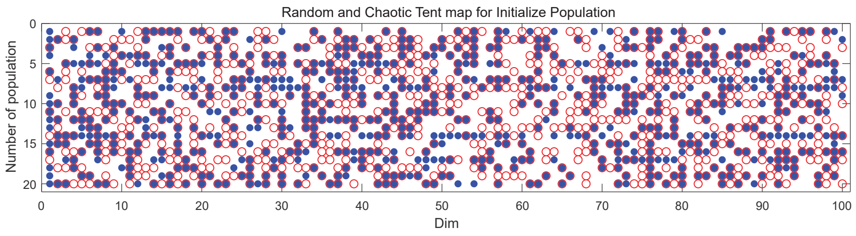


Figure 3. Random and CTM mechanisms for initialize population, where red dot represents the random mechanism, and blue represents CTM mechanism.

4.2.2. Cosine Similarity for Position Update

The conventional GJO algorithm (Algorithm 3) updates the position of jackals by Equation (9) during the iteration process, equivalent to using the mean as a more optimal solution. Although this method can ensure the smoothness of jackal position updates, it has some drawbacks. The most obvious flaw is that it does not consider the correlation between different features. When there is a correlation between features, using the mean update mechanism may lead to some features being overemphasized or ignored, thereby affecting the model's performance. In addition, when the data distribution is uneven, using the mean update mechanism may lead to poor prediction performance of the model for specific data. Therefore, we propose cosine similarity for positions updating of FMJ and MJ. Compared with the mean update mechanism, the advantage of using cosine similarity as the update mechanism is that it can consider the correlation between different features, thus updating model parameters more accurately. In addition, cosine similarity is not affected by vector length and data distribution and is suitable for high-dimensional data [43]. The mathematical model of cosine similarity is defined as follows:

$$Cos_{sim}(Y_1(t), Y_2(t)) = \frac{Y_1(t) \cdot Y_2(t)}{\|Y_1(t)\| \|Y_2(t)\|} \tag{11}$$

where the $Y_1(t)$ and $Y_2(t)$ represent the position of FMJ and MJ, respectively, and the $\cdot$ means dot product. $||Y_1(t)||$ and $||Y_2(t)||$ represent the lengths of FMJ and MJ, respectively. The value range of $Cos_{sim}(Y_1(t), Y_2(t))$ is [−1, 1]. In this paper, we improve the cosine similarity between golden jackal pairs, using the absolute value as the weight of position update, which is defined as follows:

$$Y(t+1) = Y_1(t) \times |Cos_{sim}(Y_1(t), Y_2(t))| + Y_2(t) \times (1 - |Cos_{sim}(Y_1(t), Y_2(t))|) \tag{12}$$

Algorithm 3 Improved Binary Golden Jackal Optimization

```
Require: The size of population $N_{pop}$, solution dimension $N_{dim}$, the max number of iterations $T_{max}$, lower and upper bounds $L_b$, $U_b$, the fitness function, the golden jackal $GJ$, $prey$, etc.
Ensure: The best solution found in the search process
 1: Initializing the population through chaotic tent mechanism by Algorithm 2
 2: While $(t < T_{max})$
 3:   Calculate the fitness values of preys
 4:   $Y_1$ = best prey (Male Jackal position)
 5:   $Y_2$ = second best prey (Female Jackal Position)
 6:   for (each prey)
 7:     Update the evading energy $E$ using Equations (6a), (6b) and (6c)
 8:     Update $rl$ using Equations (7) and (8)
 9:     if $(|E| >= 1)$   // (Exploration phase)
10:       Update the prey position using Equations (5b), (5a) and (12)
11:     if $(|E| < 1)$   //(Exploitation phase)
12:       Update the prey position using Equations (10a), (10b) and (12)
13:   end for
14:   $t = t + 1$
15: end While
16: return $Y_1$
```

4.2.3. Binary Mechanism Sigmoid

The solutions in conventional GJO are continuous and can be updated using the Equations (5a), (5b), (10a) and (10b) directly. However, the solution space of the formulated feature selection problem is discrete, which cannot be handled by conventional GJO. Therefore, it is suitable for feature selection problems by introducing a binary mechanism to map the solutions from continuous to discrete space. For the solution mappings in this work, the commonly used *S*-shaped transfer function [44], i.e., the Sigmoid function, is applied to conventional GJO and IBGJO. The details of this function are as follows. Moreover, the binary mechanism is elucidated in Equations (13) and (14) as follows:

$$x_{sig} = \frac{1}{1 + e^{-x}}, \tag{13}$$

$$x_{binary} = \begin{cases} 1, & N_{random} \le x_{sig} \\ 0, & N_{random} > x_{sig} \end{cases} \tag{14}$$

where x_{binary} is the converted binary solution of the feature selection problem, and N_{random} is a random number used as the threshold. Figure 4 presents the binary mechanism that we used in this paper.

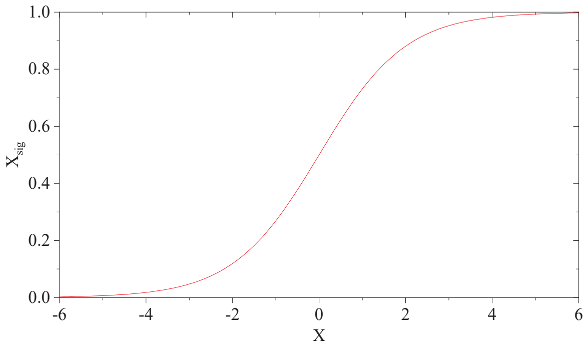


Figure 4. Sigmoid Binary transfer function.

4.3. Feature Selection Based on IBGJO

A solution could be viewed as a golden jackal for the formulated feature selection problem when employing the suggested IBGJO. Consequently, the answer could be stated as follows:

$$g = (G_1, G_2, G_3 \ldots, G_{N_{dim}}) \tag{15}$$

where N_{dim} represents the number of features while N_{pop} is the number of individuals, thus, the IBGJO population is expressed as follows:

$$\begin{aligned} pop &= \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_{N_{pop}} \end{bmatrix} \\ &= \begin{bmatrix} G_1^1 & G_2^1 & G_3^1 & \cdots & G_{N_{dim}}^1 \\ G_1^2 & G_2^2 & G_3^2 & \cdots & G_{N_{dim}}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ G_1^{N_{pop}} & G_2^{N_{pop}} & G_3^{N_{pop}} & \cdots & G_{N_{dim}}^{N_{pop}} \end{bmatrix} \end{aligned} \tag{16}$$

4.4. Computational Complexity

The complexity of the conventional GJO algorithm depends on various factors, including the size of the individuals N_{pop}, and the number of iterations T_{max}. The exploration phase or exploitation phase is performed in each iteration. Therefore, the overall time complexity of conventional GJO consists of the exploration and exploitation phase. Thus, the overall time complexity of conventional GJO is given as follows:

$$\begin{aligned} O(GJO) &= O(T_{max}(O(exploration) + O(exploration))) \\ &= O(T_{max}(N_{pop} \cdot N_{dim} + N_{pop} \cdot N_{dim})) \\ &= O(T_{max} \cdot N_{pop} \cdot N_{dim}) \end{aligned} \tag{17}$$

Since the structure of the proposed IBGJO is similar to conventional GJO, therefore, the computational complexity of IBGJO is also determined to be $O(T_{max} \cdot N_{pop} \cdot N_{dim})$. As a result, for a given feature selection problem, IBGJO does not require noticeably more computation time than conventional GJO, as both conventional GJO and IBGJO algorithms possess equivalent computational complexity. Notably, the average execution time of IBGJO in the experimental results is better than that of conventional GJO; this may be attributed to the enhanced factors employed in IBGJO, which will help improve the searchability of IBGJO and promote its fast convergence.

5. Experiments and Analysis

In this section, we conduct tests to evaluate the performance of the proposed IBGJO algorithm for dealing with feature selection problems. First, the datasets and setups used in the experiments are introduced. Then, the test results obtained by IBGJO and several comparison algorithms are presented and analyzed. Moreover, several other algorithms are selected for comparison.

5.1. Datasets and Setup

In this work, we provide the datasets used in this article and the parameter settings for the experiment.

5.1.1. Benchmark Datasets

This section introduces the benchmark datasets used in different algorithms' evaluations and parameter setups. Due to the fact that the UCI dataset covers multiple fields, such as Life, Social, Physical and so on, many research works use the UCI dataset as the benchmark data. For example, 10, 14, 16 and 20 datasets in the UCI dataset were respectively selected as experimental data in [21,45–47]. Therefore, the datasets used in our experiments refer to some datasets in their work and has been expanded to 28 datasets. By using these well-known datasets, we intended to facilitate comparisons with existing algorithms and provide a basis for future research. The primary information of these datasets is shown in Table 1.

Table 1. Benchmark datasets.

No.	Dataset	Instances	Features	Classses	Attribute Type	Area
D1	Arrhythmia	452	278	16	Categorical, Integer, Real	Life
D2	Breastcancer	699	10	5	Integer	Life
D3	BreastEW	569	30	2	Real	Life
D4	Congress	435	16	2	Categorical	Social
D5	Connectionist	208	60	2	Real	Physical
D6	Dermatology	366	34	6	Categorical, Integer	Life
D7	Diabets	768	8	7	Categorical, Integer	Computer
D8	German	1000	24	2	Categorical, Integer	Business
D9	HeartEW	270	13	2	Categorical, Real	Life
D10	Heart-StatLog	270	13	2	Categorical, Real	Life
D11	Hillvalley	606	100	2	Real	N/A
D12	Ionosphere	351	34	2	Integer, Real	Life
D13	Krvskp	3196	36	2	Categorical	Game
D14	Low-res-spect	531	102	9	Integer, Real	Physical
D15	Lung	72	326	7	Integer	Life
D16	Lung-Cancer	32	56	3	Integer	Life
D17	Lymphography	148	18	8	Categorical	Physical
D18	Parkinsons	1040	26	2	Integer, Real	Life
D19	Planning	182	13	2	Real	Computer
D20	Sonar	208	60	2	Real	Physical
D21	Spect	267	22	2	Categorical	Life
D22	Steel-plates	1941	27	7	Integer, Real	Physical
D23	Thyroid	7200	21	3	Categorical, Real	Life
D24	Tic-tac-toe	958	9	2	Categorical	Game
D25	WDBC	569	31	2	Real	Life
D26	Wine	178	13	3	Integer, Real	Physical
D27	Zoo	101	16	7	Categorical, Integer	Life
D28	RNA-Seq	802	16384	4	Real	Life

5.1.2. Experiment Setup

We compare IBMRFO with several other algorithms for feature selection experiments, including BCS, BGWO, BHBA, BMPA, BGJO, and IBGJO. It should be noted that all algorithms use the exact binary mechanisms. At the same time, BGJO is a binary version of the conventional GJO algorithm. IBGJO parameters are based on those of the conventional GJO algorithm, which has only one adaptive coefficient vector. Unlike other algorithms, conventional GJO and IBGJO require no additional tuning. The critical parameter choices for these algorithms are presented in Table 2, with specific values based on prior evidence of consistently strong performance in the literature for each algorithm, enabling effective feature comparison.

Table 2. Key parameters of different algorithms.

No.	Algorithm	Parameters
1	BCS	Discovery probability = 0.25, α = 1
2	BGWO	α =[2,0]
3	HBA	β = 6, C = 2, vec_{flag}= [1, −1]
4	MPA	P = 0.5, $FADS$ = 0.2
5	BGJO	e_0 = [−1, 1]
6	IBGJO	e_0 = [−1, 1]

Moreover, because both the proposed IBGJO and these comparison algorithms are metaheuristics, the size of the population and the number of iterations directly impact them. To guarantee that the comparison is fair, the population size and the number of algorithm iterations must be consistent. The population size and iteration count for each algorithm in this study are set to 20 and 200, respectively. Additionally, to prevent the experiment's random bias, each algorithm is independently performed 30 times in these chosen datasets, as suggested by the central limit theorem. The experiment's Intel(R) Core(R) I9-12900KF CPU and 64 GB of RAM were employed. Using Python 3.9.12 and the KNN [48] (k = 5) based on Euclidean distance measurement, we put the trials into practice. It is worth noting that a common approach has been employed in several previous works where 80% of the instances are used for training purposes, while the remaining instances are reserved for testing. Moreover, in the fitness function α and β are set to 0.99 and 0.01, respectively.

5.2. Feature Selection Results

This section presents the feature selection results of various algorithms in terms of average fitness function value, convergence speed, average accuracy, and average CPU time. Also, the best results are shown in bold.

5.2.1. Performance Evaluation

To explicitly demonstrate the performance of various algorithms, the fitness function values achieved by those algorithms are shown in Table 3. Table 3 shows the numerical statistical results of each dataset's average fitness function value and standard deviation (std) of different algorithms. For the average fitness values on 28 datasets, BCS, BGWO, BHBA, BMPA, BGJO, and IBGJO, they achieved the best performance on 3, 3, 5, 7, 9, and 14 datasets, respectively. This demonstrates our conjecture that BGJO may have a good exploration ability but lacks exploitation performance. Thus, by introducing the improved factors to conventional BGJO, the proposed improvement factors are practical. Compared with conventional BGJO, IBGJO has an advantage on average fitness value in 21 datasets. Moreover, IBGJO obtains the best stds of fitness values in 11 datasets, which means that IBGJO is more stable than others regarding feature selection.

Table 3. Average fitness function values obtained by different algorithms.

Datastet	BCS	BGWO	BHBA	BMPA	BGJO	IBGJO
	Fitness	Fitness	Fitness	Fitness	Fitness	Fitness
D1	0.3526 ± 0.0042	**0.3440** ± 0.0080	0.3489 ± 0.0052	0.3491 ± 0.0050	0.3477 ± 0.0041	0.3482 ± **0.0038**
D2	**0.0268** ± **0.0000**	0.0297 ± 0.0029	**0.0268** ± **0.0000**	**0.0268** ± **0.0000**	**0.0268** ± **0.0000**	0.0272 ± **0.0000**
D3	0.0452 ± 0.0015	0.0505 ± 0.0054	0.0440 ± 0.0006	**0.0433** ± 0.0009	0.0437 ± **0.0006**	**0.0433** ± 0.0009
D4	0.0274 ± 0.0025	0.0309 ± 0.0045	0.0256 ± 0.0020	0.0259 ± 0.0028	**0.0253** ± 0.0020	0.0269 ± **0.0018**
D5	0.1427 ± **0.0055**	0.1391 ± 0.0131	0.1382 ± 0.0072	0.1383 ± 0.0095	0.1378 ± 0.0070	**0.1362** ± 0.0095
D6	0.0165 ± 0.0017	0.0175 ± 0.0032	0.0149 ± **0.0013**	0.0169 ± 0.0017	0.0144 ± 0.0017	**0.0139** ± 0.0017
D7	**0.2557** ± **0.0000**	0.2588 ± 0.0042	**0.2557** ± **0.0000**	**0.2557** ± **0.0000**	**0.2557** ± **0.0000**	**0.2557** ± **0.0000**
D8	0.2523 ± 0.0037	0.2572 ± 0.0076	0.2517 ± **0.0027**	0.2535 ± 0.0032	**0.2503** ± 0.0030	0.2505 ± 0.0035
D9	0.1629 ± 0.0066	0.1733 ± 0.0104	0.1590 ± 0.0052	0.1587 ± 0.0044	0.1543 ± **0.0040**	**0.1537** ± 0.0046
D10	0.1463 ± 0.0054	0.1595 ± 0.0238	**0.1392** ± 0.0030	0.1414 ± 0.0049	0.1450 ± 0.0025	0.1440 ± **0.0024**
D11	0.4113 ± **0.0031**	**0.3995** ± 0.0064	0.4089 ± 0.0039	0.4076 ± 0.0041	0.4088 ± 0.0042	0.4079 ± 0.0042
D12	0.1374 ± **0.0048**	0.1326 ± 0.0112	0.1325 ± 0.0074	**0.1267** ± 0.0056	0.1307 ± 0.0058	0.1293 ± 0.0070
D13	0.0361 ± 0.0039	0.0373 ± 0.0053	0.0336 ± **0.0024**	0.0379 ± 0.0029	0.0325 ± 0.0025	**0.0316** ± 0.0026
D14	0.1170 ± **0.0016**	0.1148 ± 0.0035	0.1156 ± 0.0021	0.1154 ± 0.0019	0.1157 ± 0.0023	**0.1148** ± 0.0018
D15	0.1316 ± **0.0069**	0.1265 ± 0.0171	0.1230 ± 0.0093	0.1224 ± 0.0088	0.1216 ± 0.0098	**0.1197** ± 0.0107
D16	0.0386 ± 0.0113	0.0379 ± 0.0116	0.0327 ± 0.0072	0.0312 ± 0.0061	0.0302 ± 0.0006	**0.0301** ± **0.0005**
D17	0.5696 ± 0.0088	0.5837 ± 0.0167	**0.5618** ± 0.0069	0.5639 ± 0.0104	0.5651 ± 0.0055	0.5645 ± **0.0053**
D18	0.0988 ± 0.0116	0.1129 ± **0.0071**	0.0962 ± 0.0108	**0.0879** ± 0.0077	0.0903 ± 0.0104	0.0882 ± 0.0085
D19	0.2752 ± 0.0054	0.2992 ± 0.0233	0.2719 ± 0.0011	0.2749 ± 0.0052	**0.2716** ± **0.0000**	**0.2716** ± **0.0000**
D20	0.1147 ± **0.0057**	0.1094 ± 0.0137	0.1095 ± 0.0059	0.1092 ± 0.0080	0.1070 ± 0.0081	**0.1035** ± 0.0078
D21	0.2766 ± 0.0067	0.2792 ± 0.0121	0.2707 ± 0.0076	0.2736 ± 0.0075	**0.2678** ± 0.0071	0.2708 ± **0.0055**
D22	0.3193 ± 0.0324	0.4235 ± 0.1062	0.3286 ± 0.0344	0.2953 ± 0.0272	**0.2687** ± **0.0126**	0.2731 ± 0.0140
D23	0.0276 ± 0.0019	0.0314 ± 0.0049	0.0248 ± 0.0014	0.0252 ± 0.0016	0.0239 ± 0.0014	**0.0238** ± 0.0014
D24	**0.1523** ± **0.0000**	0.1541 ± 0.0098	**0.1523** ± **0.0000**	**0.1523** ± **0.0000**	0.1564 ± **0.0000**	0.1564 ± **0.0000**
D25	0.0462 ± 0.0018	0.0537 ± 0.0082	0.0443 ± 0.0007	**0.0437** ± 0.0007	0.0440 ± 0.0006	**0.0437** ± 0.0008
D26	0.0517 ± 0.0033	0.0621 ± 0.0128	0.0491 ± 0.0021	0.0494 ± 0.0021	**0.0483** ± 0.0005	**0.0483** ± **0.0004**
D27	0.0534 ± 0.0090	0.0794 ± 0.0180	0.0463 ± 0.0061	0.0512 ± 0.0070	**0.0442** ± **0.0047**	0.0634 ± 0.0051
D28	0.6865 ± **0.0024**	**0.6634** ± 0.0045	0.6844 ± **0.0024**	0.6826 ± 0.0029	0.6828 ± 0.0030	0.6817 ± 0.0.0027

Due to space restrictions, many such figures are divided into three parts, and each curve is taken from the 15th test. The convergence rates of various algorithms used in the optimization processes are shown in Figures 5–7. These figures demonstrate that the proposed IBGJO exposes the best curves on 20 datasets and has the best convergence capability compared to all other comparison algorithms. Overall, the proposed IBGJO performs better than other comparison algorithms for solving the formulated feature selection problem. Note that the effectiveness of different improved factors is further verified and discussed in Section 5.3.

5.2.2. Features Selection Accuracy of Algorithms

The feature selection accuracy obtained by various algorithms is shown in Table 4. The IBGJO algorithm achieves the best average accuracies of feature selection results on 14 datasets. Moreover, IBGJO obtains better accuracy than conventional BGJO in 21 datasets. Thus, compared with other algorithms, the IBGJO algorithm has the best performance in terms of feature selection accuracies on these selected datasets. The reason could be that the improved factors can balance the exploration and exploitation abilities, improving the algorithm's performance. However, it is crucial to recognize that achieving optimal results for accuracy and the number of selected features is a challenging tradeoff that varies across datasets.

Therefore, it can be concluded that the proposed IBGJO algorithm displays superior overall performance in feature selection across the selected datasets as compared to the other algorithms according to Tables 4 and 5.

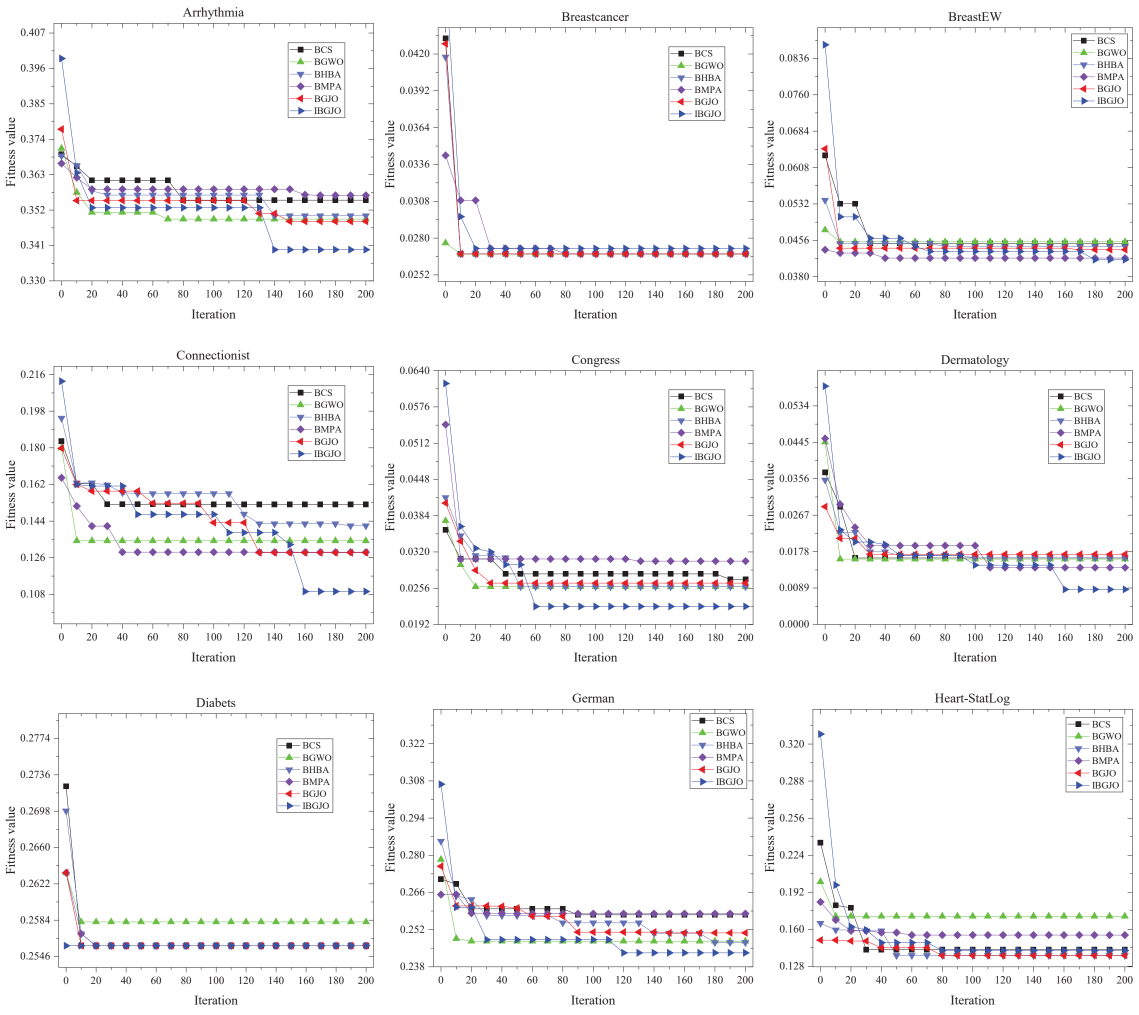


Figure 5. Convergence rates obtained by different algorithms (Part 1).

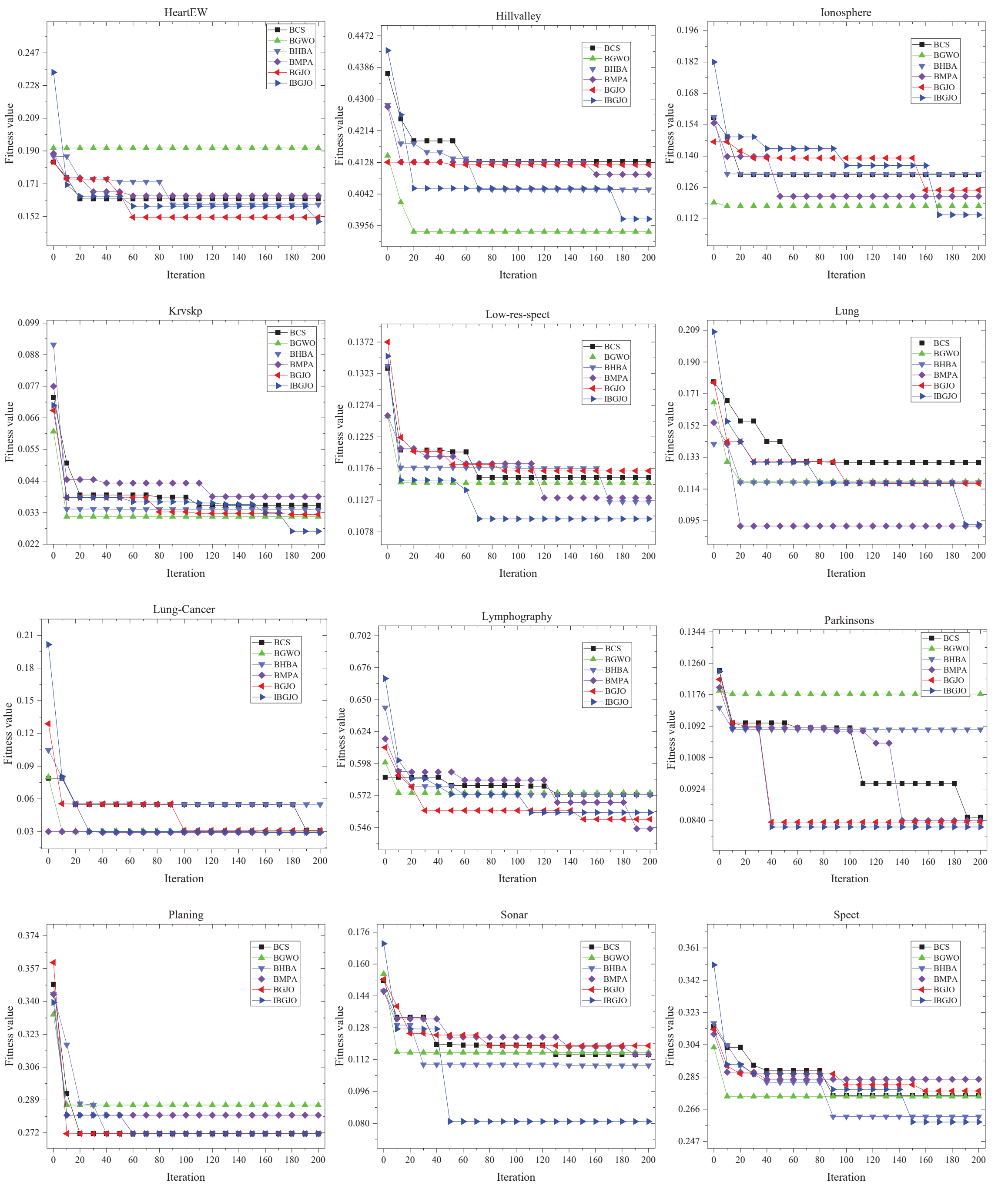


Figure 6. Convergence rates obtained by different algorithms (Part 2).

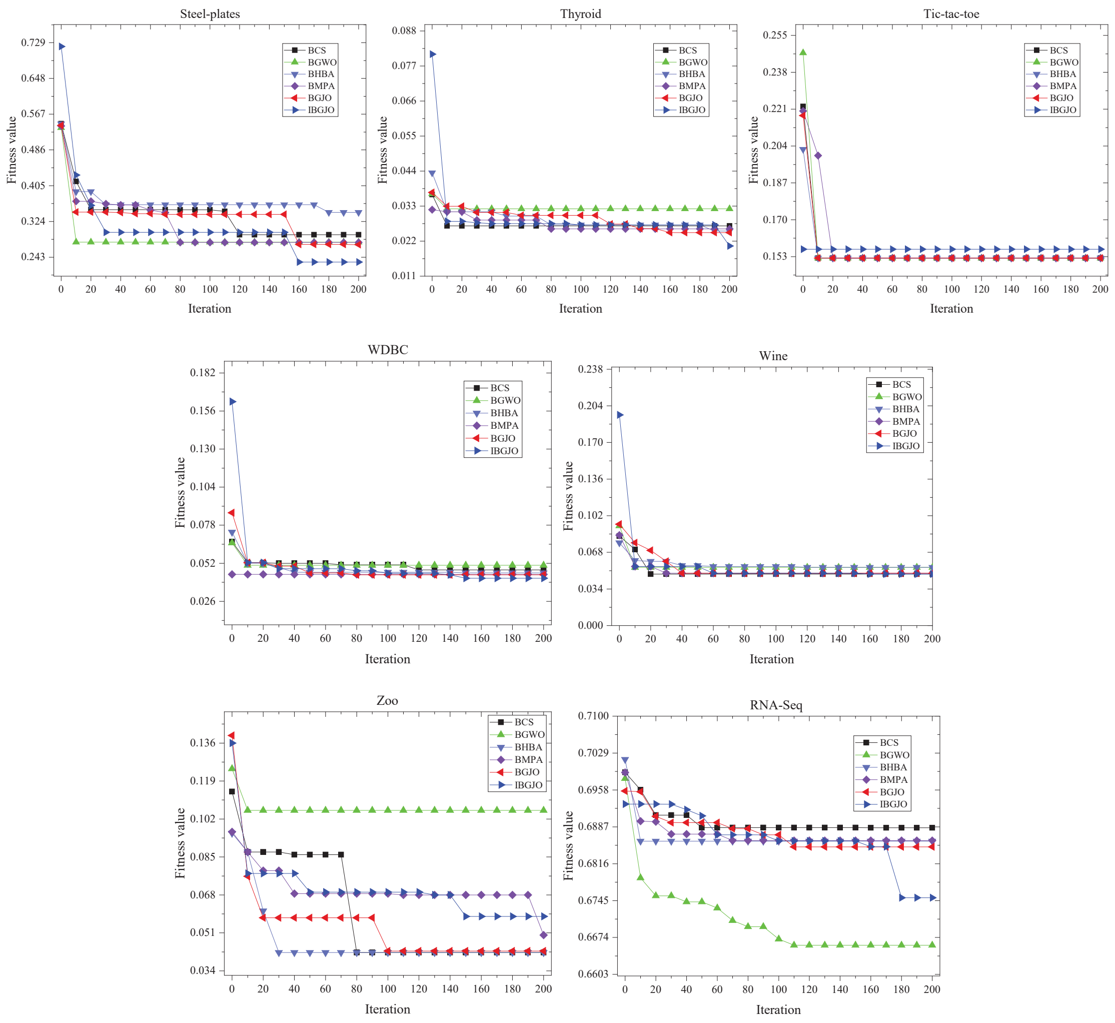


Figure 7. Convergence rates obtained by different algorithms (Part 3).

5.2.3. Number of Selected Features

The counts of the selected features from the datasets acquired by various techniques are displayed in Table 5. Similar to the accuracy results, these tables likewise display the outcomes of numerical statistics. BMPA obtains the best average number of selected features in the majority of the datasets (20 of 28), which may be regarded as the best results in the tests compared to other algorithms. This is shown in Table 5. Meanwhile, the number of features of IBGJO has an advantage in 15 datasets compared to that of BGJO. It is important to note that there exists a tradeoff between accuracy and the number of selected features, making it challenging to achieve optimal results for both objectives in each dataset.

Table 4. Classification accuracies and standard deviation achieved by different algorithms.

Datastet	BCS	BGWO	BHBA	BMPA	BGJO	IBGJO
	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
D1	0.6501 ± 0.0043	**0.6598** ± 0.0081	0.6539 ± 0.0053	0.6529 ± 0.0050	0.6551 ± 0.0042	0.6546 ± **0.0038**
D2	**0.9800** ± **0.0000**	0.9771 ± 0.0028	**0.9800** ± **0.0000**	**0.9800** ± **0.0000**	**0.9800** ± **0.0000**	0.9786 ± **0.0000**
D3	0.9594 ± 0.0010	0.9540 ± 0.0053	0.9598 ± **0.0004**	0.9599 ± 0.0007	0.9598 ± 0.0005	**0.9603** ± 0.0009
D4	0.9777 ± 0.0023	0.9748 ± 0.0041	0.9795 ± 0.0015	0.9790 ± 0.0025	**0.9797** ± 0.0016	0.9777 ± **0.0015**
D5	0.8617 ± **0.0054**	0.8662 ± 0.0130	0.8663 ± 0.0073	0.8654 ± 0.0095	0.8668 ± 0.0069	**0.8683** ± 0.0098
D6	0.9897 ± 0.0020	0.9900 ± 0.0031	0.9915 ± **0.0015**	0.9892 ± 0.0018	0.9921 ± 0.0017	**0.9923** ± .0018
D7	**0.7468** ± **0.0000**	0.7445 ± 0.0043	**0.7468** ± **0.0000**	**0.7468** ± **0.0000**	**0.7468** ± **0.0000**	**0.7468** ± **0.0000**
D8	0.7518 ± 0.0039	0.7476 ± 0.0077	0.7522 ± **0.0028**	0.7497 ± 0.0034	**0.7538** ± 0.0032	0.7535 ± 0.0037
D9	0.8394 ± 0.0059	0.8299 ± 0.0098	0.8430 ± 0.0048	0.8427 ± 0.0043	0.8473 ± 0.0038	**0.8496** ± **0.0037**
D10	0.8567 ± 0.0057	0.8444 ± 0.0235	**0.8646** ± 0.0035	0.8617 ± 0.0055	0.8569 ± **0.0018**	0.8575 ± 0.0019
D11	0.5907 ± **0.0031**	**0.6037** ± 0.0062	0.5931 ± 0.0039	0.5937 ± 0.0041	0.5932 ± 0.0042	0.5942 ± 0.0042
D12	0.8661 ± **0.0047**	0.8715 ± 0.0106	0.8710 ± 0.0072	**0.8758** ± 0.0054	0.8727 ± 0.0055	0.8738 ± 0.0066
D13	0.9699 ± 0.0040	0.9698 ± 0.0052	0.9725 ± 0.0024	0.9678 ± 0.0029	0.9736 ± **0.0024**	**0.9746** ± 0.0026
D14	0.8881 ± **0.0015**	**0.8910** ± 0.0035	0.8894 ± 0.0021	0.8887 ± 0.0022	0.8893 ± 0.0022	0.8902 ± 0.0018
D15	0.8733 ± **0.0070**	0.8788 ± 0.0174	0.8821 ± 0.0095	0.8817 ± 0.0090	0.8833 ± 0.0100	**0.8854** ± 0.0109
D16	0.9667 ± 0.0118	0.9675 ± 0.0115	0.9725 ± 0.0075	0.9733 ± 0.0062	**0.9750** ± **0.0000**	**0.9750** ± **0.0000**
D17	0.4302 ± 0.0087	0.4164 ± 0.0165	**0.4380** ± 0.0069	0.4353 ± 0.0103	0.4344 ± 0.0056	0.4356 ± **0.0056**
D18	0.9050 ± 0.0118	0.8905 ± **0.0072**	0.9073 ± 0.0112	**0.9155** ± 0.0080	0.9132 ± 0.0109	0.9153 ± 0.0090
D19	0.7275 ± 0.0057	0.7039 ± 0.0233	0.7312 ± 0.0013	0.7279 ± 0.0056	**0.7316** ± **0.0000**	**0.7316** ± **0.0000**
D20	0.8903 ± **0.0056**	0.8963 ± 0.0136	0.8956 ± 0.0061	0.8949 ± 0.0081	0.8978 ± 0.0081	**0.9016** ± 0.0079
D21	0.7265 ± 0.0066	0.7243 ± 0.0118	0.7322 ± 0.0077	0.7291 ± 0.0077	**0.7353** ± 0.0071	0.7323 ± **0.0054**
D22	0.6826 ± 0.0325	0.5773 ± 0.1076	0.6732 ± 0.0347	0.7060 ± 0.0273	**0.7332** ± **0.0125**	0.7289 ± 0.0140
D23	0.9765 ± 0.0018	0.9733 ± 0.0047	0.9791 ± 0.0014	0.9787 ± 0.0016	**0.9800** ± **0.0012**	**0.9800** ± **0.0012**
D24	**0.8563** ± **0.0000**	0.8543 ± 0.0103	**0.8563** ± **0.0000**	**0.8563** ± **0.0000**	0.8521 ± **0.0000**	0.8521 ± **0.0000**
D25	0.9585 ± 0.0015	0.9508 ± 0.0082	0.9596 ± **0.0000**	0.9596 ± 0.0005	0.9598 ± 0.0004	**0.9599** ± 0.0006
D26	0.9528 ± 0.0031	0.9428 ± 0.0125	0.9548 ± 0.0019	0.9546 ± 0.0021	**0.9556** ± **0.0000**	**0.9556** ± **0.0000**
D27	0.9524 ± 0.0090	0.9270 ± 0.0183	0.9597 ± 0.0065	0.9539 ± 0.0074	**0.9618** ± 0.0050	0.9412 ± **0.0046**
D28	0.3130 ± **0.0024**	**0.3379** ± 0.0046	0.3152 ± 0.0025	0.3160 ± 0.0029	0.3168 ± 0.0030	0.0.3179 ± 0.0028

Table 5. Number of selected features obtained by different algorithms with standard deviation.

Datastet	BCS	BGWO	BHBA	BMPA	BGJO	IBGJO
	Features	Features	Features	Features	Features	Features
D1	174.07 ± 3.05	200.67 ± 10.34	174.30 ± 8.14	**151.73** ± 11.51	175.03 ± 8.78	175.13 ± **7.79**
D2	7.00 ± **0.00**	7.07 ± 0.73	7.00 ± **0.00**	7.00 ± **0.00**	7.00 ± **0.00**	**6.00** ± **0.00**
D3	14.93 ± 2.43	14.67 ± 2.61	12.57 ± 1.82	**11.03** ± 1.87	11.80 ± 1.69	11.87 ± **1.31**
D4	8.60 ± 1.62	9.40 ± 1.43	8.47 ± 1.65	8.13 ± 1.50	8.37 ± 1.50	**7.80** ± **1.06**
D5	35.20 ± 4.37	39.67 ± 4.11	35.43 ± 3.14	**30.37** ± **3.11**	35.80 ± 3.78	34.63 ± 3.81
D6	21.63 ± 2.63	25.73 ± 2.14	22.23 ± **1.84**	**21.07** ± 2.10	22.37 ± 2.04	21.63 ± 2.24
D7	**4.00** ± **0.00**	4.73 ± 0.73	**4.00** ± **0.00**	**4.00** ± **0.00**	**4.00** ± **0.00**	**4.00** ± **0.00**
D8	15.83 ± 2.19	17.53 ± 2.00	15.30 ± 1.99	**13.73** ± 1.88	15.87 ± 2.13	15.53 ± **1.55**
D9	5.10 ± 1.19	6.33 ± 1.30	4.60 ± 0.88	**3.80** ± 0.79	4.07 ± **0.58**	6.23 ± 1.61
D10	5.70 ± 0.90	7.17 ± 1.04	6.60 ± **0.80**	5.90 ± 1.33	4.30 ± 1.21	**3.90** ± 0.96
D11	60.43 ± 6.32	71.83 ± 4.31	60.30 ± 4.45	**52.93** ± 5.30	60.60 ± 4.59	60.83 ± **4.18**
D12	16.53 ± 3.19	18.23 ± 2.78	16.20 ± 2.91	**12.70** ± **2.25**	15.77 ± 2.45	14.73 ± 3.02
D13	22.63 ± 2.12	26.57 ± 2.49	22.87 ± 2.03	**21.57** ± 2.73	22.87 ± **1.80**	23.13 ± 2.50
D14	61.77 ± 4.47	68.37 ± **4.31**	61.03 ± 4.56	**52.53** ± 4.79	60.67 ± 4.95	61.03 ± 5.63
D15	200.27 ± 11.05	208.70 ± 14.09	203.43 ± **8.02**	**170.80** ± 11.77	197.43 ± 11.71	202.13 ± 8.95
D16	31.53 ± 3.74	32.03 ± 3.82	30.57 ± **2.56**	**27.13** ± 3.20	30.70 ± 3.14	30.03 ± 2.64
D17	9.97 ± 1.70	10.77 ± 1.61	9.70 ± **1.32**	**8.77** ± 1.82	9.37 ± 1.38	10.20 ± 1.42
D18	10.87 ± 1.77	10.37 ± 2.01	10.27 ± **1.59**	**9.87** ± 1.86	10.00 ± 1.89	10.17 ± 1.86
D19	**6.57** ± 0.50	7.20 ± 1.42	6.93 ± 0.25	6.60 ± 0.49	7.00 ± **0.00**	7.00 ± **0.00**
D20	36.80 ± **2.79**	40.57 ± 3.88	36.53 ± 3.94	**31.17** ± 2.88	34.83 ± 3.27	36.30 ± 3.59
D21	12.83 ± **1.27**	13.87 ± 1.61	12.40 ± 1.45	**12.00** ± 1.83	12.63 ± 1.50	12.77 ± 1.55
D22	13.67 ± 2.01	13.37 ± 1.99	13.80 ± 2.66	**11.33** ± 1.78	12.40 ± **1.71**	12.63 ± 1.97
D23	8.97 ± 1.28	10.50 ± 1.20	8.63 ± 1.38	8.60 ± **0.95**	8.67 ± 1.32	**8.27** ± 1.20
D24	9.00 ± **0.00**	**8.93** ± 0.36	9.00 ± **0.00**	9.00 ± **0.00**	9.00 ± **0.00**	9.00 ± **0.00**
D25	15.67 ± 2.37	15.57 ± 2.42	13.50 ± 2.06	**11.57** ± 1.71	12.80 ± 1.61	12.23 ± **1.50**
D26	6.47 ± 0.76	7.03 ± 1.02	5.67 ± 0.60	5.83 ± 0.90	5.63 ± 0.67	**5.53** ± **0.57**
D27	10.00 ± 1.53	11.30 ± 1.49	10.17 ± 1.24	9.00 ± 1.21	10.27 ± **0.74**	**8.33** ± 1.58
D28	10,531.2 ± 442.10	12,951.73 ± 384.55	10,605.6 ± 75.16	8971.07 ± 365.87	10,548.87 ± 109.24	10,511.37 ± 164.68

5.2.4. Algorithm Execution Time

The average running time of all algorithms is shown in Table 6. Based on the data presented in Table 6, it is evident that the IBGJO has an advantage in algorithm execution time. IBGJO is experimented on 28 datasets and compares the performance of different feature selection algorithms. Among these 28 datasets, our algorithm converged in the least average time on 19 datasets. This means that our algorithm has higher efficiency and faster convergence and can select the best subset of features in less time, thereby improving the performance of the model. This result shows that our algorithm has higher practicability and feasibility in practical applications.

Table 6. Average CPU time and standard deviation occupied by different algorithms (/s).

Datastet	**BCS**	**BGWO**	**BHBA**	**BMPA**	**BGJO**	**IBGJO**
	Time	**Time**	**Time**	**Time**	**Time**	**Time**
D1	94.33 ± **0.89**	588.37 ± 5.68	**92.33** ± 1.22	94.50 ± 1.10	449.13 ± 7.94	115.36 ± 12.48
D2	97.03 ± 1.37	121.97 ± 1.25	97.24 ± 0.08	96.84 ± **0.08**	111.89 ± 2.34	**79.24** ± 5.11
D3	**97.64** ± 2.10	157.52 ± 2.76	98.31 ± **0.73**	99.59 ± 1.05	139.61 ± 1.83	124.48 ± 10.02
D4	84.62 ± 0.30	116.75 ± 1.30	85.76 ± 0.54	83.37 ± **0.09**	108.24 ± 0.22	**72.70** ± 8.21
D5	389.48 ± 70.34	234.38 ± 46.57	423.26 ± 68.75	315.16 ± 75.90	471.42 ± 70.29	**85.94** ± **8.25**
D6	**69.31** ± 0.20	135.53 ± 1.93	71.70 ± 0.84	70.96 ± **0.09**	115.97 ± 1.54	89.77 ± 3.83
D7	100.13 ± 0.11	122.23 ± 1.44	102.85 ± 0.77	100.28 ± 0.11	112.59 ± **0.10**	**97.86** ± 14.02
D8	137.66 ± 0.98	192.54 ± 2.72	140.54 ± 1.04	135.96 ± **0.53**	174.12 ± 0.63	**125.01** ± 16.93
D9	65.53 ± 1.32	90.74 ± 0.51	65.83 ± 0.22	65.31 ± **0.04**	82.29 ± 0.38	**65.51** ± 8.20
D10	66.23 ± **0.03**	92.81 ± 0.78	67.01 ± 0.05	66.53 ± 0.08	61.54 ± 6.94	**58.60** ± 3.28
D11	96.49 ± 0.83	278.54 ± 3.70	97.79 ± 0.89	98.97 ± **0.49**	228.26 ± 4.15	**94.67** ± 19.25
D12	**75.61** ± 1.18	142.31 ± 2.82	75.95 ± 0.54	77.32 ± **0.26**	122.87 ± 1.38	104.94 ± 6.15
D13	664.07 ± 6.49	635.02 ± 6.44	578.17 ± **4.53**	707.68 ± 25.26	630.96 ± 15.30	**507.11** ± 89.94
D14	417.71 ± 32.64	192.66 ± **6.09**	242.64 ± 49.37	371.89 ± 77.69	123.42 ± 7.40	**120.51** ± 7.98
D15	52.69 ± **0.44**	637.31 ± 8.29	**51.72** ± 0.54	55.87 ± 0.67	473.91 ± 8.00	64.38 ± 6.19
D16	41.03 ± 0.42	144.48 ± 0.45	41.60 ± **0.05**	40.72 ± 0.06	111.79 ± 0.68	**28.65** ± 0.09
D17	53.97 ± 0.84	88.93 ± 0.49	54.70 ± 0.36	**53.14** ± **0.06**	54.37 ± 11.06	58.93 ± 10.84
D18	122.17 ± 21.75	126.41 ± 39.18	117.47 ± 9.85	122.53 ± 29.08	115.12 ± **7.12**	**75.68** ± 7.61
D19	86.25 ± 20.06	120.99 ± **0.71**	107.40 ± 20.81	105.63 ± 21.03	100.11 ± 23.92	**56.36** ± 8.58
D20	60.99 ± 0.57	173.97 ± 2.53	**60.23** ± **0.06**	61.72 ± 0.83	138.64 ± 0.14	71.42 ± 12.01
D21	65.07 ± **0.09**	107.92 ± 1.98	65.81 ± 0.10	66.57 ± 0.38	97.23 ± 1.23	**50.55** ± 0.75
D22	232.57 ± 22.23	478.93 ± 141.93	249.02 ± 76.10	304.31 ± 39.79	281.79 ± 38.30	**78.64** ± **2.29**
D23	1181.20 ± **68.71**	1204.65 ± 108.30	1204.61 ± 86.49	1178.79 ± 69.78	1186.05 ± 72.15	**576.16** ± 77.74
D24	123.54 ± 1.21	155.56 ± 2.17	128.09 ± 0.16	121.07 ± **0.15**	119.57 ± 10.52	**88.02** ± 0.27
D25	**96.00** ± 0.86	159.74 ± 2.28	97.21 ± 0.80	98.47 ± **0.63**	138.97 ± 1.37	126.37 ± 5.85
D26	55.53 ± 0.10	81.56 ± 0.62	57.16 ± **0.04**	55.83 ± 0.07	50.07 ± 4.24	**42.41** ± 0.05
D27	51.53 ± 1.03	79.69 ± 0.18	50.41 ± 0.05	49.34 ± **0.04**	70.65 ± 0.14	**45.02** ± 2.34
D28	9275.34 ± 1507.77	10,200.77 ± 2736.28	9238.49 ± 1519.09	8351.13 ± 1349.94	3107.74 ± **482.36**	6432.33 ± 801.44

5.3. Effectiveness of the Improved Factors

In this section, we conduct experiments to evaluate the effectiveness of the introduced factors in IBGJO. To observe whether these factors can impove the performance of BGJO, we use BGJO, BGJO with CTM mechanism (T-BGJO), BGJO with CS (C-BGJO), and BGJO both with CTM mechanism and CS together (IBGJO) to solve the formulated feature selection problem, respectively. The tests are also conducted on the nine selected datasets: Arrhythmia, Diabets, Heart-StatLog, Ionosphere, Krvskp, Lung, Parkinsons, Thyroid and WDBC. The numerical findings generated by these abovementioned algorithms are listed in Table 7. Overall, all algorithms obtain the same results on the Diabets dataset. This may be because this dataset has the lowest solution dimension, making it easy to solve. Furthermore, the convergence rates of different improvement factors used in optimization are shown in Figure 8. The remaining outcomes are discussed in detail as follows.

Table 7. Average results obtained by different improved factors of IBGJO.

		D1	D7	D10	D12	D13	D15	D18	D23	D25
BGJO	Accuracy	0.6551	**0.7468**	0.8569	0.8727	0.9736	0.8833	0.9132	**0.9800**	0.9598
	Feature.N	175.03	**4.00**	4.30	15.77	22.87	197.43	10.00	8.67	12.80
	Fitness	0.3477	**0.2557**	0.1450	0.1307	0.0325	0.1216	0.0903	0.0239	0.0440
	Time	449.13	112.59	61.54	122.87	630.96	473.91	115.12	1186.05	138.97
T-BGJO	Accuracy	0.6543	**0.7468**	0.8519	0.8728	0.9733	0.8829	0.9102	0.9794	0.9598
	Feature.N	**173.53**	**4.00**	5.27	**14.60**	23.07	198.47	**9.60**	8.47	12.80
	Fitness	0.3485	**0.2557**	0.1507	0.1302	0.0328	0.1220	0.0931	0.0244	0.0440
	Time	152.07	**93.60**	**56.72**	**80.57**	607.59	**41.60**	**61.62**	**539.04**	**104.90**
C-BGJO	Accuracy	**0.6553**	**0.7468**	0.8574	0.8727	0.9742	0.8838	0.9122	0.9797	0.9597
	Feature.N	174.90	**4.00**	4.37	15.53	**22.60**	**197.07**	10.33	8.57	12.90
	Fitness	**0.3476**	**0.2557**	0.1445	0.1306	0.0318	0.1212	0.0914	0.0241	0.0441
	Time	119.85	100.28	63.25	107.20	**503.21**	59.36	73.73	550.47	127.88
IBGJO	Accuracy	0.6546	**0.7468**	**0.8575**	**0.8738**	**0.9746**	**0.8854**	**0.9153**	**0.9800**	**0.9599**
	Feature.N	175.13	**4.00**	**3.90**	14.73	23.13	202.13	10.17	**8.27**	**12.23**
	Fitness	0.3482	**0.2557**	**0.1440**	**0.1293**	**0.0316**	**0.1197**	**0.0882**	**0.0238**	**0.0437**
	Time	**115.36**	97.86	58.60	104.94	507.11	64.38	75.68	576.16	126.37

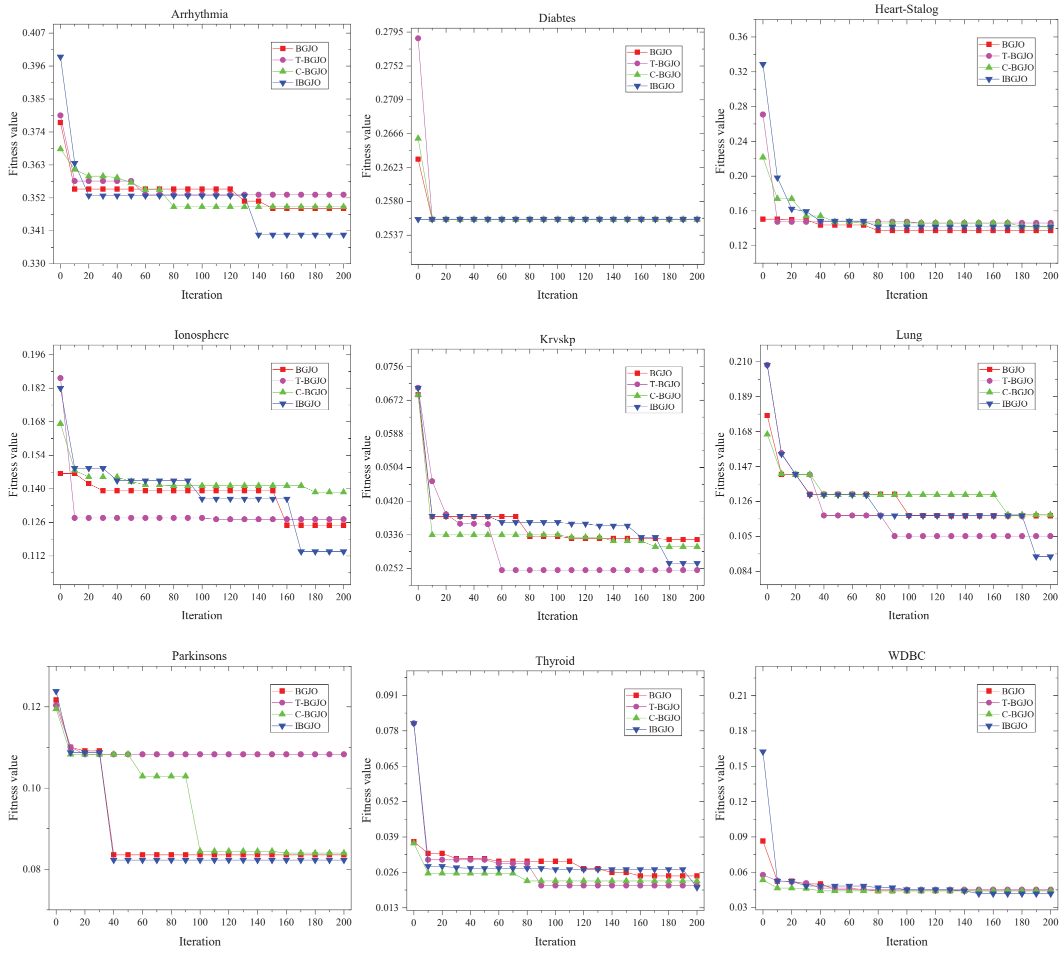


Figure 8. Convergence rate comparisons between conventional BGJO, T-BGJO, C-BGJO and IBGJO on different datasets.

5.3.1. Effectiveness of the Chaotic Tent Map (CTM) Mechanism

It can be seen from Table 7 that compared with the traditional BGJO algorithm, the T-BGJO algorithm does not have many advantages in fitness function value or classification accuracy. However, T-BGJO could efficiently select a fewer number of features than BGJO. Therefore, CTM has the advantage in feature number over other improved factors with cosine similarity that can help IBGJO obtain better performance.

5.3.2. Effectiveness of Cosine Similarity Position Update

In most datasets, especially medium-dimensional datasets, C-BGJO outperforms BGJO and T-BGJO in terms of accuracy of the fitness function values obtained, as shown in Table 7. This is due to the ability of the proposed CS position updating system to adaptively modify the searching scope to enhance BGJO's exploration capabilities. Note that, compared with the location update mechanism of the conventional BGJO, the CS requires additional calculation in each iteration. However, the searchability of IBGJO will be more robust with CS. Therefore, it could increase the convergence time.

5.3.3. Effectiveness of CTM and CS

It can be seen from Table 7 that the CTM mechanism effectively improves the representativeness and diversity of the initial population through a chaotic tent map and prevents the algorithm from falling into local optimum. Using CS as the jackal position update strategy in the golden jackal optimization algorithm can speed up the convergence speed of the algorithm and help IBGJO to converge faster. The combination of the two enhancement factors can effectively improve the algorithmic performance of BGJO in the field of feature selection.

To summarize, incorporating the two enhancement factors and binary mechanisms has effectively elevated the performance of the conventional BGJO algorithm and made it well-suited for feature selection. Furthermore, these components exhibit a complementary relationship. For instance, utilizing the CTM mechanism on small-sized datasets may cause the algorithm to encounter local optima frequently. Therefore, incorporating the CS is essential to address this problem.

5.4. Limitation of IBGJO

Although the experimental simulation results show that the proposed IBGJO algorithm outperforms some comparative algorithms, it still has some limitations. One limitation of the IBGJO algorithm is its sensitivity to parameter settings, requiring careful tuning for optimal performance. Additionally, the scalability of IBGJO to large-scale or high-dimensional datasets is a concern, as its computational complexity may become prohibitive. The generalization of IBGJO to different domains and problem types needs further exploration, as specific data characteristics may influence its performance. Furthermore, the interpretability of the selected feature subsets may not be guaranteed, as the algorithm prioritizes classification performance over intuitive feature combinations. The effectiveness and applicability of IBGJO can be improved by reducing the dimensionality of the feature set on the original dataset and then using the IBGJO algorithm for feature selection and optimizing the algorithm parameters.

6. Conclusions

The focus of this research work is to improve the classification performance of machine learning by addressing the issue of feature selection. An improved version of the BGJO algorithm, referred to as the IBGJO algorithm, is proposed to solve the feature selection problem. The IBGJO algorithm incorporates the CTM mechanism, CS location updating mechanism, and *S*-shape binary mechanism, designed to improve the performance of conventional BGJO and make it suitable for feature selection problems. Utilizing these improved factors allows the algorithm to balance its exploitation and exploration abilities while maintaining population diversity.

By using the improved factors, we can balance the development of the algorithm and its ability to explore different options while maintaining diversity within the population. We conducted experiments to test our proposed algorithm, IBGJO, on 28 well-known datasets and found that it outperformed other state-of-the-art algorithms such as BCS, BGWO, BHBA, BMPA and BGJO in terms of feature selection. We also evaluated the effectiveness of the improvement factors. We found that they helped to enhance the performance of the conventional golden jackal optimization algorithm. In the future, we plan to propose additional ways to update the location of the population and combine them with other evolutionary algorithms to tackle a broader range of optimization problems.

Author Contributions: Methodology, K.Z.; software, F.M. and J.J.; Validation, G.S. and K.Z.; writing—original draft preparation, K.Z. and G.S.; writing review and editing, G.S., Y.L. and K.Z.; visualization, K.Z. and J.J.; supervision, Y.L. and S.G.; funding acquisition, Y.L. and G.S. All authors have read and agreed to the published version of the manuscript.

Funding: This study is supported in part by the National Natural Science Foundation of China (62172186, 62002133, 61872158, 61806083), in part by the Science and Technology Development Plan Project of Jilin Province (20190701019GH, 20190701002GH, 20210101183JC, 20210201072GX, 20220101101JC), and in part by the Young Science and Technology Talent Lift Project of Jilin Province (QT202013).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Abbreviations

The following abbreviations are used in this manuscript:

Acc	Average accuracy
CS	Cosine similarity
EA	Evolutionary algorithm
FS	Feature selection
LD	Linear dichroism
MJ	Male golden jackal
FMJ	Female golden jackal
Std	Standard deviation

References

1. Cichos, F.; Gustavsson, K.; Mehlig, B.; Volpe, G. Machine learning for active matter. *Nat. Mach. Intell.* **2020**, *2*, 94–103. [CrossRef]
2. Alber, M.; Buganza Tepole, A.; Cannon, W.R.; De, S.; Dura-Bernal, S.; Garikipati, K.; Karniadakis, G.; Lytton, W.W.; Perdikaris, P.; Petzold, L.; et al. Integrating machine learning and multiscale modeling—Perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences. *NPJ Digit. Med.* **2019**, *2*, 115. [CrossRef] [PubMed]
3. Proto, S.; Di Corso, E.; Ventura, F.; Cerquitelli, T. Useful ToPIC: Self-tuning strategies to enhance latent Dirichlet allocation. In Proceedings of the 2018 IEEE International Congress on Big Data (BigData Congress), San Francisco, CA, USA, 2–7 July 2018; pp. 33–40.
4. Faris, H.; Heidari, A.A.; Ala'M, A.Z.; Mafarja, M.; Aljarah, I.; Eshtay, M.; Mirjalili, S. Time-varying hierarchical chains of salps with random weight networks for feature selection. *Expert Syst. Appl.* **2020**, *140*, 112898. [CrossRef]
5. Yu, L.; Liu, H. Efficient feature selection via analysis of relevance and redundancy. *J. Mach. Learn. Res.* **2004**, *5*, 1205–1224.
6. Daraio, E.; Di Corso, E.; Cerquitelli, T.; Chiusano, S. Characterizing air-quality data through unsupervised analytics methods. In Proceedings of the European Conference on Advances in Databases and Information Systems, Barcelona, Spain, 4–7 August 2018; pp. 205–217.
7. Solorio-Fernández, S.; Carrasco-Ochoa, J.A.; Martínez-Trinidad, J.F. A review of unsupervised feature selection methods. *Artif. Intell. Rev.* **2020**, *53*, 907–948. [CrossRef]
8. Li, J.; Kang, H.; Sun, G.; Feng, T.; Li, W.; Zhang, W.; Ji, B. IBDA: Improved binary dragonfly algorithm with evolutionary population dynamics and adaptive crossover for feature selection. *IEEE Access* **2020**, *8*, 108032–108051. [CrossRef]

9. Jović, A.; Brkić, K.; Bogunović, N. A review of feature selection methods with applications. In Proceedings of the 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 25–29 May 2015; pp. 1200–1205.
10. Chandrashekar, G.; Sahin, F. A survey on feature selection methods. *Comput. Electr. Eng.* **2014**, *40*, 16–28. [CrossRef]
11. Aličković, E.; Subasi, A. Breast cancer diagnosis using GA feature selection and Rotation Forest. *Neural Comput. Appl.* **2017**, *28*, 753–763. [CrossRef]
12. Huda, R.K.; Banka, H. Efficient feature selection and classification algorithm based on PSO and rough sets. *Neural Comput. Appl.* **2019**, *31*, 4287–4303. [CrossRef]
13. Osanaiye, O.; Cai, H.; Choo, K.K.R.; Dehghantanha, A.; Xu, Z.; Dlodlo, M. Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. *EURASIP J. Wirel. Commun. Netw.* **2016**, *2016*, 130. [CrossRef]
14. Agrawal, P.; Abutarboush, H.F.; Ganesh, T.; Mohamed, A.W. Metaheuristic algorithms on feature selection: A survey of one decade of research (2009–2019). *IEEE Access* **2021**, *9*, 26766–26791. [CrossRef]
15. Yang, X.S.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, 9–11 December 2009; pp. 210–214.
16. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
17. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
18. Too, J.; Liang, G.; Chen, H. Memory-based Harris hawk optimization with learning agents: A feature selection approach. *Eng. Comput.* **2021**, *38*, 4457–4478. [CrossRef]
19. Hussain, K.; Neggaz, N.; Zhu, W.; Houssein, E.H. An efficient hybrid sine-cosine Harris hawks optimization for low and high-dimensional feature selection. *Expert Syst. Appl.* **2021**, *176*, 114778. [CrossRef]
20. Hegazy, A.E.; Makhlouf, M.; El-Tawel, G.S. Improved salp swarm algorithm for feature selection. *J. King Saud-Univ.-Comput. Inf. Sci.* **2020**, *32*, 335–344. [CrossRef]
21. Dadaneh, B.Z.; Markid, H.Y.; Zakerolhosseini, A. Unsupervised probabilistic feature selection using ant colony optimization. *Expert Syst. Appl.* **2016**, *53*, 27–42. [CrossRef]
22. Chopra, N.; Ansari, M.M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications. *Expert Syst. Appl.* **2022**, *198*, 116924. [CrossRef]
23. Hou, Y.; Li, J.; Yu, H.; Li, Z. BIFFOA: A novel binary improved fruit fly algorithm for feature selection. *IEEE Access* **2019**, *7*, 81177–81194. [CrossRef]
24. Sayed, S.A.F.; Nabil, E.; Badr, A. A binary clonal flower pollination algorithm for feature selection. *Pattern Recognit. Lett.* **2016**, *77*, 21–27. [CrossRef]
25. Baldo, A.; Boffa, M.; Cascioli, L.; Fadda, E.; Lanza, C.; Ravera, A. The polynomial robust knapsack problem. *Eur. J. Oper. Res.* **2023**, *305*, 1424–1434. [CrossRef]
26. Aghdam, M.H.; Ghasem-Aghaee, N.; Basiri, M.E. Text feature selection using ant colony optimization. *Expert Syst. Appl.* **2009**, *36*, 6843–6853. [CrossRef]
27. Nakamura, R.Y.M.; Pereira, L.A.M.; Costa, K.A.; Rodrigues, D.; Papa, J.P.; Yang, X.-S. BBA: A binary bat algorithm for feature selection. In Proceedings of the Graphics, Patterns and Images: 25th SIBGRAPI Conference, SIBGRAPI 2012, Ouro Preto, Brazil, 22–25 August 2012; pp. 291–297.
28. Li, W.; Kang, H.; Feng, T.; Li, J.; Yue, Z.; Sun, G. Swarm Intelligence-Based Feature Selection: An Improved Binary Grey Wolf Optimization Method. In Proceedings of the Knowledge Science, Engineering and Management: 14th International Conference, KSEM 2021, Tokyo, Japan, 14–16 August 2021; pp. 98–110.
29. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. BGSA: Binary gravitational search algorithm. *Nat. Comput.* **2010**, *9*, 727–745. [CrossRef]
30. Al-Tashi, Q.; Kadir, S.J.A.; Rais, H.M.; Mirjalili, S.; Alhussian, H. Binary optimization using hybrid grey wolf optimization for feature selection. *IEEE Access* **2019**, *7*, 39496–39508. [CrossRef]
31. Arora, S.; Anand, P. Binary butterfly optimization approaches for feature selection. *Expert Syst. Appl.* **2019**, *116*, 147–160. [CrossRef]
32. Xue, Y.; Xue, B.; Zhang, M. Self-adaptive particle swarm optimization for large-scale feature selection in classification. In *ACM Transactions on Knowledge Discovery from Data (TKDD)*; ACM Press: New York, NY, USA, 2019; Volume 13, pp. 1–27.
33. Zhang, Y.; Cheng, S.; Shi, Y.; Gong, D.w.; Zhao, X. Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm. *Expert Syst. Appl.* **2019**, *137*, 46–58. [CrossRef]
34. Tao, Z.; Huiling, L.; Wenwen, W.; Xia, Y. GA-SVM based feature selection and parameter optimization in hospitalization expense modeling. *Appl. Soft Comput.* **2019**, *75*, 323–332. [CrossRef]
35. Zhang, J.; Luo, Z.; Li, C.; Zhou, C.; Li, S. Manifold regularized discriminative feature selection for multi-label learning. *Pattern Recognit.* **2019**, *95*, 136–150. [CrossRef]
36. Dey, A.; Chattopadhyay, S.; Singh, P.K.; Ahmadian, A.; Ferrara, M.; Sarkar, R. A hybrid meta-heuristic feature selection method using golden ratio and equilibrium optimization algorithms for speech emotion recognition. *IEEE Access* **2020**, *8*, 200953–200970. [CrossRef]

37. Wang, M.; Chen, H. Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis. *Appl. Soft Comput.* **2020**, *88*, 105946. [CrossRef]
38. Anter, A.M.; Ali, M. Feature selection strategy based on hybrid crow search optimization algorithm integrated with chaos theory and fuzzy c-means algorithm for medical diagnosis problems. *Soft Comput.* **2020**, *24*, 1565–1584. [CrossRef]
39. Devi, R.M.; Premkumar, M.; Kiruthiga, G.; Sowmya, R. IGJO: An Improved Golden Jackel Optimization Algorithm Using Local Escaping Operator for Feature Selection Problems. *Neural Process. Lett.* **2023**, *14*, 1–89. [CrossRef]
40. Abdollahzadeh, B.; Gharehchopogh, F.S. A multi-objective optimization algorithm for feature selection problems. *Eng. Comput.* **2022**, *38*, 1845–1863. [CrossRef]
41. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
42. Sayed, G.I.; Tharwat, A.; Hassanien, A.E. Chaotic dragonfly algorithm: An improved metaheuristic algorithm for feature selection. *Appl. Intell.* **2019**, *49*, 188–205. [CrossRef]
43. Zhang, K.; Liu, Y.; Mei, F.; Jin, J.; Wang, Y. Boost Correlation Features with 3D-MiIoU-Based Camera-LiDAR Fusion for MODT in Autonomous Driving. *Remote Sens.* **2023**, *15*, 874. [CrossRef]
44. Ghosh, K.K.; Guha, R.; Bera, S.K.; Kumar, N.; Sarkar, R. S-shaped versus V-shaped transfer functions for binary Manta ray foraging optimization in feature selection problem. *Neural Comput. Appl.* **2021**, *33*, 11027–11041. [CrossRef]
45. Wang, P.; Xue, B.; Liang, J.; Zhang, M. Multiobjective differential evolution for feature selection in classification. *IEEE Trans. Cybern.* **2021**, *53*, 4579–4593. [CrossRef]
46. Wang, P.; Xue, B.; Liang, J.; Zhang, M. Feature clustering-Assisted feature selection with differential evolution. *Pattern Recognit.* **2023**, *140*, 109523. [CrossRef]
47. Xu, H.; Xue, B.; Zhang, M. A duplication analysis-based evolutionary algorithm for biobjective feature selection. *IEEE Trans. Evol. Comput.* **2020**, *25*, 205–218. [CrossRef]
48. Deng, Z.; Zhu, X.; Cheng, D.; Zong, M.; Zhang, S. Efficient kNN classification algorithm for big data. *Neurocomputing* **2016**, *195*, 143–148. [CrossRef]

entropy

Article

Water Quality Prediction Based on Machine Learning and Comprehensive Weighting Methods

Xianhe Wang [1,2], Ying Li [1,2], Qian Qiao [1], Adriano Tavares [2] and Yanchun Liang [3,4,*]

[1] School of Applied Chemistry and Materials, Zhuhai College of Science and Technology, Zhuhai 519041, China; wxh@zcst.edu.cn (X.W.); liying@zcst.edu.cn (Y.L.)
[2] Department of Industrial Electronics, School of Engineering, University of Minho, 4704-553 Braga, Portugal
[3] School of Computer Science, Zhuhai College of Science and Technology, Zhuhai 519041, China
[4] Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, College of Computer Science and Technology, Jilin University, 2699 Qianjin Street, Changchun 130012, China
* Correspondence: ycliang@jlu.edu.cn

Abstract: In the context of escalating global environmental concerns, the importance of preserving water resources and upholding ecological equilibrium has become increasingly apparent. As a result, the monitoring and prediction of water quality have emerged as vital tasks in achieving these objectives. However, ensuring the accuracy and dependability of water quality prediction has proven to be a challenging endeavor. To address this issue, this study proposes a comprehensive weight-based approach that combines entropy weighting with the Pearson correlation coefficient to select crucial features in water quality prediction. This approach effectively considers both feature correlation and information content, avoiding excessive reliance on a single criterion for feature selection. Through the utilization of this comprehensive approach, a comprehensive evaluation of the contribution and importance of the features was achieved, thereby minimizing subjective bias and uncertainty. By striking a balance among various factors, features with stronger correlation and greater information content can be selected, leading to improved accuracy and robustness in the feature-selection process. Furthermore, this study explored several machine learning models for water quality prediction, including Support Vector Machines (SVMs), Multilayer Perceptron (MLP), Random Forest (RF), XGBoost, and Long Short-Term Memory (LSTM). SVM exhibited commendable performance in predicting Dissolved Oxygen (DO), showcasing excellent generalization capabilities and high prediction accuracy. MLP demonstrated its strength in nonlinear modeling and performed well in predicting multiple water quality parameters. Conversely, the RF and XGBoost models exhibited relatively inferior performance in water quality prediction. In contrast, the LSTM model, a recurrent neural network specialized in processing time series data, demonstrated exceptional abilities in water quality prediction. It effectively captured the dynamic patterns present in time series data, offering stable and accurate predictions for various water quality parameters.

Keywords: water quality prediction; comprehensive weight-based approach; feature selection; machine learning; LSTM

Citation: Wang, X.; Li, Y.; Qiao, Q.; Tavares, A.; Liang, Y. Water Quality Prediction Based on Machine Learning and Comprehensive Weighting Methods. *Entropy* **2023**, *25*, 1186. https://doi.org/10.3390/e25081186

Academic Editor: Jose Santamaria

Received: 30 June 2023
Revised: 26 July 2023
Accepted: 2 August 2023
Published: 9 August 2023

1. Introduction

With the increasing human activities associated with industrialization and urbanization development, the water quality of coastal rivers is facing escalating and severe threats and degradation [1]. Coastal rivers play a critical role in connecting land and ocean, and the water quality directly impacts the well-being and sustainable development of coastal ecosystems. Therefore, it is imperative to recognize and address the pressing issue of deteriorating water quality in coastal rivers [2]. To protect river water quality, maintain the integrity of coastal ecosystems, and ensure sustainable human development, effective management and protection measures must be implemented [3]. Advanced technological means and scientific methods should be employed to strengthen water quality monitoring, early warning systems, and governance capabilities [4].

Traditional river water quality monitoring and warning technology relies on theoretical models that encompass physical, chemical, and biological processes [5]. These models describe and predict changes in water quality parameters by establishing mathematical equations. Mechanism models typically consider factors such as water flow velocity, flow rate, water quality parameters, as well as the transport and transformation of pollutants [6]. Common mechanism models include hydrodynamic models, water quality models, and ecological models. Traditional water-quality-monitoring and early warning technologies based on mechanism models offer certain advantages [7]. They are grounded in a profound understanding of hydrology, hydrodynamics, water quality, and ecological processes, thereby exhibiting high interpretability and reliability [8]. These models enable quantitative prediction and analysis of water quality variations, facilitating the assessment of the health of the water environment and the formulation of strategies to improve water quality [9]. However, traditional mechanism models also possess certain limitations. Firstly, they typically require extensive input data and parameters, including flow rate, rainfall, sediment characteristics, etc., which entail complex data acquisition and processing [10]. Secondly, establishing and calibrating the model necessitate deep professional knowledge and a substantial volume of measured data, demanding high technical expertise [11]. Moreover, the representation of complex water environments and ecosystems by mechanism models may involve simplifications and idealizations that fail to fully capture the complexity of real-world situations.

In recent years, there has been extensive research and applications of machine-learning-based technology for predicting river water quality [12]. In the context of river water quality prediction, machine learning utilizes a large amount of historical water quality data to construct accurate prediction models and enable early warning. This technology offers several advantages [13]. Firstly, it facilitates real-time and continuous monitoring and prediction of water quality, enhancing the responsiveness and effectiveness of water quality management. Secondly, machine learning models can automatically learn and adapt to the complex relationships within water quality data, resulting in more-accurate predictions [14]. Additionally, these models can incorporate other environmental factors and meteorological data, thereby improving the accuracy and reliability of water quality prediction. However, machine learning methods also face challenges and limitations in predicting river water quality. Issues such as data quality and missing data can impact the model's performance [15]. Moreover, training and parameter selection require a certain level of professional knowledge and experience. Additionally, the interpretability of the model is relatively low, making it difficult to interpret the predicted results. Therefore, further research and improvement are necessary to enhance the effectiveness and reliability of machine learning in river water quality prediction.

The entropy weighting method is an information-theory-based approach used to evaluate the information content and importance of features [16]. By computing the entropy value of features, the purity and discriminability of the features can be measured [17]. By combining the Pearson correlation coefficient with the entropy weighting method, the correlation and information content of the features can be comprehensively considered, avoiding over-reliance on a single criterion for feature selection. This approach enables a more-comprehensive evaluation of the contribution and importance of features, reducing subjectivity and uncertainty. By balancing different factors, features with higher correlation and greater information content can be selected, thereby improving the accuracy and stability of feature selection. Both the Pearson correlation coefficient and the entropy weighting method are relatively simple and intuitive approaches, making them easy to understand and interpret [18]. By integrating them into feature selection in machine learning, feature-selection results with higher interpretability and practicality can be obtained. This enhances the transparency and reliability of the feature-selection process, helping decision-makers understand the importance and contribution of features while improving the performance and interpretability of the model [19].

The objective of this study was to utilize machine learning techniques, in conjunction with the entropy weight method and Pearson correlation coefficient method as feature-selection methods, to achieve high-precision prediction of major water quality indicators, such as Dissolved Oxygen (DO), Ammonia Nitrogen (NH_3-N), Total Phosphorus (TP), and Total Nitrogen (TN). The models considered in this study encompass Long Short-Term Memory (LSTM), Support Vector Machine (SVM), Multilayer Perceptron (MLP), Random Forest (RF), and XGBoost. LSTM, as a variant of the Recurrent Neural Network (RNN) suitable for processing time series data, exhibited superior performance in predicting water quality changes. By inputting historical water quality data as time series, the LSTM model can be used to effectively capture long-term dependencies and accurately forecast future trends in water quality changes. Furthermore, other machine learning models offer distinct advantages and applicability in water quality prediction, enabling the selection of appropriate models based on specific requirements.

2. Materials and Methods

2.1. Data Acquisition

The data used for this research were sourced from the China Environmental Monitoring General Station, specifically from the monitoring point at Shijiaoju Section in the Pearl River Basin. The dataset comprises water quality measurements collected at four-hour intervals, covering the time period from 8 November 2020 to 28 February 2023. In total, there are 5058 samples in this dataset, encompassing 9 water quality parameters: Ammonia Nitrogen (NH_3-N), water Temperature (Temp), pH, Dissolved Oxygen (DO), the permanganate index ($KMnO_4$, Total Phosphorus (TP), Total Nitrogen (TN), Conductivity (Cond), and Turbidity (Turb).

2.2. Data Preprocessing

Data preprocessing is a vital step in machine learning, encompassing various tasks such as handling outliers, missing values, and data normalization [20]. For this study, historical monitoring data were collected, including water quality indicators such as temperature, pH, the potassium permanganate index, dissolved oxygen, ammonia nitrogen, total phosphorus, total nitrogen, turbidity, and conductivity. Firstly, outlier detection was performed on the data. Outliers can arise due to sensor malfunctions, human errors, or other factors, resulting in abnormal data points. Statistical analysis is commonly employed for outlier detection, involving calculations of the mean and standard deviation to identify values significantly deviating from the mean. If outliers are detected, they can be treated as missing values or corrected based on the specific circumstances [21]. Next, the focus was placed on addressing missing values in the data. Linear interpolation was utilized in this study to fill in the missing values. Linear interpolation estimates the missing values by considering the linear relationship between known data points. Specifically, for time series data, the observed values of the preceding and succeeding time points are used to perform linear interpolation and estimate the missing values. Suppose we want to estimate the missing value x between the known data points x_1 and x_2, corresponding to observed values y_1 and y_2, respectively. The linear interpolation formula for estimating the value y is as follows:

$$y = y_1 + (x - x_1)\frac{y_2 - y_1}{x_2 - x_1} \tag{1}$$

Here, $(x - x_1)$ represents the offset of x relative to x_1 and $((y_2 - y_1)/(x_2 - x_1))$ represents the slope from x_1 to x_2. By multiplying the offset by the slope and adding it to y_1, the missing value y can be estimated [22].

The advantages of linear interpolation include its simplicity, ease of use, and the ability to produce reasonably accurate estimation results in certain cases [23]. However, linear interpolation also has limitations. Firstly, it assumes a linear relationship between data points, which may not hold true in all situations. Secondly, it requires a high density of data points, and sparse or unevenly distributed data may lead to inaccurate estimates. Additionally, linear interpolation cannot capture nonlinear trends or special patterns in the data [24]. Therefore, when applying linear interpolation, it is crucial to assess the characteristics and validity of the data within the specific context and consider the suitability of alternative interpolation methods [25].

Lastly, data normalization will be performed to eliminate dimensional differences among different water quality indicators [26]. The chosen method was min–max normalization, which linearly transforms the data to a specific range, typically [0, 1] or [−1, 1], ensuring that feature variables have similar scales. Min–max normalization can be calculated using the following formula:

$$X_N = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2}$$

Here, X_N represents the normalized value, X represents the original value, X_{min} represents the minimum value and X_{max} represents the maximum value. This formula maps the original data to a range between 0 and 1.

By following these data preprocessing steps, we can obtain cleaned and prepared data suitable for the subsequent feature selection, training, and prediction of machine learning models. This process will enhance the accuracy and stability of the models and provide a reliable foundation for predicting river water quality.

2.3. Feature Variable Selection

2.3.1. Entropy Weighting Method

The entropy weight method is a technique employed to determine the weights of multiple indicators. It utilizes the concept of entropy to measure the uncertainty or diversity of indicators by computing their information entropy [27]. This method finds widespread application in multi-indicator decision-making, evaluation, and ranking, aiding in addressing challenges associated with trade-offs and optimization among multiple indicators [28]. The steps involved in calculating weights using the entropy weighting method based on information entropy are as follows:

- Calculate information entropy: Compute the information entropy for each indicator. The information entropy quantifies the uncertainty or diversity of an indicator, with higher entropy values indicating greater diversity. The calculation formula for the information entropy is as follows:

$$H(X) = -\sum (P_i \times \log_2 P_i) \tag{3}$$

  Here, $H(X)$ represents the information entropy of the indicator and P_i represents the normalized value of the indicator.
- Calculate information weight: Determine the information weight for each indicator based on its information entropy. The calculation formula for the information weight is as follows:

$$W_i = (1 - H(X_i)) \tag{4}$$

  Here, W_i denotes the information weight of indicator X_i and $H(X_i)$ represents the information entropy of indicator X_i.

The information weight reflects the level of information contained within an indicator. A higher information weight signifies a greater impact of the indicator on the decision outcome. Hence, information weights can be utilized to assess the significance of indicators and their contributions to the decision-making process. By calculating the information entropy and information weights for each indicator, a weight vector can be derived for subsequent tasks such as multi-indicator decision-making, evaluation, or optimization [29]. Incorporating information weights assists decision-makers in making informed trade-offs and selections among indicators, thus enhancing the accuracy and credibility of decisions. However, it is essential to recognize that information weights solely consider the diversity and uncertainty of indicators, disregarding their interrelationships. Therefore, in practical applications, it is crucial to consider additional methods or domain knowledge to comprehensively assess the indicators [30].

According to the analysis presented in Table 1 and Figure 1, several variables stand out with relatively higher weights. Specifically, NH_3-N, DO, $KMnO_4$, TP, Cond, and Turb exhibit higher weights compared to other variables. Among these, NH_3-N, Cond, and Turb emerge as particularly influential indicators, indicating their significance and stronger influence on the decision outcome. Conversely, Temp, pH, and TN display relatively lower weights, implying their diminished importance and weaker impact on the decision outcome.

Table 1. Results of information weight calculation using entropy weight method.

Variables	Mean	Standard Deviation	CV Coefficient [1]	Weight [2]
NH_3-N	0.484	0.464	0.958	0.232
Temp	24.504	4.763	0.194	0.047
pH	7.554	0.540	0.072	0.017
DO	7.887	3.642	0.462	0.112
$KMnO_4$	3.715	1.591	0.428	0.104
TP	0.110	0.047	0.427	0.103
TN	3.274	0.672	0.205	0.050
Cond	1264.521	979.968	0.775	0.188
Turb	49.761	30.356	0.610	0.148

[1] CV coefficient = standard deviation/mean. [2] The weights are calculated by normalizing the CV coefficients.

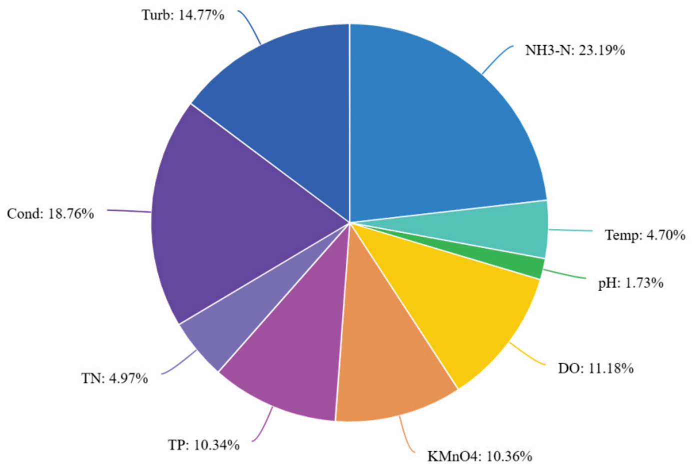


Figure 1. Information weight distribution map (according to Table 1).

2.3.2. Pearson Correlation Coefficient Method

The Pearson correlation coefficient is a statistical measure utilized to evaluate the linear correlation between two continuous variables [31]. It provides information about the strength and direction of the linear relationship between the variables, making it a commonly employed method for feature variable selection and evaluation [32]. The coefficient, denoted as "r", ranges from -1 to 1. A value of r = 1 indicates a perfect positive linear relationship, while r = -1 indicates a perfect negative linear relationship. A value of r = 0 suggests no linear relationship, indicating no correlation between the variables. The calculation formula is as follows:

$$R = \frac{\sum(X_i - X_{mean})(Y_i - Y_{mean})}{NX_{std}Y_{std}} \tag{5}$$

Here, R represents the Pearson correlation coefficient, X_i and Y_i denote the values of the variables in the observation matrix, X_{mean} and Y_{mean} represent the means of the variables, X_{std} and Y_{std} represent the standard deviations of the variables, and N denotes the number of observations in the sample.

According to Figure 2, significant correlation coefficients were observed between NH_3-N and pH, DO, $KMnO_4$, TP, and TN, indicating a substantial relationship. DO exhibited significant correlation coefficients with NH_3-N, temperature, pH, $KMnO_4$, TP, conductivity, and turbidity, indicating significant relationships. Similarly, TP showed significant correlation coefficients with NH_3-N, temperature, pH, DO, $KMnO_4$, TN, conductivity, and turbidity, indicating significant relationships. Likewise, TN exhibited significant correlation coefficients with NH_3-N, temperature, pH, $KMnO_4$, TP, conductivity, and turbidity, indicating significant relationships.

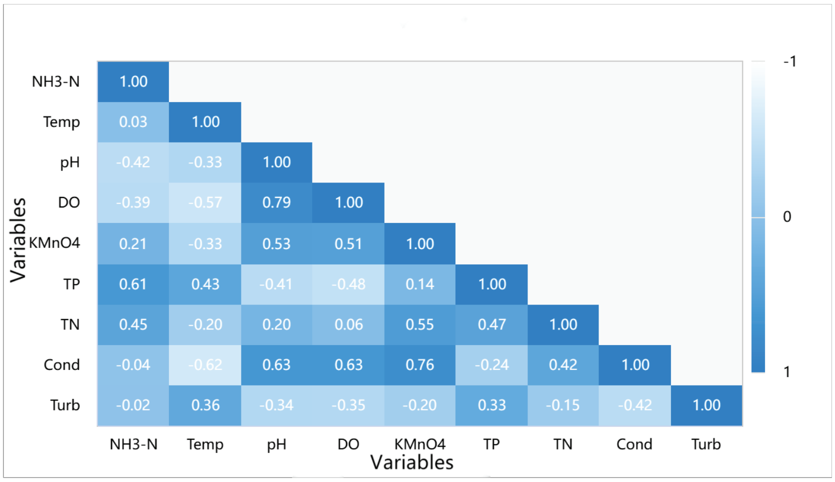


Figure 2. Visualization of Pearson correlation.

2.3.3. Comprehensive Weight Method

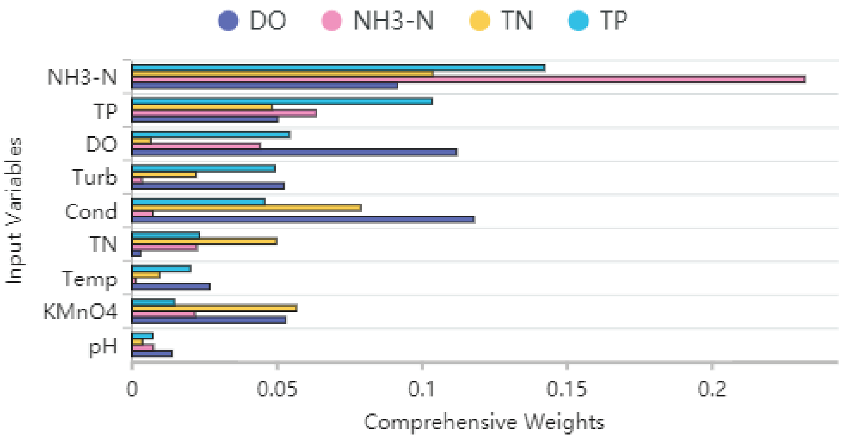
The comprehensive weight method is an approach for selecting feature variables that combines the Pearson correlation coefficient method and the entropy weight method. It aims to evaluate and select feature variables by calculating their comprehensive weights, which are obtained by multiplying the Pearson correlation coefficient value of each feature variable with its corresponding information weight. The formula for calculating the comprehensive weight is as follows:

$$V_{CW} = V_{PCC} \times V_{IW} \tag{6}$$

Here, V_{CW} represents the comprehensive weight, V_{PCC} represents the Pearson correlation coefficient, and V_{IW} represents the information weight. A higher comprehensive weight indicates a greater importance and relevance of the feature variable in predicting the target variable.

The comprehensive weight method offers the advantage of considering both linear relationships and importance factors, resulting in a more-comprehensive evaluation and selection of feature variables. This, in turn, improves the accuracy and stability of the feature-selection process. Based on the comprehensive weight calculation results presented in Table 2 and Figure 3, the following input variables were selected for predicting the respective target variables:

- Dissolved Oxygen (DO) prediction: DO, NH_3-N, Temp, pH, $KMnO_4$, TP, Cond, and Turb;
- Ammonia Nitrogen (NH_3-N) prediction: NH_3-N, DO, $KMnO_4$, TP, and TN;
- Total Nitrogen (TN) prediction: TN, NH_3-N, $KMnO_4$, TP, Cond, and Turb;
- Total Phosphorus (TP) prediction: TP, NH_3-N, Temp, DO, $KMnO_4$, TN, Cond, and Turb.



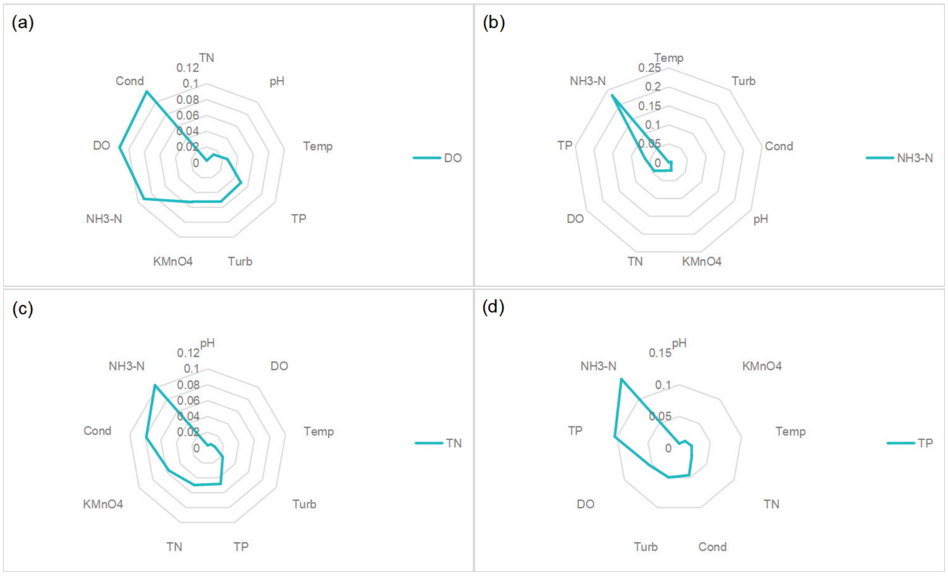


Figure 3. The comprehensive weight of input variables under each output variable: (**a**) DO and comprehensive weights of its corresponding input variables; (**b**) NH_3-N and comprehensive weights of its corresponding input variables; (**c**) TN and comprehensive weights of its corresponding input variables; (**d**) TP and comprehensive weights of its corresponding input variables.

Table 2. Summary of comprehensive weight results (The comprehensive weight values of different input variables for predicting the target variable).

Variables	**DO**	**NH_3-N**	**TN**	**TP**
NH_3-N	0.091	0.232	0.104	0.142
Temp	0.027	0.001	0.009	0.020
pH	0.014	0.007	0.003	0.007
DO	0.112	0.044	0.006	0.054
$KMnO_4$	0.053	0.022	0.057	0.015
TP	0.050	0.063	0.048	0.103
TN	0.003	0.022	0.050	0.023
Cond	0.118	0.007	0.079	0.046
Turb	0.052	0.003	0.022	0.049

These selected input variables were determined based on their respective comprehensive weights, which consider both the Pearson correlation coefficient and information weight. By including these variables in the prediction models, it is expected to enhance the accuracy and reliability of the predictions for Dissolved Oxygen (DO), Ammonia Nitrogen (NH_3-N), Total Nitrogen (TN), and Total Phosphorus (TP).

2.4. Models

2.4.1. Support Vector Machine

Support Vector Machine (SVM) is a popular supervised learning algorithm utilized for classification and regression tasks. Its objective is to discover an optimal hyperplane that effectively separates different classes of samples while maximizing the margin between them, thereby achieving robust generalization performance [33]. SVM achieves this by mapping the samples into a high-dimensional feature space and identifying the hyperplane that maximizes the margin within this space. This hyperplane is defined as the one with the greatest distance to the nearest samples of different classes, referred to as support vectors, which play a crucial role in determining the hyperplane's position.

SVM demonstrates high accuracy and generalization performance, particularly for small-sized datasets. It exhibits robustness against noise and outliers and is well-suited for handling high-dimensional data. Moreover, the decision function of SVM is based on the support vectors, which provide valuable insights into the data distribution and decision boundary, thus offering interpretability to some extent. However, SVM also has certain limitations. It can be computationally slow when applied to large-scale datasets, and its performance may degrade on high-dimensional data or when dealing with imbalanced classes. Additionally, the selection of appropriate kernel functions and tuning of the related parameters are important considerations when utilizing SVM [34].

2.4.2. Multilayer Perceptron

Multilayer Perceptron (MLP) is a neural network with an input layer, multiple hidden layers, and an output layer. It uses weighted connections and nonlinear activation functions to process data. The network is trained using the backpropagation algorithm, updating weights to minimize prediction errors. MLP excels in modeling complex patterns and can be adjusted to fit different task complexities [35]. However, training and prediction times may be longer for large-scale or high-dimensional data. The model's performance depends on factors such as activation functions, the architecture, and the hyperparameters [36]. Multiple metrics should be used for evaluation, considering the model structure, feature selection, and data distribution. Enhancements can be made through adjustments, optimization, additional features, or alternative algorithms [37].

2.4.3. Random Forest

Random Forest is an ensemble learning method that constructs multiple weak learners based on decision trees. It combines the predictions of individual trees through voting or averaging to make the final predictions. In each node of the decision tree, Random Forest considers only a random subset of features for splitting. This selective feature consideration reduces the correlation between trees, leading to increased model diversity. To create diverse decision trees, multiple training sets are generated using bootstrap sampling. This process involves randomly selecting samples with replacement from the original training set, enabling the training of different decision trees [17]. The utilization of bootstrap sampling enhances model diversity and mitigates overfitting. Random Forest generates predictions by aggregating the collective decisions of multiple decision trees. In classification tasks, the prediction is determined by the majority class obtained through voting, while in regression tasks, the average prediction of the multiple decision trees is used. Random Forest models are highly proficient in handling high-dimensional and large-scale data, demonstrating their suitability for complex nonlinear relationships [33]. Furthermore, they exhibit robustness in the presence of missing values and outliers.

2.4.4. Extreme Gradient Boosting

Extreme Gradient Boosting (XGBoost) is an algorithm developed based on the gradient boosting decision trees approach. It enhances model accuracy and efficiency by incorporating regularization techniques and parallel computing. XGBoost leverages the gradient boosting algorithm, which iteratively trains a sequence of decision trees to progressively enhance the predictive model's performance. Each tree is trained to rectify the prediction errors made by the preceding tree, gradually aligning with the negative gradient of the objective function. To address the risk of overfitting, XGBoost employs various regularization techniques, including L1 and L2 regularization. Additionally, constraints on tree depth and leaf weights are applied to manage the model's complexity and prevent overfitting. These measures collectively contribute to improving the overall performance and generalization capability of the XGBoost algorithm [38].

2.4.5. Long Short-Term Memory

Long Short-Term Memory (LSTM) is a specialized variant of Recurrent Neural Networks (RNNs) that excels in processing time series data. Unlike conventional RNNs, LSTM incorporates gating mechanisms that effectively capture and retain long-term dependencies [39]. The core of an LSTM network comprises three essential gate units: the forget gate, the input gate, and the output gate. These gate units regulate the flow and manipulation of information through learnable weights, thereby controlling the input, output, and memory processes. By selectively forgetting, updating, and outputting information, LSTM enables the model to effectively retain and utilize long-term data information, effectively addressing the challenge of long-term dependencies encountered in traditional RNNs. The gating mechanisms employed by LSTM also address the issues of vanishing and exploding gradients, ensuring smooth gradient propagation over extended time intervals [40]. LSTM exhibits versatility in handling diverse input and output types, including univariate and multivariate time series data, as well as text data. It offers flexibility in adjusting input and output dimensions and possesses strong representational capabilities [12]. LSTM has achieved remarkable success in various domains such as natural language processing, speech recognition, machine translation, and time series prediction. Consequently, LSTM finds widespread application in modeling and prediction tasks across a wide range of practical problems.

2.4.6. Grid Search (GridSearchCV)

In this study, the Grid Search technique (GridSearchCV) was utilized to fine-tune the parameters of the machine learning model and identify the optimal combination of parameters, thereby enhancing the model's performance and predictive capability. Grid search systematically explores all potential parameter combinations within the defined parameter ranges, extensively investigating the parameter space to determine the best configuration [41]. In practical applications, machine learning models often possess adjustable parameters such as the learning rate, regularization parameter, and tree depth, which significantly impact model performance. By employing grid search, we can methodically evaluate the performance of diverse parameter combinations and identify the optimal set of parameters to achieve the best model performance. The primary advantage of this approach lies in its comprehensiveness and intuitive nature. It eliminates the need for intricate mathematical derivations or optimization algorithms; instead, it involves specifying the parameter value ranges and exhaustively iterating through all feasible parameter combinations. Consequently, grid search is straightforward to understand and implement, providing interpretable results that facilitate a clear understanding of how different parameter combinations affect the model's performance [42].

2.5. Model Evaluation

Model evaluation refers to assessing the performance of a trained model to understand how well it performs on unseen data. The following Table 3 provides the model evaluation metrics used in this study.

Table 3. Model evaluation methods.

Metric	Description	Formula
Mean-Squared Error (MSE)	The MSE measures the average difference between predicted values and true values in regression models. It is calculated as the mean of the squared differences between the predicted and true values.	MSE = $(1/n) * \Sigma(y_{pred} - y_{true})^2$
Root-Mean-Squared Error (RMSE)	The RMSE is the square root of the MSE and provides a measure of the average error between predicted and true values. It is consistent with the scale of the true values, making it easier to interpret.	RMSE = sqrt (MSE)
Nash–Sutcliffe Efficiency (NSE)	The NSE is a metric commonly used in hydrological models to evaluate the fit between model predictions and observed values. It considers the ratio of the sum of squared differences to the variance of observed values, subtracted from 1.	NSE = $1 - (\Sigma(y_{pred} - y_{true})^2/\Sigma(y_{true} - y_{mean})^2)$
Coefficient of Determination (R^2 Score)	The R^2 score evaluates the model's ability to explain the variance in the observed data. It calculates the ratio of the sum of squared differences between predicted and observed values to the total variance of the observed values.	R^2 Score = $1 - (\Sigma(y_{pred} - y_{true})^2/\Sigma(y_{true} - y_{mean})^2)$

The evaluation metrics, the MSE and RMSE, are considered favorable when they have smaller values, while the NSE and R^2 score should approach 1. These metrics provide insights into the predictive performance of the model and facilitate the comparison of different models to select the most-suitable one [43].

Using GridSearchCV, we conducted a grid search to identify the optimal model structures and parameter combinations for each model. We also integrated the input variable sets selected by the comprehensive weight method to make separate predictions for Dissolved Oxygen (DO), Ammonia Nitrogen (NH_3-N), Total Nitrogen (TN), and Total Phosphorus (TP) in the surface water. Moreover, we divided the dataset into a training set and a test set. The first 4552 samples were allocated to the training set, while the remaining 506 samples were designated as the test set. The training set was used to train the models, and the test set was employed to evaluate the model's performance and assess its effectiveness in making predictions.

To enhance the stability and reliability of predictions by mitigating the impact of random factors, the model prediction experiments were conducted in parallel for 10 sets. The final evaluation result for the model's prediction parameter was obtained by calculating the average of the R^2 values, MSE values, RMSE values, and NSE values from these 10 parallel experiments. This approach allows for a comprehensive assessment of the model's performance across diverse experiments, reducing the potential influence of random errors associated with a single experiment [44]. Running multiple experiments yields a larger set of data points, which enhances the statistical significance of the evaluation results and provides a more-comprehensive and -accurate evaluation of the model's performance.

3. Results and Discussion

3.1. Data Statistics

Based on the analysis of Table 4 and Figure 4, insights can be obtained regarding the distribution of the data for each variable. Notably, Dissolved Oxygen (DO) and Ammonia Nitrogen (NH_3-N) exhibited substantial fluctuations throughout the year, while Total Nitrogen (TN) and Total Phosphorus (TP) demonstrated relatively smaller fluctuations. The standard deviations for Dissolved Oxygen (DO), Ammonia Nitrogen (NH_3-N), and Total Nitrogen (TN) were calculated as 3.642, 0.464, and 0.672, respectively, indicating a higher level of data variability associated with these variables. Conversely, the standard deviation for Total Phosphorus (TP) was determined to be 0.047, suggesting a lower level of data variability.

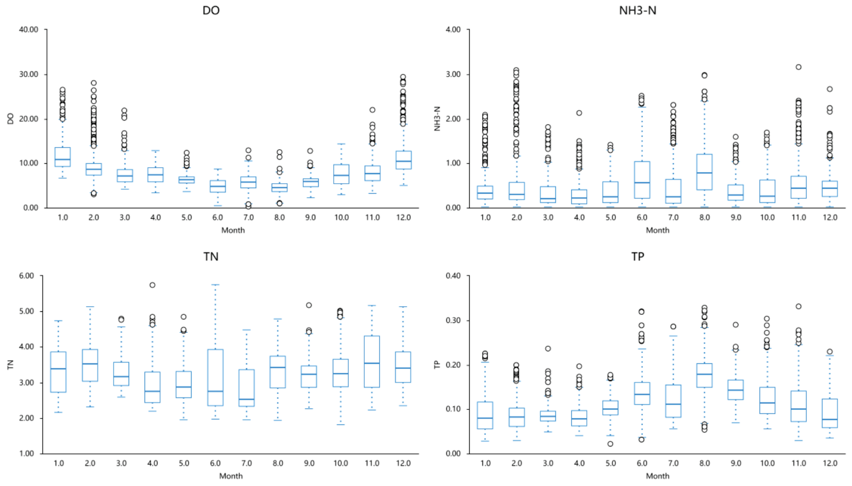


Figure 4. Fluctuationof water quality variables (The black circle in figure body represents Outlier).

Table 4. Statistics of various water quality variables' data.

Variables	Sample Size	Min	Max	Mean	Standard Deviation	Median
DO	5058	0.323	29.370	7.887	3.642	7.160
NH_3-N	5058	0.025	3.162	0.484	0.464	0.352
TN	5058	1.811	5.755	3.274	0.672	3.243
TP	5058	0.022	0.331	0.110	0.047	0.101

Dissolved Oxygen (DO) represents the amount of oxygen dissolved in water and is influenced by various factors, with temperature being one of the primary influences. Generally, as temperature increases, the concentration of dissolved oxygen decreases. This relationship implies that higher temperatures lead to lower dissolved oxygen concentrations, while lower temperatures result in higher dissolved oxygen concentrations. The summer season, typically occurring from June to September in most regions, is associated with elevated temperatures. Consequently, water temperature rises during this period, leading to a decrease in dissolved oxygen concentration and lower values of dissolved oxygen. Conversely, in January and December, lower temperatures prevail, causing the water temperature to be relatively colder. As a result, the dissolved oxygen concentration increases, yielding higher values of dissolved oxygen. Although other factors such as oxygen supply, environmental conditions in the water (e.g., plant growth, water body mixing), and meteorological changes can influence the dissolved oxygen concentration, temperature remains the primary factor among them.

Ammonia Nitrogen (NH_3-N) exhibits relatively higher concentrations from June to August each year due to several reasons. Firstly, the summer season corresponds to the peak period of biological activity in water, involving microorganisms, algae, and bacteria. These organisms absorb nutrients, including ammonia compounds, from the water for growth and metabolism, contributing to the release of ammonia nitrogen. Therefore, increased biological activity during summer results in higher concentrations of ammonia nitrogen. Secondly, higher air temperatures during summer lead to elevated water temperatures. Ammonia nitrogen's solubility is positively correlated with water temperature, meaning that higher temperatures facilitate the dissociation of ammonia nitrogen molecules from solids or organic matter, increasing its solubility in water. Additionally, the concentration of ammonia nitrogen can be influenced by factors such as sediment release, agricultural and urban discharges, rainfall, and flow variations.

The concentration of Total Phosphorus (TP) shows relatively small and stable fluctuations throughout the year due to several factors. Stable input sources, such as consistent surface runoff, groundwater, or controlled sediment release, contribute to smaller fluctuations in total phosphorus concentration. Additionally, biological absorption and deposition processes play a role. Organisms present in the water, such as phytoplankton and algae, absorb total phosphorus and convert it into biomass. Moreover, some total phosphorus can also deposit into sediment. These processes help stabilize the concentration of total phosphorus and reduce fluctuations. Environmental conditions in the water, such as light intensity, temperature, and dissolved oxygen levels, can also influence total phosphorus concentration. When these conditions remain relatively stable without significant changes, the biological transformation and sedimentation processes related to total phosphorus also remain stable, leading to smaller fluctuations in total phosphorus concentration.

The concentration of Total Nitrogen (TN) remains relatively high from June to September each year. Several factors contribute to this observation. Firstly, the summer season is associated with increased nutrient inputs due to vigorous plant growth. Factors such as fertilization in farmlands and green spaces, irrigation in farmlands, and rainfall contribute to higher nutrient (including nitrogen) inputs into water bodies, resulting in relatively higher concentrations of total nitrogen during summer. Additionally, summer is characterized by abundant sunlight and higher temperatures, providing favorable conditions for the growth of algae and phytoplankton in the water. This enhances the mixing of nitrogen-rich water from the bottom layer with surface water, contributing to an increase in total nitrogen concentration.

3.2. Performance Comparison of Models

Based on the findings presented in Table 5 and Figure 5, the predictive performance of various machine learning models on Dissolved Oxygen (DO) content was evaluated, leading to a comparative analysis and discussion of each model's performance in predicting water quality.

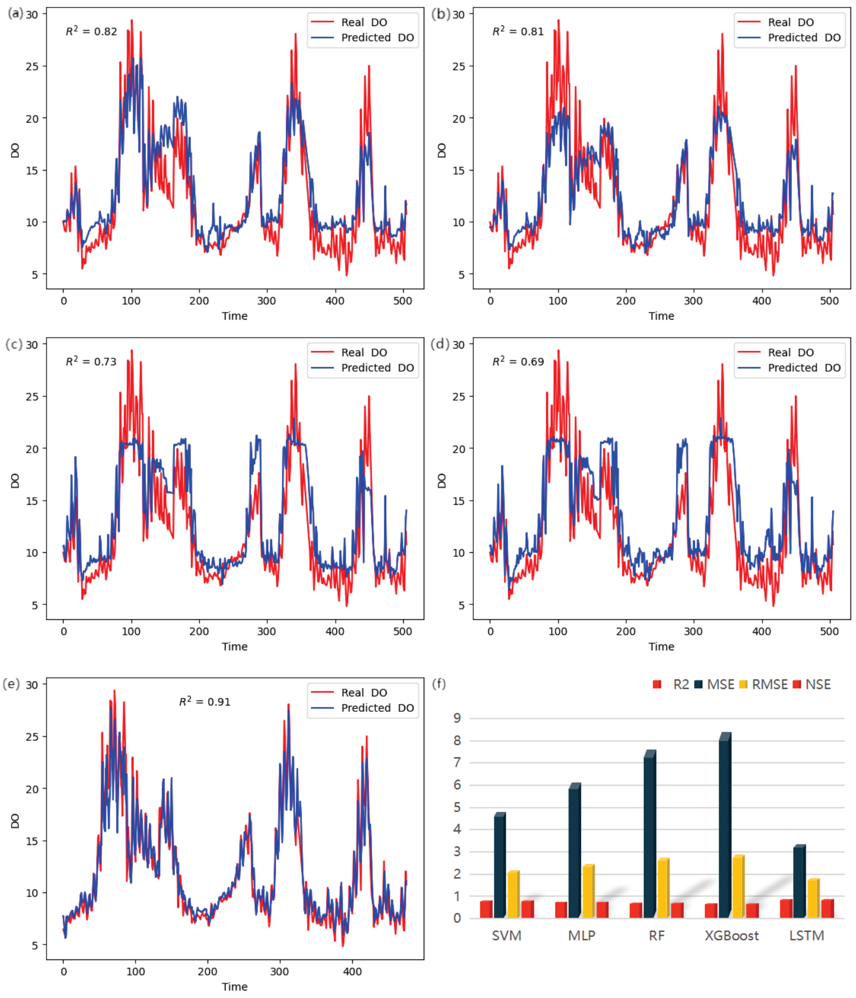


Figure 5. (**a**) Comparison of real values and predicted values for the SVM model; (**b**) Comparison of real values and predicted values for the MLP model; (**c**) Comparison of real values and predicted values for the RF model; (**d**) Comparison of real values and predicted values for the XGBoost model; (**e**) Comparison of real values and predicted values for the LSTM model; (**f**) Performance evaluation graph for each model.

Table 5. Performance evaluation of various models for dissolved oxygen prediction.

Models	R^2	MSE	RMSE	NSE
SVM	0.820	4.816	2.195	0.823
MLP	0.775	6.128	2.473	0.775
RF	0.720	7.613	2.759	0.720
XGBoost	0.690	8.403	2.899	0.691
LSTM	0.882	3.361	1.827	0.877

The Support Vector Machine (SVM) model achieved notable results with an R^2 score of 0.820, an MSE of 4.816, and an RMSE of 2.195 for DO prediction. These values indicate a relatively small average prediction error, showcasing the model's ability to accurately predict the DO values. Moreover, the NSE value of 0.823 suggests that the model outperformed predictions based on the mean value alone. The SVM model's proficiency in handling nonlinear relationships is crucial since DO levels are influenced by intricate nonlinear associations with multiple factors. By mapping the input space to a higher-dimensional feature space using a kernel function, the SVM model achieves improved fitting of nonlinear relationships. Additionally, the model's decision boundary is determined by maximizing the margin, highlighting its strong generalization capabilities. This capability enables the SVM model to maintain good prediction performance when confronted with new samples, effectively avoiding overfitting or underfitting issues. Furthermore, the SVM model demonstrates resilience in the presence of noisy or outlier-laden data by selectively considering support vectors, thereby enhancing prediction accuracy. These characteristics collectively contributed to the model's commendable performance in predicting the DO levels.

On the other hand, the Multilayer Perceptron (MLP) model exhibited slightly larger prediction errors compared to the SVM model. The MLP model achieved an R^2 score of 0.775, an MSE of 6.128, and an RMSE of 2.473 for the DO prediction. While the prediction error was slightly larger than that of the SVM model, the NSE value of 0.775 suggests that the predicted results were comparable to those obtained using the average value. The performance of the MLP model heavily relies on the design of its network structure. In cases where the network structure is not suitable, the model may struggle to capture the complex nonlinear relationship of DO content accurately. Moreover, the MLP model typically performs better when applied to large-scale and high-quality datasets. Insufficient training data or the presence of numerous noises or outliers may adversely affect the model's performance. Unlike the SVM model, the MLP model involves numerous hyperparameters that require optimization, such as the number of hidden layer nodes, the selection of the activation function, and the learning rate. The improper selection or insufficient tuning of these hyperparameters can negatively impact the predictive performance of the model.

The Random Forest (RF) model demonstrated larger prediction errors compared to the SVM and MLP models, with an R^2 score of 0.720, an MSE of 7.613, and an RMSE of 2.759 for DO prediction. The NSE value of 0.720 indicates that the RF model's predictions were slightly inferior to the baseline prediction using the mean value. The performance of the RF model in predicting DO levels can be influenced by the presence of class imbalance in the training data. Specifically, the distribution of DO levels showed an imbalance, with fewer samples in the 0–5 and 15–20 concentration ranges and more samples in the 5–15 concentration range. This imbalance can result in poorer performance of the RF model when predicting the DO levels within the underrepresented concentration ranges. Additionally, the performance of the RF model heavily relies on the number and depth of the decision trees. Opting for a small number of trees or shallow trees may hinder the model's ability to capture the complex relationships in the DO levels, leading to larger prediction errors. Thus, it is essential to carefully select the number and depth of the decision trees to enhance the RF model's predictive performance.

The XGBoost model exhibited larger prediction errors compared to the other models, with an R^2 score of 0.690, an MSE of 8.403, and an RMSE of 2.899 for DO prediction. In comparison to the other models, the prediction error was relatively high. The NSE value of 0.691 indicates that the XGBoost model's predictions were comparable to the baseline prediction using the mean value. Similar to the Random Forest (RF) model, the performance of the XGBoost model can be influenced by class imbalance in the training data. If there is an imbalance in the distribution of the DO levels, with certain concentration ranges having fewer samples, it can impact the model's ability to predict DO levels within those ranges. The XGBoost model possesses strong capabilities in capturing interactions and nonlinear relationships among features. When the variations in DO levels are complex and driven by intricate interactions or nonlinear relationships, the XGBoost model may require a larger number of trees or deeper trees to effectively capture these relationships and enhance the prediction performance. Consequently, the selection of an appropriate number and depth of the trees becomes crucial for achieving improved performance with the XGBoost model.

The LSTM model demonstrated high prediction accuracy for the DO values, with an R^2 score of 0.882, an MSE of 3.361, and an RMSE of 1.827. These values indicate that the LSTM model had a small average prediction error and performed with high accuracy in predicting the DO values. The NSE of 0.877 suggests that the model's predictions were superior to those obtained using the average value. The LSTM model is a recurrent neural network model specifically designed for processing time series data. Given the time dependence of the DO content, the LSTM model can effectively capture the dynamic changes and trends, thereby improving the prediction accuracy. Through its gating mechanisms and memory units, the LSTM model is capable of retaining and updating important information while disregarding irrelevant information. This long-term memory capability enables the LSTM model to capture the long-term dependence of the DO content, resulting in improved prediction accuracy. Moreover, the LSTM model has the ability to automatically learn and extract features relevant to the DO content prediction. It can adaptively adjust the weight of features to maximize the extraction of useful information. This feature extraction capability contributes to the model's enhanced predictive performance for the DO content. In summary, the LSTM model performed well in predicting the Dissolved Oxygen (DO) content due to its effective processing of time series data, long-term memory ability, feature extraction ability, and capacity to capture the sequential nature and patterns of the DO content. These factors collectively enable the LSTM model to achieve high prediction accuracy and demonstrate relatively good performance in DO content prediction.

3.3. Comprehensive Prediction Performance of the LSTM Model

The LSTM model's architecture comprises of four LSTM layers, each consisting of 56 neurons. To mitigate overfitting, a dropout layer was added after each LSTM layer with a dropout rate of 0.2. The final layer of the model was a dense layer containing only one neuron, responsible for generating the prediction results. The input variable sequence length was set to 30, implying that the model uses water quality data from the previous 5 days to predict the water quality parameter for the subsequent time period. Parameter optimization for the model was performed using GridSearchCV, leading to the identification of the optimal parameter combination as follows: the batch size was 32; the epochs were 60; the optimizer was Adam.

Based on the findings presented in Table 6 and Figure 6, the LSTM model demonstrated exceptional predictive performance for key variables such as Dissolved Oxygen (DO), Ammonia Nitrogen (NH_3-N), Total Nitrogen (TN), and Total Phosphorus (TP). Specifically, the R^2 values for these variables were 0.882, 0.830, 0.745, and 0.773, respectively. These values indicated a strong correlation between the predicted and actual values, highlighting the effectiveness of the LSTM model. Additionally, the corresponding MSE and RMSE values were relatively small, indicating low average prediction errors and the overall high performance of the LSTM model. The NSE values of 0.829, 0.745, and 0.763 further supported these results, surpassing the predictions based on the mean values for these variables.

Table 6. Performance evaluation of the LSTM model in predicting other water quality variables.

Variables	R^2	MSE	RMSE	NSE
DO	0.882	3.361	1.827	0.877
NH_3-N	0.830	5.614	2.330	0.829
TN	0.745	5.747	2.352	0.745
TP	0.773	5.683	2.332	0.763

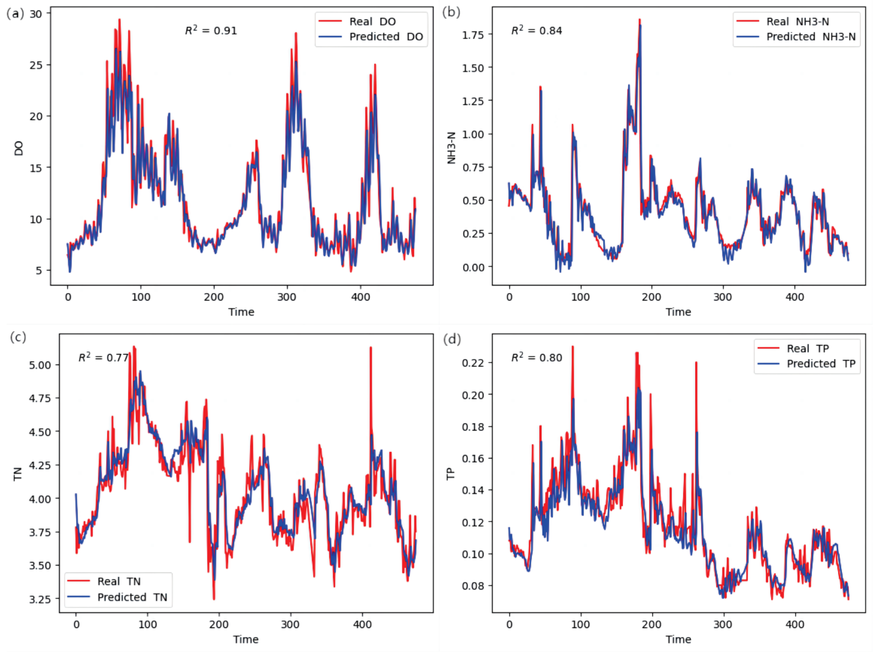


Figure 6. Comparison of LSTM model's actual and predicted values for other water quality parameters: (**a**) Comparison of true and predicted values of DO; (**b**) Comparison of true and predicted values of NH_3-N; (**c**) Comparison of true and predicted values of TN; (**d**) Comparison of true and predicted values of TP.

However, it is noteworthy that Figure 6c reveals a relatively weaker predictive performance of the LSTM model for TN content, particularly in capturing extreme values accurately. This observation can be attributed to several contributing factors. Firstly, the complexity of the model may not be adequate to capture the intricate nonlinear relationships and long-term dependencies within the data, particularly when predicting extreme values. Therefore, it is recommended to consider utilizing a more-sophisticated model structure or enhancing the model's capacity to improve its ability to predict extreme values. Secondly, improper feature selection could be another influential factor. Although a comprehensive weight method, along with the entropy weight method and Pearson correlation coefficient method, was employed for feature screening, the selection of individual feature variables may have been subjective. Specifically, when predicting TN content, the input features of the LSTM model may not effectively capture the characteristics necessary to handle extreme information. Thirdly, the presence of noise or outliers in the data can negatively impact the accurate prediction of extreme values by the model. Lastly, insufficient training could also contribute to the relatively weaker performance of the LSTM model. To enhance TN content prediction, it is advisable to increase the training sample size and extend the training duration to enable the model to fully capture extreme patterns. Insufficient training samples or a relatively short training period may hinder the model's ability to effectively capture extreme features. To address these potential limitations, future improvements can include increasing the training sample size, adjusting the model structure, optimizing the feature selection, and conducting longer training sessions to enhance the overall predictive performance of the LSTM model for TN content.

The LSTM model's proficiency in capturing dynamic features and trends in time series data significantly contributed to its accurate prediction of water quality variables. By leveraging its ability to learn temporal relationships and long-term dependencies within the sequence data, the LSTM model excelled in predicting future values of water quality variables. Furthermore, its capability to handle nonlinear relationships and complex temporal patterns further strengthened its performance in predicting water quality variables.

4. Conclusions

The primary objective of this study was to evaluate the predictive capabilities of various machine learning models for water quality parameters using the entropy weighting method. A comprehensive weighting method was proposed, which combines the entropy weighting method with the Pearson correlation coefficient method, for feature selection in water quality prediction. This method takes into account both the information entropy of the input features and their correlation with the target variable, effectively identifying the features possessing a significant impact on water quality variable prediction. The method offers valuable insights for subsequent water quality prediction modeling, including feature set selection, the reduction of redundant features, and the optimization of model performance.

Multiple machine learning models were investigated for their applicability in water quality prediction, the Support Vector Machine (SVM), Multilayer Perceptron (MLP), Random Forest (RF), XGBoost, and LSTM models. These models demonstrated varying capabilities in water quality prediction. SVM, in particular, exhibited good generalization performance and high prediction accuracy, specifically for the prediction of Dissolved Oxygen (DO). The MLP model, known for its strong nonlinear modeling capability, performed well in predicting DO and NH_3-N, explaining a significant proportion of the target variable's variance and exhibiting relatively small prediction errors.

In contrast, the RF model, despite its ability to handle high-dimensional data and complex relationships, showed relatively poor performance in water quality prediction. It displayed lower R^2 values and higher MSE and RMSE values, indicating larger prediction errors. This could be attributed to the model's limitations in capturing complex relationships and extreme values in water quality data, leading to decreased prediction accuracy. Similarly, the XGBoost model also exhibited relatively poor predictive performance, with lower R^2 values and higher MSE and RMSE values, indicating larger prediction errors. This might be due to the model's limited ability to capture complex relationship patterns and extreme values in the water quality data, resulting in lower prediction accuracy compared to the other models.

The LSTM model demonstrated excellent water quality prediction capabilities. As a recurrent neural network model designed to handle sequential data, LSTM possesses strong memory and long-term dependency modeling capabilities. In water quality prediction, the LSTM model effectively captured dynamic changes in time series data and consistently delivered outstanding predictive performance for various water quality parameters. Its high R^2 values and NSE values, along with low MSE and RMSE values, indicated small average prediction errors and significant improvements over simple mean value prediction methods.

In summary, the comprehensive weighting method that combines the entropy weighting method and the Pearson correlation coefficient method showed effectiveness in selecting a feature set for water quality prediction, enhancing the predictive performance of the models. Through comparative studies, the LSTM model emerged as the top-performing model for water quality prediction, accurately forecasting variations in different water quality variables in a stable manner. These research findings provide essential insights for water quality monitoring and management, assisting water quality management agencies in making informed decisions and devising effective management strategies. However, further research and applications are necessary to explore optimized feature-selection methods, improve machine learning models, and enhance the accuracy and reliability of water quality prediction.

Author Contributions: Conceptualization, X.W., Y.L. (Yanchun Liang) and A.T.; methodology, Q.Q.; software, X.W. and Y.L. (Ying Li); validation, A.T., X.W. and Y.L. (Yanchun Liang); formal analysis, X.W. and A.T.; investigation, X.W.; resources, Q.Q.; data curation, X.W. and Y.L. (Ying Li); writing—original draft preparation, X.W.; writing—review and editing, Q.Q., A.T. and Y.L. (Yanchun Liang); visualization, X.W.; supervision, Y.L. (Yanchun Liang); project administration, Y.L. (Ying Li). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the NSFC Grant Number 61972174, the Guangdong Universities' Innovation Team Grant Number 2021KCXTD015, the Key Disciplines Projects Grant Number 2021ZDJS138, and the Guangdong Provincial Junior Innovative Talents Project for Ordinary Universities Number 2022KQNCX146.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data shall be provided by the corresponding authors upon special request.

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

References

1. Deng, T.; Chau, K.W.; Duan, H.F. Machine learning based marine water quality prediction for coastal hydro-environment management. *J. Environ. Manag.* **2021**, *284*, 112051. [CrossRef] [PubMed]
2. Azrour, M.; Mabrouki, J.; Fattah, G.; Guezzaz, A.; Aziz, F. Machine learning algorithms for efficient water quality prediction. *Model. Earth Syst. Environ.* **2022**, *8*, 2793–2801. [CrossRef]
3. Sinha, K.K.; Gupta, M.K.; Banerjee, M.K.; Meraj, G.; Singh, S.K.; Kanga, S.; Farooq, M.; Kumar, P.; Sahu, N. Neural Network-Based Modeling of Water Quality in Jodhpur, India. *Hydrology* **2022**, *9*, 92. [CrossRef]
4. Jung, W.S.; Kim, S.E.; Kim, Y.D. Prediction of Surface Water Quality by Artificial Neural Network Model Using Probabilistic Weather Forecasting. *Water* **2021**, *13*, 2392. [CrossRef]
5. Wang, S.; Peng, H.; Liang, S. Prediction of estuarine water quality using interpretable machine learning approach. *J. Hydrol.* **2022**, *605*, 127320. [CrossRef]
6. Wan H.; Xu, R.; Zhang, M.; Cai, Y.; Li, J.; Shen, X. A novel model for water quality prediction caused by non-point sources pollution based on deep learning and feature extraction methods. *J. Hydrol.* **2022**, *612*, 128081. [CrossRef]
7. Liu, J.; Wang, X.; Zhao, Y.; Xu, Y.; Pan, Y.; Feng, S.; Liu, J.; Huang, X.; Wang, H. Nh3 plasma functionalization of UiO-66-NH2 for highly enhanced selective fluorescence detection of u (vi) in water. *Anal. Chem.* **2022**, *94*, 10091–10100. [CrossRef]
8. Zhu, R.; Wang, X.; Panther, J.G.; Wang, Q.; Chakir, S.; Ding, Y.; Huang, Y.; Wang, H. Micro/nanostructured MgO hollow spheres with selective adsorption performance and their application for fluoride monitoring in water. *Sep. Purif. Technol.* **2022**, *299*, 121703. [CrossRef]
9. Seo, D.; Sigdel, R.; Kwon, K.H.; Lee, Y.S. 3-D hydrodynamic modeling of Yongdam Lake, Korea using EFDC. *Desalin. Water Treat.* **2010**, *19*, 42–48. [CrossRef]
10. Rand, J.M.; Nanko, M.O.; Lykkegaard, M.B.; Wain, D.; King, W.; Bryant, L.D.; Hunter, A. The human factor: Weather bias in manual lake water quality monitoring. *Limnol. Oceanogr. Methods* **2022**, *20*, 288–303. [CrossRef]
11. Wang, K.; Band, S.S.; Ameri, R.; Biyari, M.; Hai, T.; Hsu, C.C.; Hadjouni, M.; Elmannai, H.; Chau, K.W.; Mosavi, A. Performance improvement of machine learning models via wavelet theory in estimating monthly river streamflow. *Eng. Appl. Comput. Fluid Mech.* **2022**, *16*, 1833–1848. [CrossRef]
12. Zhu, M.; Wang, J.; Yang, X.; Zhang, Y.; Zhang, L.; Ren, H.; Wu, B.; Ye, L. A review of the application of machine learning in water quality evaluation. *Eco-Environ. Health* **2022**, *1*, 10. [CrossRef]
13. Alizadeh, M.J.; Kavianpour, M.R.; Danesh, M.; Adolf, J.; Shamshirband, S.; Chau, K.W. Effect of river flow on the quality of estuarine and coastal waters using machine learning models. *Eng. Appl. Comput. Fluid Mech.* **2018**, *12*, 810–823. [CrossRef]
14. Omambia, A.; Maake, B.; Wambua, A. Water quality monitoring using IoT & machine learning. In Proceedings of the 2022 IST-Africa Conference (IST-Africa 2022), Virtual Conference, 16–20 May 2022; pp. 1–8. https://doi.10.23919/IST-Africa56635.2022.9845590.
15. Kayhomayoon, Z.; Arya Azar, N.; Ghordoyee Milan, S.; Kardan Moghaddam, H.; Berndtsson, R. Novel approach for predicting groundwater storage loss using machine learning. *J. Environ. Manag.* **2021**, *296*, 113237. [CrossRef] [PubMed]

16. Cao, R.; Yuan, J. Selection Strategy of Vibration Feature Target under Centrifugal Pumps Cavitation. *Appl. Sci.* **2020**, *10*, 8190. [CrossRef]
17. Yan, X.; Liu, Y.; Jia, M. A Feature Selection Framework-Based Multiscale Morphological Analysis Algorithm for Fault Diagnosis of Rolling Element Bearing. *IEEE Access* **2019**, *7*, 123436–123452. [CrossRef]
18. Li, G.; Zhang, A.; Zhang, Q.; Wu, D.; Zhan, C. Pearson correlation coefficient-based performance enhancement of broad learning system for stock price prediction. *IEEE Trans. Circuits Syst. II Express Briefs* **2022**, *69*, 2413–2417. [CrossRef]
19. Zheng, Y.; Li, Y.; Wang, G.; Chen, Y.; Xu, Q.; Fan, J.; Cui, X. A novel hybrid algorithm for feature selection based on whale optimization algorithm. *IEEE Access* **2018**, *7*, 14908–14923. [CrossRef]
20. Zhang, Y.; Li, C.; Jiang, Y.; Sun, L.; Zhao, R.; Yan, K.; Wang, W. Accurate prediction of water quality in urban drainage network with integrated EMD-LSTM model. *J. Clean. Prod.* **2022**, *354*, 131724. [CrossRef]
21. Ransom, K.M.; Nolan, B.T.; Traum, J.A.; Faunt, C.C.; Bell, A.M.; Gronberg, J.A.M.; Wheeler, D.C.; Rosecrans, C.Z.; Jurgens, B.; Schwarz, G.E.; et al. A hybrid machine learning model to predict and visualize nitrate concentration throughout the Central Valley aquifer, California, USA. *Sci. Total Environ.* **2017**, *601–602*, 1160–1172. [CrossRef]
22. Rostam, N.A.P.; Malim, N.H.A.H.; Abdullah, R.; Ahmad, A.L.; Ooi, B.S.; Chan, D.J.C. A Complete Proposed Framework for Coastal Water Quality Monitoring System With Algae Predictive Model. *IEEE Access* **2021**, *9*, 108249–108265. [CrossRef]
23. Wang, Z. A Numerical Method for Delayed Fractional-Order Differential Equations. *J. Appl. Math.* **2013**, *2013*, 707–724. [CrossRef]
24. Noor, N.M.; Al Bakri Abdullah, M.M.; Yahaya, A.S.; Ramli, N.A. Comparison of linear interpolation method and mean method to replace the missing values in environmental data set. In *Materials Science Forum*; Trans Tech Publications: Baech, Switzerland, 2015; Volume 803, pp. 278–281.
25. Liu, P.; Wang, J.; Sangaiah, A.; Xie, Y.; Yin, X. Analysis and Prediction of Water Quality Using LSTM Deep Neural Networks in IoT Environment. *Sustainability* **2019**, *11*, 2058. [CrossRef]
26. Hu, Z.; Zhang, Y.; Zhao, Y.; Xie, M.; Zhong, J.; Tu, Z.; Liu, J. A Water Quality Prediction Method Based on the Deep LSTM Network Considering Correlation in Smart Mariculture. *Sensors* **2019**, *19*, 1420. [CrossRef]
27. Kumar, R.; Singh, S.; Bilga, P.S.; Jatin, K.; Pruncu, C.I. Revealing the Benefits of Entropy Weights Method for Multi-Objective Optimization in Machining Operations: A Critical Review. *J. Mater. Res. Technol.* **2021**, *10*, 1471–1492. [CrossRef]
28. Liu, J.; Hu, Q.; Yu, D. A weighted rough set based method developed for class imbalance learning. *Inf. Sci.* **2008**, *178*, 1235–1256. [CrossRef]
29. Mármol, A.M.; Puerto, J.; Fernández, F.R. The use of partial information on weights in multicriteria decision problems. *J. Multi-Criteria Decis. Anal.* **2015**, *7*, 322–329. [CrossRef]
30. Salman, R.; Nikoo, M.R.; Shojaeezadeh, S.A.; Beiglou, P.H.B.; Sadegh, M.; Adamowski, J.F.; Alamdari, N. A novel Bayesian maximum entropy-based approach for optimal design of water quality monitoring networks in rivers. *J. Hydrol.* **2021**, *603*, 126822. [CrossRef]
31. Ly, A.; Marsman, M.; Wagenmakers, E.J. Analytic posteriors for Pearson's correlation coefficient. *Stat. Neerl.* **2018**, *72*, 4–13. [CrossRef]
32. Qiang, L.; Yang, Y.; Yang, L.; Wang, Y. Comparative analysis of water quality prediction performance based on LSTM in the Haihe River Basin, China. *Environ. Sci. Pollut. Res.* **2022**, *30*, 7498–7509. [CrossRef]
33. Naghibi, S.A.; Ahmadi, K.; Daneshi, A. Application of Support Vector Machine, Random Forest, and Genetic Algorithm Optimized Random Forest Models in Groundwater Potential Mapping. *Water Resour. Manag.* **2017**, *31*, 2761–2775. [CrossRef]
34. Kisi, O.; Shiri, J.; Karimi, S.; Shamshirband, S.; Motamedi, S.; Petkovi, D.; Hashim, R. A survey of water level fluctuation predicting in Urmia Lake using support vector machine with firefly algorithm. *Appl. Math. Comput.* **2015**, *270*, 731–743. [CrossRef]
35. Yu, X.; Cui, T.; Sreekanth, J.; Mangeon, S.; Gilfedder, M. Deep learning emulators for groundwater contaminant transport modelling. *J. Hydrol.* **2020**, *590*, 125351. [CrossRef]
36. Gambín, Á.F.; Angelats, E.; González, J.S.; Miozzo, M.; Dini, P. Sustainable Marine Ecosystems: Deep Learning for Water Quality Assessment and Forecasting. *IEEE Access* **2021**, *9*, 121344–121365. [CrossRef]
37. Mohammadi, B.; Guan, Y.; Moazenzadeh, R.; Safari, M.J.S. Implementation of hybrid particle swarm optimization-differential evolution algorithms coupled with multi-layer perceptron for suspended sediment load estimation. *Catena* **2021**, *198*, 105024. [CrossRef]
38. Osman, A.I.A.; Ahmed, A.N.; Chow, M.F.; Huang, Y.F.; El-Shafie, A. Extreme gradient boosting (Xgboost) model to predict the groundwater levels in Selangor Malaysia. *Ain Shams Eng. J.* **2021**, *12*, 1545–1556. [CrossRef]
39. Ni, L.; Wang, D.; Singh, V.P.; Wu, J.; Wang, Y.; Tao, Y.; Zhang, J. Streamflow and rainfall forecasting by two long short-term memory-based models. *J. Hydrol.* **2020**, *583*, 124296. [CrossRef]
40. Zhanga, J.; Zhub, Y.; Zhanga, X.; Yec, M.; Yangb, J. Developing a Long Short-Term Memory (LSTM) based model for predicting water table depth in agricultural areas. *J. Hydrol.* **2018**, *561*, 918–929. [CrossRef]
41. Jiang, Y.; Li, C.; Zhang, Y.; Zhao, R.; Yan, K.; Wang, W. Data-driven method based on deep learning algorithm for detecting fat, oil, and grease (FOG) of sewer networks in urban commercial areas. *Water Res.* **2021**, *207*, 117797. [CrossRef]
42. Jiang, Y.; Li, C.; Song, H.; Wang, W. Deep learning model based on urban multi-source data for predicting heavy metals (Cu, Zn, Ni, Cr) in industrial sewer networks. *J. Hazard. Mater.* **2022**, *432*, 128732. [CrossRef]

43. Aldhyani, T.H.H.; Al-Yaari, M.; Alkahtani, H.; Maashi, M. Water Quality Prediction Using Artificial Intelligence Algorithms. *Appl. Bionics Biomech.* **2020**, *2020*, 6659314. [CrossRef] [PubMed]
44. Qian, J.; Liu, H.; Qian, L.; Bauer, J.; Xue, X.; Yu, G.; He, Q.; Zhou, Q.; Bi, Y.; Norra, S. Water quality monitoring and assessment based on cruise monitoring, remote sensing, and deep learning: A case study of Qingcaosha Reservoir. *Front. Environ. Sci.* **2022**, *10*, 979133. [CrossRef]

entropy

Article

Cross Entropy in Deep Learning of Classifiers Is Unnecessary—ISBE Error Is All You Need

Władysław Skarbek

Faculty of Electronics and Information Technology, Warsaw University of Technology, 00-661 Warszawa, Poland; wladyslaw.skarbek@pw.edu.pl

Abstract: In deep learning of classifiers, the cost function usually takes the form of a combination of SoftMax and CrossEntropy functions. The SoftMax unit transforms the scores predicted by the model network into assessments of the degree (probabilities) of an object's membership to a given class. On the other hand, CrossEntropy measures the divergence of this prediction from the distribution of target scores. This work introduces the ISBE functionality, justifying the thesis about the redundancy of cross-entropy computation in deep learning of classifiers. Not only can we omit the calculation of entropy, but also, during back-propagation, there is no need to direct the error to the normalization unit for its backward transformation. Instead, the error is sent directly to the model's network. Using examples of perceptron and convolutional networks as classifiers of images from the MNIST collection, it is observed for ISBE that results are not degraded with SoftMax only but also with other activation functions such as Sigmoid, Tanh, or their hard variants HardSigmoid and HardTanh. Moreover, savings in the total number of operations were observed within the forward and backward stages. The article is addressed to all deep learning enthusiasts but primarily to programmers and students interested in the design of deep models. For example, it illustrates in code snippets possible ways to implement ISBE functionality but also formally proves that the SoftMax trick only applies to the class of dilated SoftMax functions with relocations.

Keywords: deep learning; cross entropy; normalization function; neural network; model inference; gradient backpropagation

Citation: Skarbek, W. Cross Entropy in Deep Learning of Classifiers Is Unnecessary—ISBE Error Is All You Need. *Entropy* **2024**, *26*, 65. https://doi.org/10.3390/e26010065

Academic Editor: Yanchun Liang

Received: 29 November 2023
Revised: 8 January 2024
Accepted: 9 January 2024
Published: 12 January 2024

1. Introduction

A deep model is a kind of mental shortcut [1], broadly understood as a model created in deep learning of a certain artificial neural network $\mathcal{N}$, designed for a given application. What, then, is an artificial neural network [2], its deep learning [3,4], and what applications [5] are we interested in?

From a programmer's perspective, an artificial neural network is a type of data processing algorithm [6], in which subsequent steps are carried out by configurable computational units, and the order of processing steps is determined by (dynamically created) computing graph. The computing graph is always directed and acyclic (DAG). Interestingly, even recurrent neural networks, such as the LSTM (Long Short-Term Memory) nets, which are trained using gradient methods, have DAG-type computational graphs defined.

At the training stage, each group of input data X, i.e., each group of training examples, technically each batch of training examples, first undergoes the *inference (forward)* phase on the current model, i.e., processing through the network $\mathcal{N}$ at its current parameters W. As a result, network outputs $Y \leftarrow \mathcal{F}_{\mathcal{N}}(X;W)$ are produced [7]. The result of the entire network $\mathcal{N}$ with the functionality $\mathcal{F}_{\mathcal{N}}$ and joined set of parameters W is the result of combining the results of the activities for individual units $\mathcal{U}$ with individual functionalities $\mathcal{F}_{\mathcal{U}}$ and with possible individual parameters $W_{\mathcal{U}}$, as well.

$$\overbrace{\xrightarrow{X;W}\boxed{\mathcal{F}_{\mathcal{N}}}\xrightarrow{Y\leftarrow\mathcal{F}_{\mathcal{N}}(X;W)}}^{\text{Inference}} \equiv \overbrace{\boxed{\quad\cdots\quad \xrightarrow{X_u;W_u}\boxed{\mathcal{F}_{\mathcal{U}}}\xrightarrow{Y_u\leftarrow\mathcal{F}_{\mathcal{U}}(X_u;W_u)}\quad\cdots\quad}}^{\text{Inference}}$$

After the inference phase comes the model update phase, where the current model is modified (improved) according to the selected optimization procedure [8]. *The model update phase* begins with calculating the loss (cost) value $Z \leftarrow \mathcal{L}(Y, Y^{\circ})$ defined by the chosen loss function $\mathcal{L}$ as well as the inference outcome Y and the target result Y°.

$$\overbrace{\xrightarrow{X;W}\boxed{\mathcal{F}_{\mathcal{N}}}\xrightarrow{Y\leftarrow\mathcal{F}_{\mathcal{N}}(X;W)}}^{\text{Inference}} \quad\cdots\quad \overbrace{\xrightarrow{Y,Y^{\circ}}\boxed{\mathcal{L}}\xrightarrow{Z\leftarrow\mathcal{L}(Y,Y^{\circ})}}^{\text{Model update - start}}$$

The loss Z depends (indirectly through Y) on all parameters W and what conditions the next step of the update phase is the determination of sensitivity $\overline{W}$ of the loss function $\mathcal{L}$ to their changes. The mathematical model of sensitivity is the gradient $\overline{W} \doteq \frac{\partial\mathcal{L}}{\partial W}$. Knowing this gradient, the optimizer will make the actual modification of W in a direction that also takes into account the values of gradients obtained for previous training batches.

Calculating the gradient with respect to parameters actually assigned to different computational units required the development of an efficient algorithm for its propagation in the opposite direction to inference [9,10].

Just as in the inference phase, each unit $\mathcal{U}$ has its formula $Y_u \leftarrow \mathcal{F}_{\mathcal{U}}(X_u, W_u)$ for processing data from input X_u to output Y_u with parameters W_u, so in the *backward gradient propagation phase*, it must have a formula $\overline{X}_u, \overline{W}_u \leftarrow \overline{\mathcal{F}}_{\mathcal{U}}(\overline{Y}_u)$ for transforming the gradients assigned to its outputs $\overline{Y}_u$ into gradients assigned to its inputs $\overline{X}_u$ and its parameters $\overline{W}_u$.

$$\overbrace{\xleftarrow{\overline{X};\overline{W}\leftarrow\overline{\mathcal{F}}_{\mathcal{N}}(\overline{Y};X,Y,W)}\boxed{\overline{\mathcal{F}}_{\mathcal{N}}}\xleftarrow{\overline{Y}}}^{\text{BackPropagation}} \quad\cdots\quad \overbrace{\xleftarrow{\overline{Y}\leftarrow\overline{\mathcal{L}}(\overline{Z};Y,Z)}\boxed{\overline{\mathcal{L}}}\xleftarrow{\mathbf{1}=\frac{\partial Z}{\partial Z}}}^{\text{Loss function gradient } \frac{\partial L}{\partial Y}}$$

$$\overbrace{\boxed{\quad\cdots\quad \xleftarrow{\overline{X}_u;\overline{W}_u\leftarrow\overline{\mathcal{F}}_{\mathcal{U}}(\overline{Y}_u;X_u,Y_u,W_u)}\boxed{\overline{\mathcal{F}}_{\mathcal{U}}}\xleftarrow{\overline{Y}_u}\quad\cdots\quad}}^{\text{Gradient BackPropagation}}$$

Based on such local rules of gradient backpropagation and the created computation graph, the *backpropagation* algorithm can determine the gradients of the cost function with respect to each parameter in the network. The computation graph is created during the inference phase and is essentially a stack of links between the arguments and results of calculations performed in successive units [10,11].

Deep learning is precisely a concert of these inference and update phases in the form of gradient propagation, calculated for randomly created groups of training examples. These phases, intertwined, operate on multidimensional, deep tensors (arrays) of data, processed with respect to network inputs, and on deep tensors of gradient data, processed with respect to losses, determined for the output data of the trained network.

Here, by a deep tensor, we mean a multidimensional data array that has many feature maps, i.e., its size along the feature axis is relatively large, e.g., 500, which means 500 scalar feature maps. We then say that at this point in the network, our data has a *deep representation* in a 500-dimensional space.

As for the applications we are interested in this work, the answer is those that have at least one requirement for classification [12]. An example could be crop detection from satellite images [13], building segmentation in aerial photos [14], but also text translation [15]. Classification is also related to voice command recognition [16], speaker recognition [17], segmentation of the audio track according to speakers [18], recognition of speaker emotions with visual support [19], but also classification of objects of interest along with their localization in the image [20].

It may be risky to say that after 2015, in all the aforementioned deep learning classifiers, the cost function takes the form of a composition of the $SoftMax$ function [21] and the $CrossEntropy$ function, i.e., cross-entropy [22]. The `SoftMax` unit normalizes the scores predicted by the classifier model for the input object into *SoftMax* scores that sum up to one, which can be treated as an estimation of the conditional probability distribution of classes. Meanwhile, cross-entropy measures the divergence of this estimation from the target probability distribution (class scores). In practice, the target score may be taken from a training set prepared manually by a so-called teacher [23] or may be calculated automatically by another model component, e.g., in the knowledge distillation technique [24].

For K classes and n_b training examples, the $SoftMax$ function is defined for the raw score matrix $X \in \mathbb{R}^{n_b \times K}$ as:

$$[Y \leftarrow SoftMax(X)] \quad \longrightarrow \quad \left[Y_{bi} \leftarrow \frac{e^{X_{bi}}}{\sum\limits_{j\in[K]} e^{X_{bj}}},\ b \in [n_b],\ i \in [K] \right],$$

where the notation $[K]$ denotes any K-element set of indices—in this case, they are class labels.

The $CrossEntropy$ function on the matrix $Y, Y^\circ \in \mathbb{R}^{n_b \times K}$ is defined by the formula:

$$[Z \leftarrow CrossEntropy(Y, Y^\circ)] \longrightarrow \left[Z_b \leftarrow -\sum_{j\in[K]} Y^\circ_{bj} \log_e Y_{bj},\ b \in [n_b],\ z \in \mathbb{R}^{n_b} \right]$$

$$\overbrace{\begin{array}{c} \overbrace{\xrightarrow{\text{classified object}} \boxed{\mathcal{F}_\mathcal{N}} \xrightarrow{\text{raw scores } X}}^{\text{Scores Inference}} \\ \overbrace{\xrightarrow{\text{raw scores } X} \boxed{\texttt{SoftMax}} \xrightarrow{\text{soft scores } Y, Y^\circ} \boxed{\texttt{CrossEntropy}} \xrightarrow{\text{losses } Z}}^{\text{Loss Estimation } \mathcal{L}} \end{array}}^{\text{Classifier loss function: Separated Implementation}} \tag{1}$$

When classifiers began using a separated implementation of the combination of the `SoftMax` normalization and the `CrossEntropy` loss, it quickly became evident in practice that its implementation had problems with scores close to zero, both in the inference phase and in the backward propagation of its gradient. In formulas of properties 1–3 in Theorem 1 of Section 2, we see from where the problem comes. Only the integration of `CrossEntropy` with normalization `SoftMax` eliminated these inconveniences. The integrated approach has the following form:

$$\overbrace{\begin{array}{c} \overbrace{\xrightarrow{\text{classified object}} \boxed{\mathcal{F}_\mathcal{N}} \xrightarrow{\text{raw scores } X}}^{\text{Inference}} \\ \overbrace{\xrightarrow{\text{raw scores } X\text{,soft scores ,}Y^\circ} \boxed{\texttt{CrossEntropy} \circ \texttt{SoftMax}} \xrightarrow{\text{losses } Z}}^{\text{Loss Estimation } \mathcal{L}} \end{array}}^{\text{Classifier loss function - Integrated Implementation}}$$

The integrated functionality of these two features has the following redundant mathematical notation:

$$Z \leftarrow [CrossEntropy \circ SoftMax](X, Y^\circ) \longrightarrow$$

$$Z_b \leftarrow -\sum_{j\in[K]} Y^\circ_{bj} \log_e \frac{e^{X_{bj}}}{\sum\limits_{i\in[K]} e^{X_{bi}}},\ b \in [n_b]$$

This redundancy in notation was helpful in deriving the equation for the gradient backpropagation for the integrated loss function $CrossEntropy \circ SoftMax$. Later, we will also use for such composition the name $SoftCE$.

The structure of this paper is as follows:

1. The second section is devoted to mathematics of the *SoftMax trick*. Its validity is proved in two ways:
   (a) Using gradient formulas for the composition of differentiable functions;
   (b) Using the concept of the Jacobian matrix and linear algebra calculus.
2. In the third section titled *ISBE Functionality*, the conditions that a normalization unit must meet for its combination with a cross-entropy unit to have a gradient at the input equal to the difference in soft scores: $\overline{X} = Y - Y^{\circ}$ are analyzed. Then the definition of `ISBE` functionality is introduced, which in the inference phase `(I)` normalizes the raw score to a soft score `(S)`, and in the backward propagation phase `(B)` returns an error `(E)`, equal to the difference in soft scores. It is also justified why, in the case of the $SoftMax$ normalization function, the `ISBE` functionality has, from the perspective of the learning process, the functionality of the integrated element `CrossEntropy ∘ SoftMax`.
3. In the first subsection of the fourth section, using the example of the problem of recognizing handwritten digits and the standard `MNIST(60K)` image collection [25], numerous experiments show that in addition to the obvious savings in computational resources, in the case of five activations serving as normalization functions, the classifier's effectiveness is not lower than that of the combination of the normalization `SoftMax` and `Cross Entropy`. This `ISBE` property was verified for the activation units `SoftMax, Sigmoid, Hardsigmoid,` and `Tanh` and `Hardtanh`. The second subsection of the fourth section reports on how `ISBE` behaves for a more demanding dataset `CIFAR-10` and a more complex architecture VGG-16.
4. The final fifth section contains conclusions.
5. In Appendix A, the class of functions leading to the dilated *SoftMax trick* is fully characterized using concepts of dilation and relocation of function domain.
6. In Appendix B the `ISBE` functionality is integrated with `PyTorch` class `torch.autograd.Function.`

The main contributions of this research are:

1. Introducing `ISBE` functionality as simplification and, at the same time, extension of the functional combination of `SoftMax` with `CrossEntropy`.
2. Verification of `ISBE` feasibility and efficiency on two datasets and three CNN architectures.
3. Enhancement of theoretical background for the concept of *SoftMax trick* via its generalization and full characterization of normalization functions which exhibit this property.

Concluding this introduction, I would like to emphasize that this work is not intended to depreciate the concept of entropy in the context of machine learning. It has played and continues to play a key role as a loss function. Its form appears naturally in many data modeling tasks. For example, in the case of multi-class logistic regression, when computing optimal weights, maximizing the negative logarithm of the likelihood function directly leads to the cross-entropy function. In the context of modeling, cross-entropy will remain an important research tool. The context of the `ISBE` functionality concerns only the specific method of calculating the gradient of network parameters for the needs of SGD (Stochastic Gradient Descent) optimizers. Only this, nothing more.

2. Discrete Cross-Entropy and the SoftMax Trick Property

This section is devoted to the mathematics of the *SoftMax trick*. Its validity will be proved in two ways:

1. Using gradient formulas for the composition of differentiable functions,
2. Using the concept of Jacobian matrix and linear algebra calculus.

The discrete cross-entropy function CE of a target discrete probability distribution $y^\circ \in [0,1]^K$, $\sum_i y_i^\circ = 1$, relative to the calculated by the classifier the probability distribution $y \in (0,1)^K$, $\sum_i y_i = 1$, is defined by the formula:

$$CE(y^\circ, y) \doteq -\sum_{i\in[K]} y_i^\circ \log_e y_i \tag{2}$$

The notation $[K]$ refers to a sequence of K indexes, such as $(1,\ldots,K)$ or $(0,1,\ldots,K-1)$.

It appears that the gradient of the cross entropy $CE(y^\circ, y)$ with respect to y is not defined at the zero components:

$$\nabla_y CE(y^\circ, y) = -y^\circ \div y\,, \tag{3}$$

where the operation $\div$ for the expression $w = u \div v$, denotes the division of vector u components by the components of vector v, i.e., $w_i \doteq u_i/v_i$, $i \in [K]$.

In the special case when the vector $y \in (0,1)^K$ is calculated based on the vector $x \in \mathbb{R}^K$ according to the formula $y_i = SoftMax(x)_i = e^{x_i} / \sum_k e^{x_k}$, the gradient of CE with respect to x has a particularly simple resultant formula. Its simplicity was the reason for the term *SoftMax trick*:

$$\begin{aligned} & y(x) \doteq y \doteq SoftMax(x),\; SoftCE(y^\circ, x) \doteq CE(y^\circ, SoftMax(x)) \\ & \longrightarrow \nabla_x SoftCE(y^\circ, x) \overset{\text{SoftMax trick}}{=} y - y^\circ \end{aligned} \tag{4}$$

Some authors [26] also use the term *SoftMax trick* for that part of the proof showing that the derivative of the natural logarithm of the sum of functions e^{x_i} equals to the $SoftMax$ function.

The *SoftMax trick* can be described as a theorem and proved in two ways: the elementary one and via the matrix calculus. The following theorem includes elementary properties of the cross-entropy function, optionally preceded by the $SoftMax$ normalization.

Theorem 1. *Let $y^\circ \in [0,1]^K$, $\sum_{k\in[K]} y_k^\circ = 1$, be the target probability distribution and $y \in (0,1)$, $\sum_{k\in[K]} y_k = 1$, be the predicted probability distribution. Then*

1. *For any $i \in [K]$, $\frac{\partial CE(y^\circ,y)}{\partial y_i} = \frac{y_i^\circ}{y_i}$.*
2. $\nabla_y CE(y^\circ, y) = y^\circ \div y \doteq \left[\frac{y_1^\circ}{y_1}, \ldots, \frac{y_K^\circ}{y_K}\right]^\intercal$
3. *The range of CE covers the positive part of the real axis:*

$$\left\{CE(y^\circ, y) : y^\circ \in [0,1]^K,\; y \in (0,1)^K\right\} = (0,\infty)$$

4. *Let $x \in \mathbb{R}^K$ be the vector of raw scores, and y° be the target probability distribution. Then $SoftMax$ normalization followed by CE, is defined as follows*

$$SoftCE(y^\circ, x) \doteq -\sum_{j\in[K]} y_j^\circ \log_e \frac{e^{x_j}}{\sum_{k\in[K]} e^{x_k}}\,.$$

Contrary to CE only, $SoftCE$ exhibits the bounded gradient:

(a) *The Jacobian of $SoftMax$ equals to:*

$$Jacobian(SoftMax)(x) = \frac{\partial SoftMax(x)}{\partial x} = \mathtt{diag}[y] - yy^\intercal \textit{ where } y = SoftMax(x)\,.$$

(b) *For any $i \in [K]$, $\frac{\partial[SoftCE(y^\circ,x)]}{\partial x_i} = y_i - y_i^\circ$, where $y_i = (SoftMax(x))_i$.*

(c) $\nabla_x SoftCE(y^\circ, x) = y - y^\circ$, *where $y = SoftMax(x)$.*

(d) *The range of $SoftCE$ covers the interval $(-1,1)$:*

$$\left\{SoftCE(y^\circ, x) : y^\circ \in [0,1]^K,\ x \in \mathbb{R}^K\right\} = (-1, +1)$$

Elementary proof of SoftCE properties.

$$y_i \doteq \frac{e^{x_i}}{\sum_k e^{x_k}} \longrightarrow \frac{\partial[\log_e(\sum_k e^{x_k})]}{\partial x_i} = y_i \longrightarrow$$

$$\begin{aligned}\frac{\partial SoftCE(y^\circ, x)}{\partial x_i} &= \frac{\partial\left[\sum_j y_j^\circ \log_e(\sum_k e^{x_k}) - \sum_j y_j^\circ \overbrace{\log_e e^{x_j}}^{x_j}\right]}{\partial x_i} \\ &= y_i \overbrace{\sum_j y_j^\circ}^{=1} - y_i^\circ = y_i - y_i^\circ\end{aligned}$$

□

Proof of $SoftCE$ property using matrix calculus.

In matrix notation [27], the property of *SoftMax trick* has a longer proof, as we first need to calculate the Jacobian of the $SoftMax$ function [28].

$$\text{If } y_j \doteq \frac{e^{x_j}}{\sum_{k\in[K]} e^{x_k}}, \text{ then } \frac{\partial y_j}{\partial x_i} = \begin{cases} \dfrac{-e^{x_j}\cdot e^{x_i}}{\left(\sum_{k\in[K]} e^{x_k}\right)^2} = -y_i \cdot y_j, \text{ when } i \neq j \\ \dfrac{e^{x_i}\cdot\left(\sum_{k\in[K]} e^{x_k}\right) - e^{x_i}\cdot e^{x_i}}{\left(\sum_{k\in[K]} e^{x_k}\right)^2} = y_i - y_i^2 = (1-y_i)\cdot y_i, \\ \text{when } i = j \end{cases}$$

The general formula is $\dfrac{\partial y_j}{\partial x_i} = (\delta_{ij} - y_i)y_j$. Therefore:

$$\left(\frac{\partial y}{\partial x}\right)_{ij} \doteq \frac{\partial y_j}{\partial x_i} = \delta_{ij}y_j - y_iy_j = (\mathtt{diag}[y])_{ij} - (yy^\intercal)_{ij}.$$

Hence, $\frac{\partial y}{\partial x} = \mathtt{diag}[y] - yy^\intercal$. From the chain rule

$$\frac{\partial[SoftCE(y^\circ, x)]}{\partial x} \doteq \frac{\partial CE(y^\circ, y(x))}{\partial x} = \left(\frac{\partial y}{\partial x}\right)^\intercal \cdot \frac{\partial[CE(y^\circ, y)]}{\partial y}, \text{ where } y(x) \doteq SoftMax(x)$$

and the symmetry of $SoftMax$ Jacobian matrix $\frac{\partial y}{\partial x}$, we obtain:

$$\begin{aligned}\tfrac{\partial[SoftCE(y^\circ,x)]}{\partial x} &= (\mathtt{diag}[y] - yy^\intercal)(-y^\circ \div y) \\ &= y\underbrace{(y \div y)^\intercal y^\circ}_{\mathbf{1}_K^\intercal y^\circ = 1} - \underbrace{\mathtt{diag}[y \div y]}_{I_K} y^\circ = y - y^\circ \end{aligned} \qquad (5)$$

□

While looking at the above two proofs for the Theorem 1, a question can be raised: Is it only the $SoftMax$ function that has *SoftMax trick* property? The answer to this problem can be found in Appendix A. You can understand why the second proof using Jacobian matrix and matrix calculus has been presented here.

3. ISBE Functionality

The ISBE functionality is a proposed simplification of the cost function, combining the SoftMax normalization function with the cross-entropy function, hereafter abbreviated as CE$_{all}$. Its role is to *punish* those calculated probability distributions that significantly differ from the distributions of scores proposed by the *teacher*.

To understand this idea, let us extend the inference diagram for CE$_{all}$ with the backward propagation part for the gradient. We consider this diagram in its separate version, omitting earlier descriptions for the diagram (1):

$$\begin{gathered}\overbrace{\xrightarrow{X}\boxed{\texttt{SoftMax}}\xrightarrow{Y,Y^\circ}\boxed{\texttt{CrossEntropy}}\xrightarrow{Z}}^{\text{Loss Inference }\mathcal{L}}\\ \overbrace{\xleftarrow{\overline{X}\xleftarrow{Theorem\,1}Y-Y^\circ}\boxed{\overline{\texttt{SoftMax}}}\xleftarrow{\overline{Y};Y,Y^\circ}\boxed{\overline{\texttt{CrossEntropy}}}\xleftarrow{\overline{Z}=\mathbf{1}}}^{\text{BackPropagation}}\end{gathered}\tag{6}$$

The meaning of variables X, Y, Y°, Z and $\overline{Z}, \overline{Y}, \overline{X}$ appearing in the above diagram (6):

X	raw score at the input of the normalization function preceding cross-entropy CE, $X \in \mathbb{R}^K$,
Y	normalization result, so-called soft score ,$Y \in (0,1)^K$,
Y°	target soft score, assigned to the classified example ,
Z	output of cross-entropy CE ,$Z \in \mathbb{R}$,
$\overline{Z}$	formal gradient at the input of the backward propagation algorithm, $\overline{Z} = 1$,
$\overline{Y}$	gradient of cross-entropy CE with respect to Y: $\overline{Y} = \frac{\partial Z}{\partial Y} = -\frac{Y^\circ}{Y}$,
$\overline{X}$	gradient of cross-entropy CE with respect to X: $\overline{X} \xleftarrow{Theorem\,1} (Y - Y^\circ)$.

The key formula here is $\overline{X} \leftarrow (Y - Y^\circ)$. Its validity comes from the mentioned Theorem 1 which includes the proof for the Formula (4) associated with the *SoftMax trick* property.

The generalized form of this property is given in the Appendix A within the Theorem A1 which includes interesting observations on necessary and sufficient conditions for the *SoftMax trick*.

For instance, the Equation (A2) on the form of the Jacobian of the normalization unit is both a sufficient and necessary condition for its combination with the cross-entropy unit to ensure the equality (A3). Moreover, this condition implies that an activation function with a Jacobian of the *SoftMax* type is a SoftMax function with optional relocation.

Theorem A1 leads us to a seemingly pessimistic conclusion: it is not possible to seek further improvements by changing the activation and at the same time expect the *SoftMax trick* property to hold. Thus, the question arises: what will happen if, along with changing the activation unit, we change the cross-entropy unit to another or even omit it entirely?

In the *ISBE* approach, the aforementioned simplification of the CE$_{all}$ cost function involves precisely omitting the cross-entropy operation in the inference stage and practically omitting all backward operations for this cost function. So what remains? The answer is also an opportunity to decode the acronym *ISBE* again:

1. In the inference phase (I), we normalize the raw score X to $Y = SoftMax(X)$, characterized as a soft score (S).
2. In the backward propagation phase (B), we return an error (E) equal to the difference between the calculated soft score and the target score, i.e., $\overline{X} \doteq Y - Y^\circ$.

Why can we do this and still consider that in the case of the SoftMax activation function, the value of the gradient transmitted to the network is identical: $\overline{X}_{CE_{all}} = \overline{X}_{ISBE} \doteq Y - Y^\circ$?

The answer comes directly from the property $\overline{X}_{CE_{all}} = Y - Y^\circ$, formulated in Equation (4), which as it was already mentioned, was proved in the Theorem 1 as the *SoftMax trick* property.

We thus have on the left the following diagram of data and gradient backpropagation through such a unit. On the right, we have its generalization to a `ScoreNormalization` unit instead of `SoftMax` unit.

$$\left.\begin{array}{l}\overbrace{\xrightarrow{X}\boxed{\texttt{SoftMax}}\xrightarrow{Y,Y^\circ}}^{\text{ISBE Inference}}\\[1em] \overbrace{\xleftarrow{\overline{X}\leftarrow Y-Y^\circ}\boxed{\texttt{Subtract}}\xleftarrow{Y,Y^\circ}}^{\overline{ISBE}\text{ BackPropagation}}\end{array}\right\}\xrightarrow{\text{generalize}}\left\{\begin{array}{l}\overbrace{\xrightarrow{X}\boxed{\texttt{Score Normalization}}\xrightarrow{Y,Y^\circ}}^{\text{ISBE Inference}}\\[1em] \overbrace{\xleftarrow{\overline{X}\leftarrow Y-Y^\circ}\boxed{\texttt{Subtract}}\xleftarrow{Y,Y^\circ}}^{\overline{ISBE}\text{ BackPropagation}}\end{array}\right.$$

Which activation functions should we reach for in order to test them in the ISBE technique?

1. The SoftMax activation function should be the first candidate for comparison, as it theoretically guarantees behavior comparable to the system containing cross-entropy.
2. Activations should be monotonic so that the largest value of the raw score remains the largest score in the soft score sequence.
3. Soft scores should be within a limited range, e.g., $[0, 1]$ as in the case of SoftMax and Sigmoid, or $[-1, +1]$ as for Tanh.
4. The activation function should not map two close scores to distant scores. For example, normalizing a vector of scores by projecting onto a unit sphere in the p-th Minkowski norm meets all the above conditions. However, it is not stable around zero. Normalization $\frac{x}{\|x\|_p}$ maps, for example, two points $\epsilon, -\epsilon$ distant by $2 \cdot \|\epsilon\|_p$ to points distant exactly by 2, thus changing their distance $\frac{1}{\|\epsilon\|_p}$ times, e.g., a million times, when $\|\epsilon\|_p = 10^{-6}$. This operation is known in `Pytorch` library as `normalize` function.

The experiments conducted confirm the validity of the above recommendations. The `Pytorch` library functions `SoftMax, sigmoid, tanh, hardsigmoid, hardtanh` meet the above three conditions and provide effective classification at a level of effectiveness higher than 99.5%, comparable to `CrossEntropy` $\circ$ `SoftMax`. In contrast, with the function `normalize`, the optimizer failed to converge on the same `MNIST(60K)` collection and with the same architectures.

What connects these *good normalization* functions $F : \mathbb{R}^K \to \mathbb{R}^K$, of which two are not even fully differentiable? Certainly, it is the Lipschitz condition occurring in a certain neighborhood of zero [29]:

$$x \in \mathbb{R}^K,\ \|x\|_p \leq \epsilon \longrightarrow \|F(x)\|_p \leq c\|x\|_p\,, \quad \text{where } c \text{ is a certain constant}\,.$$

Note that the Lipschitz condition meets the expectations of the fourth requirement on the above list of recommendations for `ISBE`. Moreover, we do not expect here that the constant c be less than one, i.e., that the function F has a narrowing character.

We also need a recommendation for *teachers* preparing class labels, which we represent as vectors blurred around the base vectors of axes $I_K = [e_1, \ldots, e_K],\ e_i[j] \doteq \delta_{ij}$:

1. example blurring value μ, e.g., $\mu = 10^{-6}$:

$$\tilde{e}_i[j] \leftarrow (1-\mu)\delta_{ij} + \frac{\mu}{K-1}(1-\delta_{ij})$$

2. when the range of activation values is other than the interval $[0, 1]$, we adjust the vector $\tilde{e}_i$ to the new range, e.g., for `tanh` the range is the interval $(-1, +1)$ and then the adjustment has the form:

$$\tilde{e}_i \leftarrow 2 \cdot \tilde{e}_i - 1,\ i = 1, \ldots, K$$

Finally, let us take a look at the code for the main loop of the program implemented on the `Pytorch` platform.

1. This is what the code looks like when `loss_function` is chosen as `nn.CrossEntropyLoss`:

```
for (labels,images) in tgen:
    outputs = net(images)
    loss = loss_function(outputs, labels)
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

2. Now we introduce the `ISBE` option for `SoftMax` activation and replace the call for loss function by soft error calculation:

```
for (labels,images) in tgen:
    outputs = net(images)
    soft_error = SoftMax(outputs) - labels
    optimizer.zero_grad()
    outputs.backward(soft_error)
    optimizer.step()
```

More options, including the definition of `ISBE` functionality, can be found in Appendix B. Of course, the above code snippets are only intended to illustrate how easy it is to add the functionality of `ISBE` to an existing application.

4. Experiments

What do we want to learn from the planned experiments? We already know from theory that in the case of the `SoftMax` activation, we cannot worsen the parameters of the classifier using cross-entropy, both in terms of success rate and learning time.

Therefore, we first want to verify whether theory aligns with practice, but also to check for which normalization functions the `ISBE` functionality does not degrade the model's effectiveness compared to CE_{all}.

The learning time t_{ISBE} should be shorter than t_{CE}. Still, to be independent of the specific implementation, we will compare the percentage of the backpropagation time in the total time of inference and backpropagation:

$$\tau \doteq \frac{\text{backpropagation time}}{\text{inference time} + \text{backpropagation time}} \times 100\% \tag{7}$$

From many quality metrics, for simplicity, we choose the success rate (also called accuracy), defined as the percentage of correctly classified elements from the test collection `MNIST(10K)`

$$\alpha = \frac{\text{number of correct classifications}}{\text{size of the test collection}} \times 100\% \tag{8}$$

We want to know how this value changes when we choose different architectures and different activations in the ISBE technique, as well as different options for aggregating cross-entropy over the elements of the training batch.

4.1. Experiments with `MNIST` *Dataset*

Firstly, we evaluate the efficiency of the ISBE idea on the standard `MNIST(60K)` image collection and the problem of their classification.

We have the following degrees of freedom in our experiments:

1. Two architecture options
   - Architecture $\mathcal{N}_0$ consists of two convolutions $\mathbb{C}$ and two linear units $\mathbb{F}$, of which the last one is a projection from the space of deep feature vectors of dimension 512 to the space of raw scores for each of the $K = 10$ classes:

$$\xrightarrow{\text{image}} \mathcal{I}^{1}_{28_{yx}}\ \mathbb{C}^{32}_{3^k 2^s}\ \mathbb{C}^{64}_{3^k 2^s}\ \mathbb{D}_{20}\, \mathbb{F}^{512}\, \mathbb{F}^{10} \xrightarrow{\text{class scores}}$$

as by STNN notation [30], for instance

$$\begin{cases} \mathbb{C}^{32}_{3^k 2^s} & \text{means 32 convolutions with } 3 \times 3 \text{ masks, sampled with a stride of 2,} \\ \mathbb{D}_{20} & \text{DropOut—a unit zeroing 20\% of tensor elements,} \\ \mathbb{F}^{512} & \text{a linear unit with a matrix } A \in \mathbb{R}^{?\times 512}, \\ & \text{here } ? = 64\text{—it is derived from the shape of} \\ & \text{the tensor produced by the previous unit}. \end{cases}$$

- Architecture $\mathcal{N}_1$ consists of two blocks, each with three convolutions—it is a purely convolutional network, except for the final projection:

$$\xrightarrow{\text{image}} \mathcal{I}^{1}_{28_{yx}}\ \mathbb{C}^{32}_{3^k}\ \mathbb{C}^{32}_{3^k}\ {}_{p}\mathbb{C}^{32}_{5^k 2^s}\ \mathbb{D}_{40}\ \mathbb{C}^{64}_{3^k}\ \mathbb{C}^{64}_{3^k}\ {}_{p}\mathbb{C}^{64}_{5^k 2^s}\ \mathbb{D}_{40}\ \mathbb{C}^{128}_{4^k}\, \mathbb{F}^{10} \xrightarrow{\text{class scores}}$$

  Note that the last convolution in each block has a p requirement for padding, i.e., filling the domain of the image with additional lines and rows so that the image resolution does not change.

2. Three options for reducing the vector of losses in the `CrossEntropyLoss` element: `none, mean, sum`.
3. Five options for activation functions used in the ISBE technique:
   - `SoftMax`: $y_i \leftarrow \dfrac{e^{x_i}}{\sum\limits_{j\in[K]} e^{x_j}}, \quad i \in [K],$
   - `Tanh`: $y_i \leftarrow \dfrac{e^{x_i} - e^{-x_i}}{e^{x_i} + e^{-x_i}}, \quad i \in [K],$
   - `HardTanh`: $y_i \leftarrow \left\{ \begin{array}{cl} -1 & \text{if } x_i \le -1 \\ x_i & \text{if } -1 < x_i < +1 \\ +1 & \text{if } +1 \le x_i \end{array} \right\}, \quad i \in [K],$
   - `Sigmoid`: $y_i \leftarrow \dfrac{1}{1+e^{-x_i}},\ i \in [K],$
   - `HardSigmoid`: $y_i \leftarrow \left\{ \begin{array}{cl} 0 & \text{gdy } x_i \le -2 \\ \frac{x_i+2}{4} & \text{gdy } -2 < x_i < +2 \\ +1 & \text{gdy } +2 \le x_i \end{array} \right\} = \dfrac{HardTanh(x_i/2)+1}{2},$ $i \in [K]$.

The results of the experiments, on the one hand, confirm our assumption that the conceptual Occam's razor, i.e., the omission of the cross-entropy unit, results in time savings τ, and on the other hand, the results are surprisingly positive with an improvement in the metric of success rate α in the case of hard activation functions *HardTanh* and *HardSigmoid*. It was observed that only the option of reduction by `none` behaves exactly according to theory, i.e., the success rate is identical to the model using $SoftMax$ normalization. Options `mean` and `sum` for the model with *entropy* are slightly better than the model with *SoftMax*.

The consistency of models in this case means that the number of images incorrectly classified out of 10 thousand is the same. The experiments did not check whether it concerns the same images. A slight improvement, in this case, meant that there were less than a few or a dozen errors, and the efficiency of the model above 99.6% meant at most 40 errors per 10 thousand of test images.

4.1.1. Comparison of Time Complexity

We compare time complexity according to the metric given by the Formula (7).

In the context of time, Table 1 clearly shows that the total timeshare of backpropagation, obviously depending on the complexity of the architecture, affects the time savings of the ISBE technique compared to `CrossEntropyLoss`—Table 2. The absence of pluses in this table, i.e., the fact that all solutions based on `ISBE` are relatively faster in the learning phase, is an undeniable fact.

The greatest decrease in the share of backpropagation, over 3%, occurs for the $Sigmoid$ and $SoftMax$ activations. The smallest decrease in architecture $\mathcal{N}_0$ is noted for the soft (soft) normalization function $Tanh$ and its hard version $HardTanh$. This decrease refers to cross-entropy without reduction, which is an aggregation of losses calculated for all training examples in a given group into one numerical value.

Table 1. Comparison of the metric τ, i.e., the percentage share of backpropagation time in the total time with inference. The share τ_{CE} of cross-entropy with three types of reduction is compared with five functions of soft normalization. The analysis was performed for architectures $\mathcal{N}_0$ and $\mathcal{N}_1$.

Net	Mean	None	Sum	Hsigmoid	Htanh	Sigmoid	SoftMax	Tanh
$\mathcal{N}_0$	60.61%	59.56%	59.98%	58.31%	58.21%	57.38%	57.45%	59.07%
$\mathcal{N}_1$	54.89%	53.92%	53.98%	52.68%	52.33%	51.75%	51.95%	52.11%
$\mathcal{N}_1^r$	54.45%	53.92%	54.00%	52.78%	52.30%	51.67%	51.73%	52.11%

Table 2. Metric $\Delta\tau \doteq \tau_{ISBE} - \tau_{CE}$, i.e., the decrease in the percentage share of backpropagation time in the total time with inference. The analysis was performed for architectures $\mathcal{N}_0$ and $\mathcal{N}_1$.

Net	CE Loss	Hsigmoid	Htanh	Sigmoid	SoftMax	Tanh
$\mathcal{N}_0$	mean	−2.30%	−2.40%	−3.23%	−3.16%	−1.54%
$\mathcal{N}_0$	none	−1.25%	−1.35%	−2.18%	−2.11%	−0.50%
$\mathcal{N}_0$	sum	−1.67%	−1.77%	−2.60%	−2.53%	−0.92%
$\mathcal{N}_1$	mean	−2.21%	−2.56%	−3.14%	−2.94%	−2.79%
$\mathcal{N}_1$	none	−1.24%	−1.59%	−2.17%	−1.97%	−1.82%
$\mathcal{N}_1$	sum	−1.30%	−1.65%	−2.23%	−2.03%	−1.87%

Inspired by the Theorem A1, which states that the relocation of the $SoftMax$ function preserves the *SoftMax trick* property, we also add data to the Table 1 for the network $\mathcal{N}_1^r$. This network differs from the $\mathcal{N}_1$ network only because the normalization unit has a trained relocation parameter. In practice, we accomplish training with relocation for normalization by training with the relocation of the linear unit immediately preceding it. This is done by setting its parameter: `bias = True`.

As we can see, the general conclusion about the advantage of the ISBE technique in terms of time reduction for the model with the relocation of the normalization function is the same.

4.1.2. Comparison of Classifier Accuracy

Comparison of classifier accuracy and differences in this metric are contained in Tables 3 and 4. The accuracy is computed according to the Formula (8).

The number of pluses on the side of `ISBE` clearly exceeds the number of minuses. The justification for this phenomenon requires separate research. Some light will be shed on this aspect by the analysis of learning curves—the variance in the final phase of learning is clearly lower. The learning process is more stable.

Table 3. In the table, the success rate of three classifiers based on cross-entropy with different aggregation options is compared with the success rate determined for five options of soft score normalization functions. The analysis was performed for architectures $\mathcal{N}_0$ and $\mathcal{N}_1$.

Net	Mean	None	Sum	Hsigmoid	Htanh	Sigmoid	SoftMax	Tanh
$\mathcal{N}_0$	99.45%	99.41%	99.47%	99.50%	99.50%	99.56%	99.41%	99.45%
$\mathcal{N}_1$	99.61%	99.58%	99.59%	99.64%	99.66%	99.64%	99.62%	99.63%
$\mathcal{N}_1^r$	99.55%	99.64%	99.64%	99.61%	99.66%	99.69%	99.63%	99.57%

Table 4. Change in success rate between models with cross-entropy and models with soft score normalization function. The analysis was performed for architectures $\mathcal{N}_0$ and $\mathcal{N}_1$.

Net	CE Loss	Hsigmoid	Htanh	Sigmoid	SoftMax	Tanh
$\mathcal{N}_0$	mean	0.05%	0.05%	0.11%	−0.04%	0.00%
$\mathcal{N}_0$	none	0.09%	0.09%	0.15%	0.00%	0.00%
$\mathcal{N}_0$	sum	0.13%	0.03%	0.09%	−0.06%	−0.02%
$\mathcal{N}_1$	mean	0.03%	0.05%	0.03%	0.01%	0.02%
$\mathcal{N}_1$	none	0.06%	0.08%	0.06%	0.04%	0.05%
$\mathcal{N}_1$	sum	0.05%	0.07%	0.05%	0.03%	0.04%

In Table 4, we observe that, with the exception of the function $SoftMax$, which on several images of digits performed worse than the model with cross-entropy, the soft activations have an efficiency slightly or significantly better. However, we are talking about levels of tenths or hundredths of a percent here. The largest difference noted for the option `SoftMax` was 15-hundredths of a percent, meaning 15 more images correctly classified. Such differences are within the margin of statistical error.

The use of relocation for the normalization function does not provide a clear conclusion—for some models, there is a slight improvement; for others, there is a slight deterioration. It is true that the `ISBE` functionality with `sigmoid` activation achieved the best efficiency of 99.69%, but this is only a matter of a few images.

Within the limits of statistical error, we can say that the `ISBE` functionality gives the same results in recognizing `MNIST` classes. Its advantages are:

- of decrease time in the total time,
- simplification of architecture, and therefore playing the philosophical role of *Occam's razor*.

4.1.3. Visual Analysis

Further analysis of the results will be based on the visual comparison of learning curves.

First, let us see on three models `cross-entropy-mean, SoftMax, sigmoid` their loss and efficiency curves obtained on training data `MNIST(54K)` and on data intended solely for model validation `MNIST(6K)`. These two loss curves are calculated after each epoch. We supplement them with a loss curve calculated progressively after each batch of training data (see Figure 1).

Let us note the correct course of the train loss curve with respect to the progressive loss curve—both curves are close. The correct course is also for the validation loss curve—the validation curve from about epoch 30 is below the training curve, maintaining a significant distance. This effect was achieved only after applying a moderate input image augmentation procedure via random affine transformations in the pixel domain.

Correct behavior of learning curves was recorded both for the models with entropy and for models with the `ISBE` functionality. This also applies to classifier performance curves.

1. Curves of loss functions can appear together as long as the type of function is identical, which entails a similar range of variability for loss function values. One might wonder what measure of loss to adopt in the case of ISBE since this technique, in fact, does not calculate loss values. We opt for a natural choice of mean square error for errors in soft scores:
$$\mathcal{L}_{ISBE} = MSE(Y, Y^\circ) \doteq \frac{1}{n_b} \cdot \|Y - Y^\circ\|_2^2$$
where n_b is the batch size.
For such defined measures, it turns out that only the option of reduction by summing has a different range of variability, and therefore it is not on the Figure 2.
2. In the case of classifier accuracy, a common percentage scale does not exclude placing all eight curves for each considered architecture. However, due to the low transparency of such a figure, it is also worth juxtaposing different groups of curves of the dependency $\alpha(e)$. The accuracy α of the classifier `MNIST(60K)` is calculated on the test set `MNIST(10K)`.

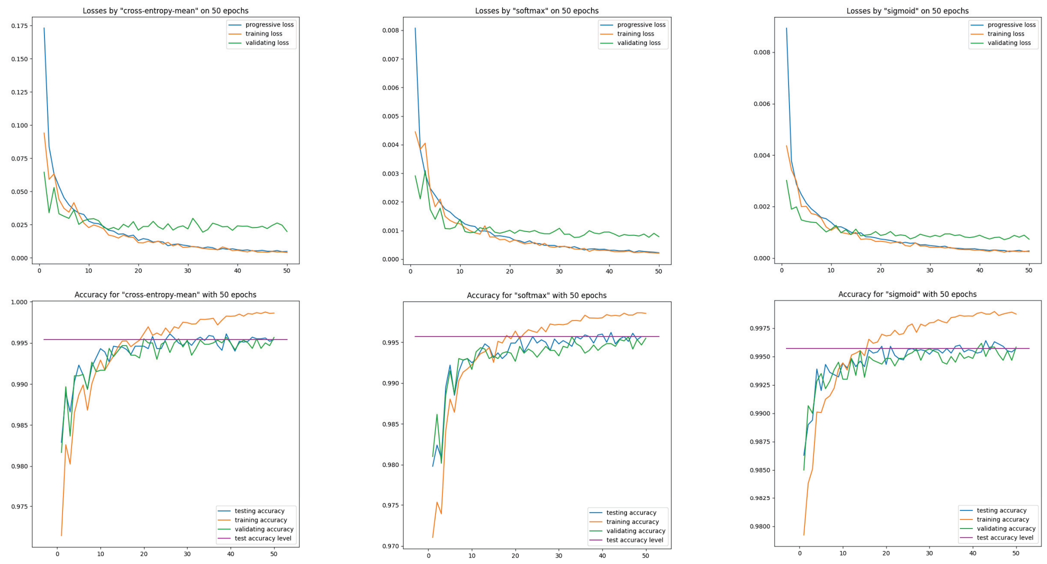


Figure 1. Learning curves on training and validation data for the $\mathcal{N}_1$ network and three models: `cross-entropy-mean`, `SoftMax`, `sigmoid`. The horizontal reference line represents the accuracy of test data computed after the last epoch.

Sets of curves, which we visualize separately for architectures $\mathcal{N}_0, \mathcal{N}_1$ are:

- All options for loss functions (3) and soft score functions (5),
- `CE none`, `CE mean`, `CE sum` versus `SoftMax`,
- `CE none`, `CE mean`, `CE sum` versus `tanh`, `hardtanh`,
- `SoftMax` versus `sigmoid`, `hardsigmoid`,
- `SoftMax` versus `tanh`, `hardtanh`,
- `SoftMax` versus `sigmoid`, `tanh`.

Due to space constraints, we show learning curves and classifier effectiveness graphs only for architecture $\mathcal{N}_1$ in Figures 2 and 3.

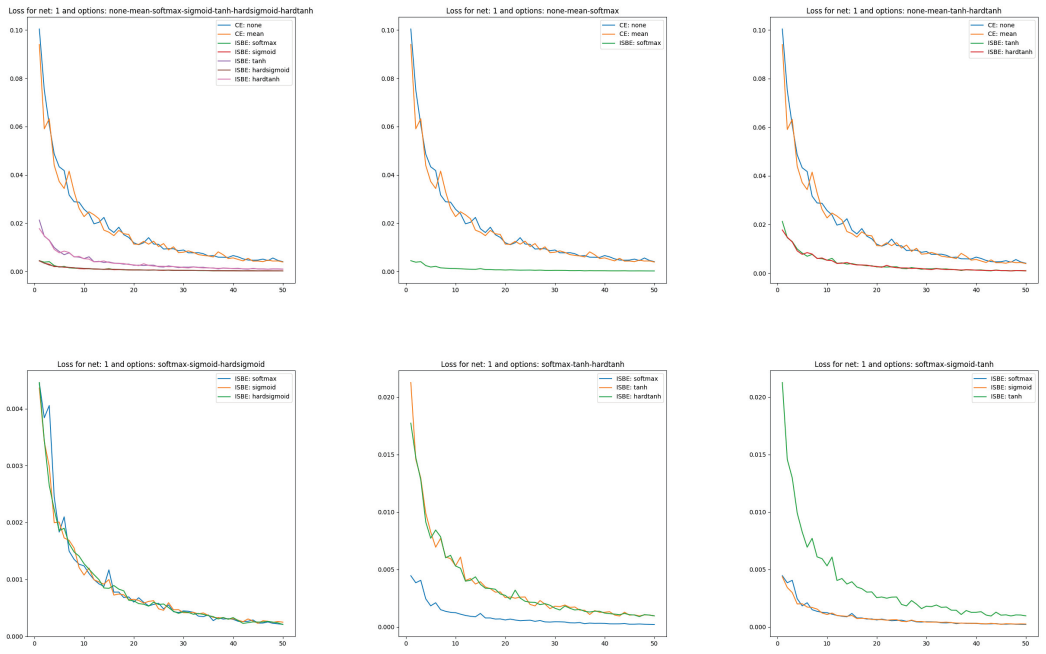


Figure 2. Loss charts of learning in comparisons of CE versus ISBE options. In the first row: (1) all options for loss functions and soft score functions; (2) `CE none, CE mean` versus `SoftMax`; (3): `CE none, CE mean` versus `tanh, hardtanh`. In the second row: (1) `SoftMax` versus `sigmoid, hardsigmoid`; (2) `SoftMax` versus `tanh, hardtanh`; (3) `SoftMax` versus `sigmoid, tanh`.

In Figure 2 we can clearly observe four clusters of models:

- `CrossEntropyLoss` based with reduction option `sum` (as out of common range it was not shown),
- `CrossEntropyLoss` based with reduction options `none`, and `mean`,
- `ISBE` based with normalizations to range $[0,1]$ including functions $SoftMax$, $Sigmoid$, and $HardSigmoid$,
- `ISBE` based with normalizations to range $[-1,1]$ including functions $Tanh$, and $HardTanh$.

Within a cluster, the loss curves behave very similarly. Interestingly, the loss curves in ISBE-based clusters tend to the same value greater than zero. In contrast, cross-entropy-based curves also tend to the same limit. However it is clearly greater than ISBE one.

Now, we will pay more attention to test learning curves. We generate test learning curves on the full set of test data `MNIST(10K)`. After each epoch, one point is scored towards the test learning curve. We will show these curves in several comparative contexts.

Accuracy charts of learning (see Figure 3) were obtained to compare cross entropy (CE) performances versus ISBE performance. We have:

- Comparison of CE versus soft options:
  1. all options for loss functions and soft score functions
  2. `CE none, CE mean` versus `SoftMax`,
  3. `CE none, CE mean` versus `tanh, hardtanh`.

- Comparison of `SoftMax` versus other soft options:
  1. `SoftMax` versus `sigmoid, hardsigmoid`,
  2. `SoftMax` versus `tanh, hardtanh`,
  3. `SoftMax` versus `sigmoid, tanh`.

In the case of classifier accuracy curves, the variances in the clusters described above are smaller than in the union of clusters. Close to the final epochs, all curves tend to be chaotic within the range of $(99.4, 99.7)$.

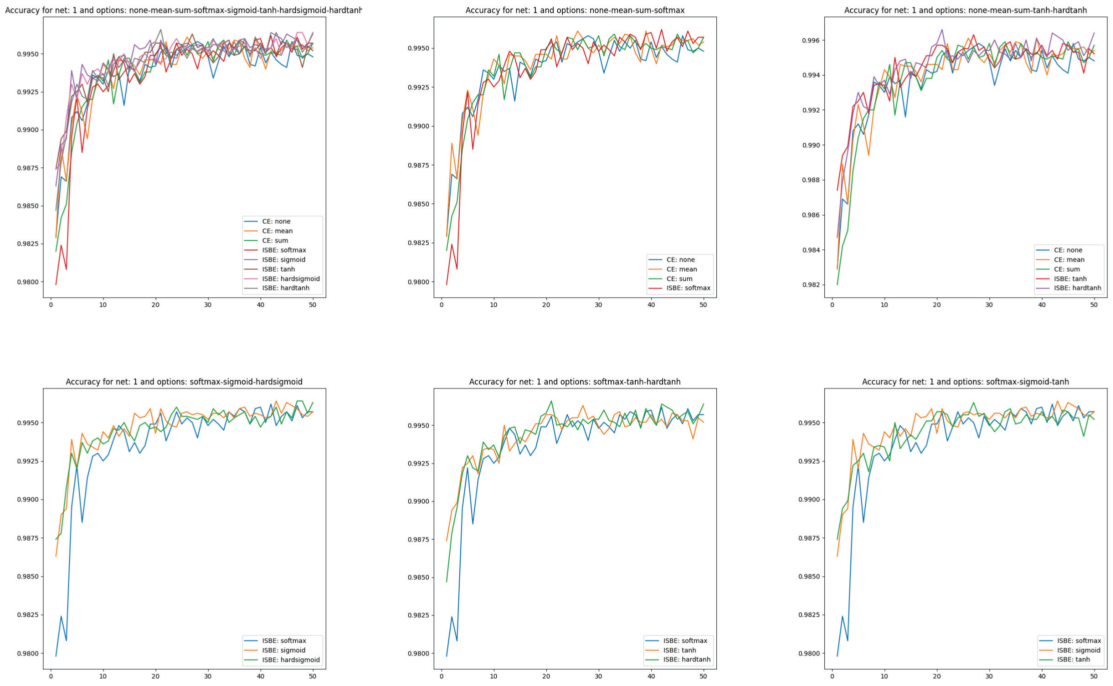


Figure 3. Accuracy charts of learning in comparisons of CE versus ISBE options. In the first row: (1) all options for loss functions and soft score functions; (2) `CE none, CE mean` versus `SoftMax`; (3): `CE none, CE mean` versus `tanh, hardtanh`. In the second row: (1) `SoftMax` versus `sigmoid, hardsigmoid`; (2) `SoftMax` versus `tanh, hardtanh`; (3) `SoftMax` versus `sigmoid, tanh`.

Visualizing the effectiveness of classifiers for different architectures of different complexities, although more obvious, also has research value (see Figure 4):

- `CE none, CE mean, CE sum` from $\mathcal{N}_0$ versus `CE none, CE mean, CE sum` from $\mathcal{N}_1$,
- `CE none, SoftMax` from $\mathcal{N}_0$ versus `CE none, SoftMax` from $\mathcal{N}_1$,
- `SoftMax, sigmoid` from $\mathcal{N}_0$ versus `SoftMax, sigmoid` from $\mathcal{N}_1$,
- `sigmoid, tanh` from $\mathcal{N}_0$ versus `sigmoid, tanh` from $\mathcal{N}_1$,
- `sigmoid, hardsigmoid` from $\mathcal{N}_0$ versus `sigmoid, hardsigmoid` from $\mathcal{N}_1$,
- `tanh, hardtanh` from $\mathcal{N}_0$ versus `tanh, hardtanh` from $\mathcal{N}_1$.

Figure 4 shows the better performance of $\mathcal{N}_1$ compared to $\mathcal{N}_0$. Moreover, we can observe slightly more stable behavior for ISBN-based curves than for cross-entropy-based.

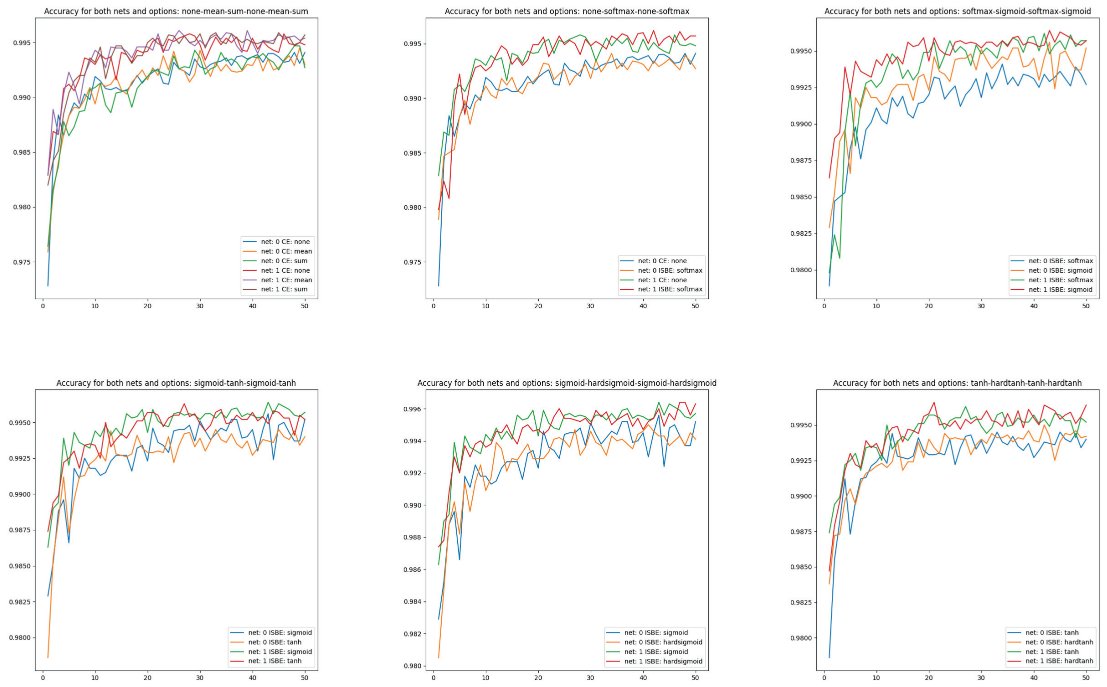


Figure 4. Accuracy charts of learning in comparisons of CE versus ISBE options and architecture $\mathcal{N}_0$ versus $\mathcal{N}_1$. In the first row: (1) `CE none, CE mean, CE sum`; (2) `CE none, SoftMax`; (3): `SoftMax, sigmoid`. In the second row: (1) `sigmoid, tanh`; (2) `sigmoid, hardsigmoid`; (3) `tanh, hardtanh`.

4.2. Experiments with `CIFAR-10` *Dataset*

In this subsection, the `CIFAR-10`—the more demanding than `MNIST` dataset is considered in the context of `ISBE` functionality. Moreover, the VGG feature extractor with more than 14 M parameters, i.e., more than 10 times larger model than $\mathcal{N}_1$, is joined to make further tests. In Figure 5, we can compare sample images from `MNIST` dataset and `CIFAR-10` dataset. What is immediately observed is the background of objects classified—the uniform black for `MNIST` and the natural scene in case of `CIFAR-10`. It is the main reason that despite the almost perfect fit achieved by VGG-16 on the training set `CIFAR-10` of 50 thousand images,the best results on the independent testing dataset of 10 thousand images are near 93%. The best results known w `CIFAR-10` for all architectures attempted so far are near 95%—about one percent more than the record achieved by human beings.

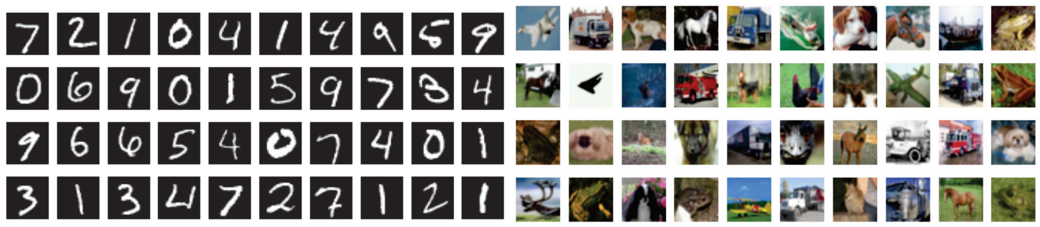


Figure 5. Comparing sample images from `MNIST` and `CIFAR-10` datasets. `CIFAR-10` classes: `plane, car, bird, cat, deer, dog, frog, horse, ship, truck`.

The architecture VGG-16 was presented by Simonyan and Zisserman in their seminal paper [31], *Very Deep Convolutional Networks for Large-Scale Image Recognition*. *VGG-16* model now serves the community as the universal image feature extractor. Its structure has the following sequential form:

$$\xrightarrow{\text{rgb}} \mathcal{I}^{3}_{32yx}\ {\mathbb{C}_{br}}^{64}_{3}\ {\mathbb{C}_{br}}^{64}_{3}\ {}_{m}\mathbb{P}_{2} \xrightarrow{vgg_1} {\mathbb{C}_{br}}^{128}_{3}\ {\mathbb{C}_{br}}^{128}_{3}\ {}_{m}\mathbb{P}_{2} \xrightarrow{vgg_2} {\mathbb{C}_{br}}^{256}_{3}\ {\mathbb{C}_{br}}^{256}_{3}\ {\mathbb{C}_{br}}^{256}_{3}\ {}_{m}\mathbb{P}_{2} \xrightarrow{vgg_3}$$
$${\mathbb{C}_{br}}^{512}_{3}\ {\mathbb{C}_{br}}^{512}_{3}\ {\mathbb{C}_{br}}^{512}_{3}\ {}_{m}\mathbb{P}_{2} \xrightarrow{vgg_4} {\mathbb{C}_{br}}^{512}_{3}\ {\mathbb{C}_{br}}^{512}_{3}\ {\mathbb{C}_{br}}^{512}_{3}\ {}_{m}\mathbb{P}_{2} \xrightarrow{vgg_5} \mathbb{F}^{10} \xrightarrow{\text{class scores}}$$

Like for the two architectures $\mathcal{N}_0, \mathcal{N}_1$ tested for MNIST, the optimizer used for model updates is still AdaM with exponential decay of learning rate with respect to epochs. However, now the initial learning rate is 0.1, not 0.01.

In Figure 6, we can observe better convergence for all ISBE options than for the cross-entropy. Moreover, during testing, the loss value for CE is slowly increasing, starting at about epoch 30, while for all ISBE options, it is stabilizing on the fixed level.

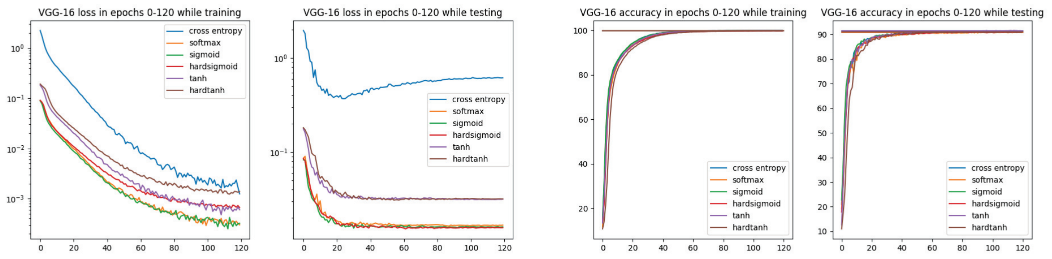


Figure 6. Loss and accuracy charts for VGG-16 architecture and `CIFAR-10` dataset. In the loss chart for training, we can observe better convergence for all ISBE options than for cross-entropy.

From the results presented in Figure 7, it is visible that in the training and testing stages, there are different clusterings for ISBE options:

- In training, there are three groups of ISBE options: {`hardtanh`}, {`tanh, hardsigmoid`}, {`sigmoid, SoftMax`}.
- In testing there are two groups: {`tanh, hardtanh`} and {`SoftMax, sigmoid, hardsigmoid`}.

The significant gap between `tanh, hardtanh` and other ISBE options can be explained by different ranges for the first group and for the second one, i.e., $(-1, +1)$ versus $(0, 1)$. It is not fully clear why in the training stage `hardtanh` is separate to `tanh`.

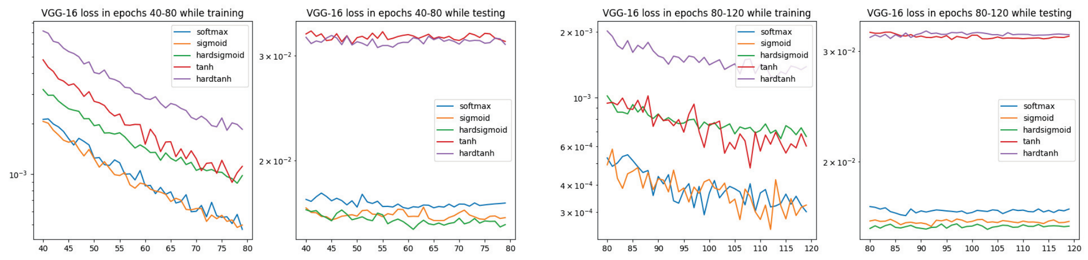


Figure 7. Loss charts for VGG-16 architecture and `CIFAR-10` dataset within epochs 40–80 and 80–120 (only ISBE options are shown). During testing, we can observe two clusters for convergence: the *sigmoid* cluster and the *tanh* cluster.

In Figure 8, the accuracy for cross-entropy and all ISBE options can be compared. It is observed that hard versions are inferior to others. However, while testing, a slight advantage is achieved by the hyperbolic tangent `tanh`.

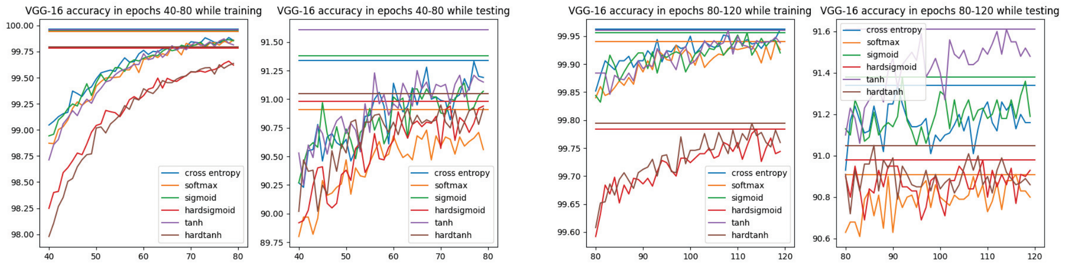


Figure 8. Accuracy charts for VGG-16 architecture and `CIFAR-10` dataset within epochs 40–80 and 80–120. Each horizontal line denotes the maximum accuracy for the option of the same color.

Ultimately, we have bad news on time savings when using `autograd` interface. Contrary to `MNIST` experiments where `ISBE` functionality was implemented by the direct replacement of CE loss in the main learning loop, the `CIFAR-10` experiments were using the definition of `ISBE_func` class being the extension to `torch.autograd.Function` class. It seems that the general mechanism of interfacing to `C++` used by `PyTorch` in this case, is less efficient than for `cross_entropy` function. This is perhaps the reason that functionality with fewer operations takes slightly more time while the same functionality without explicit use of `autograd` mechanism gives always time savings up to 3%.

5. Conclusions

Cross-entropy CE as a loss function *owes much to normalization* performed by the `SoftMax` activation function. In the backward gradient backpropagation phase, only this activation, through perfect linearization, can prevent the explosion or suppression of the gradient originating from CE. What we call the *SoftMax trick*, as a mathematical phenomenon, is explained by the theory presented in the second section and its extension in Appendix A. There is proof that such linearization can only be realized by a function $F : \mathbb{R}^K \to \mathbb{R}^K$ with a Jacobian identical to that of the `SoftMax` function. In turn, such a Jacobian can only be derived for *the dilated and relocated* versions of the `SoftMax` function.

For further research, there remain practical aspects of a more general Theorem A1 implying that dilated and relocated versions of SoftMax are the only ones having the property of *dilated SoftMax trick*. However, it is quite intuitive that the dilation vector could be used to deal with class unbalanced datasets.

Should we, therefore, celebrate this unique relationship between activation and cost function? In this work, we have shown that it is rather beneficial to use the final effect of the action of this pair, namely the linear value equal to $Y - Y^{\circ}$, which can be calculated without their participation. This is exactly what the `ISBE` functionality does—it calculates the soft score vector in the forward step to return in the backward step its error from the target score.

To determine the normalized score, the `ISBE` functionality can use not only the `SoftMax` function, as it is not necessary to meet the unity condition, i.e., to ensure a probability distribution as scores of the trained classifier. At least four other activation functions `sigmoid`, `tanh` and their hard versions `HardSigmoid` and `HardTanh` perform no worse. The choice of these final activations was rather a matter of chance, so researchers face further questions. How do we normalize raw scores and appropriately represent (encode) class labels in relation to this normalization to not degrade the classifier's results? What properties should such normalization functions have? Experiments suggest that meeting the Lipschitz condition in the vicinity of zero may be one of these properties.

The theoretical considerations presented prove that the `ISBE` functionality in the process of deep model learning correctly simulates the behavior of the `CrossEntropy` unit preceded by the `SoftMax` normalization.

The experiments showed that the `ISBE` functionality saves the time of forward and backward stages up to 3%, and the effectiveness of the classifier model remains unchanged within the margin of statistical error. Obviously, those gains are strongly dependent on datasets and network architectures.

In turn, a more complex case of integrating `ISBE` functionality with AD tools (Auto-Grad) of a given platform can be solved for `PyTorch` by copying the proven code from Appendix B. However, as we described in the section on experiments with `CIFAR-10`, the time savings were consumed by this kind of interfacing to `autograd` system.

Funding: This research received no external funding.

Data Availability Statement: Data and their references are contained within the article.

Acknowledgments: Let me express my thanks to the anonymous reviewers, whose insightful and valuable comments encouraged me to make few modifications and extensions to the original version of this article.

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

Appendix A. Functions Giving the SoftMax Trick for Cross-Entropy

While looking on the two proofs for the Theorem 1 an interesting question arises: is it only the $SoftMax$ function that has *SoftMax trick* property? It seems possible that there are others, as for any differentiable function $F : \mathbb{R}^n \to \mathbb{R}^n$, the starting point for reasoning is the same:

$$\frac{\partial[CE(y^\circ, F(x))]}{\partial x} \overset{y \doteq F(x)}{=} \left(\frac{\partial F(x)}{\partial x}\right)^\intercal \underbrace{\frac{\partial[CE(y^\circ, y)]}{\partial y}}_{-y^\circ \div y} = \left(\frac{\partial F(x)}{\partial x}\right)^\intercal (-y^\circ \div y)$$

The following theorem fully characterizes functions that have the *dilated SoftMax trick* property.

Theorem A1 (On the properties of the *SoftMax trick*)**.**
For a differentiable function $F : \mathbb{R}^K \to \mathbb{R}^K$, the following three properties are equivalent:

1. *F is a generalized SoftMax function if there exist a reference point $c \in \mathbb{R}^K$ and a dilation vector $d \in \mathbb{R}^K$, such that for every $x \in \mathbb{R}^K$:*

$$y = F(x) = SoftMax(d \odot x - c)\,, \tag{A1}$$

where $\odot$ operation is the component-wise multiplication.

2. *F has a dilated SoftMax-type Jacobian, if there exists dilation vector $d \in \mathbb{R}^k$, such that for every $x \in \mathbb{R}^K$:*

$$Jacobian(F)(x) \doteq \frac{\partial F(x)}{\partial x} = \mathtt{diag}[d \odot y] - y(d \odot y)^\intercal \doteq (D_y - yy^\intercal)D_d\,, \tag{A2}$$

where $y = F(x)$, $D_y \doteq \mathtt{diag}[y]$, $D_d \doteq \mathtt{diag}[d]$.

3. *F possesses the dilated SoftMax trick property, if for every target vector $y^\circ \in [0,1]^K$, and $x \in \mathbb{R}^K$ its Jacobian matrix $\frac{\partial F(x)}{\partial x}$ satisfies the following equation:*

$$\left(\frac{\partial F(x)}{\partial x}\right)^\intercal (-y^\circ \div y) = D_d(y - y^\circ) = d \odot (y - y^\circ)\,, \text{ where } y = F(x)\,. \tag{A3}$$

Proof of Theorem A1.

We prove the implications in the following order: $(2) \longrightarrow (3)$, $(3) \longrightarrow (2)$, $(1) \longrightarrow (2)$, $(2) \longrightarrow (1)$.

- Proof of implication $(2) \longrightarrow (3)$:
  If the Jacobian $\frac{\partial F(x)}{\partial x}$ of the function F is of the *dilated SoftMax* type, then for $y = F(x)$:
  $$\left(\frac{\partial F(x)}{\partial x}\right)^{\intercal}(-y^{\circ} \div y) = D_d(D_y - yy^{\intercal})(-y^{\circ} \div y)$$
  $$= D_d\left(y\overbrace{(y \div y)^{\intercal} y^{\circ}}^{\mathbf{1}_K^{\intercal} y^{\circ}=1} - \underbrace{\mathtt{diag}[y \div y]}_{I_K} y^{\circ} \right) = D_d(y - y^{\circ}) = d \odot (y - y^{\circ})$$

- Proof of implication $(3) \longrightarrow (2)$: Denote the axis unit vector j by $e_j \in \mathbb{R}^K$. Then $(e_j)_i = \delta_{ij}$. Substitute into property (A3) the target score vector $y^{\circ} \doteq e_j$. Then
  $$d_i(y_i - (e_j)_i) = \left(\frac{\partial F(x)}{\partial x_i}\right)^{\intercal}(-e_j \div y) = \sum_{k\in[K]} \frac{\partial y_k}{\partial x_i} \cdot \left(\frac{-\delta_{kj}}{y_k}\right) = \frac{\partial y_j}{\partial x_i} \cdot \left(\frac{-1}{y_j}\right)$$
  Therefore $\frac{\partial y_j}{\partial x_i} = d_i \cdot ((e_j)_i - y_i)y_j \overset{(e_j)_i=\delta_{ij}}{=} d_i \cdot (\delta_{ij} - y_i)y_j$. Swapping i with j we obtain: $\frac{\partial y_i}{\partial x_j} = (\delta_{ji} - y_i) \cdot d_j$.
  Thus,
  $$\frac{\partial y}{\partial x} = (\mathtt{diag}[y] - yy^{\intercal}) \odot d = (D_y - yy^{\intercal})D_d = \mathtt{diag}[d \odot y] - y(d \odot y)^{\intercal}.$$

- Proof of implication $(1) \longrightarrow (2)$:
  If $y_j \doteq \dfrac{e^{d_j x_j - c_j}}{\sum_{k\in[K]} e^{d_k x_k - c_k}}$, then
  $$\frac{\partial y_j}{\partial x_i} = \begin{cases} \dfrac{-e^{d_j x_j - c_j} \cdot e^{d_i x_i - c_i} \cdot d_i}{\left(\sum_{k\in[K]} e^{d_k x_k - c_k}\right)^2} = -(d_i y_i) \cdot y_j, \text{ when } i \neq j \\ \dfrac{d_i \cdot e^{d_i x_i - c_i} \cdot \left(\sum_{k\in[K]} e^{d_k x_k - c_k}\right) - e^{d_i x_i - c_i} \cdot e^{d_i x_i - c_i} \cdot d_i}{\left(\sum_{k\in[K]} e^{d_k x_k - c_k}\right)^2} \\ = d_i(y_i - y_i^2) = (1 - y_i) \cdot (d_i y_i), \text{ when } i = j \end{cases}$$
  The general formula is $\dfrac{\partial y_j}{\partial x_i} = (\delta_{ij} - y_j)(d_i y_i)$. Therefore:
  $$\left(\frac{\partial y}{\partial x}\right)_{ji} \doteq \frac{\partial y_j}{\partial x_i} = \delta_{ij}(d_i y_i) - y_j(d_i y_i) = (\mathtt{diag}[d \odot y])_{ji} - \left(y(d \odot y)^{\intercal}\right)_{ji}.$$
  Hence, $\frac{\partial y}{\partial x} = \mathtt{diag}[d \odot y] - y(d \odot y)^{\intercal} = (D_y - yy^{\intercal})D_d$

- Proof of implication $(2) \longrightarrow (1)$:
  If $\frac{\partial y_i}{\partial x_i} = ((1 - y_i) \cdot y_i) \cdot d_i$ and $\frac{\partial y_j}{\partial x_i} = -y_j y_i d_i$ then the diagonal of the Jacobian matrix gives us differential Equations [32], from which we can determine the general form of the function $y_i(x)$, $i \in [K]$:

$$\frac{\partial y_i}{(1-y_i)\cdot y_i} = d_i \partial x_i \longrightarrow \int \left(\frac{1}{y_i} + \frac{1}{1-y_i}\right)\partial y_i = \int d_i \partial x_i$$

$$\longrightarrow \log_e \frac{y_i}{1-y_i} = d_i x_i + C(k \neq i) \longrightarrow y_i = \frac{e^{d_i x_i}}{\underbrace{e^{d_i x_i} + e^{-C(k\neq i)}}_{z_i}}$$

$$\longrightarrow y_i = \frac{e^{d_i x_i}}{z_i}, \text{ where } z_i = z_i(x_1, \ldots, x_K) > 0$$

$$\longrightarrow \log_e z_i = d_i x_i - \log_e y_i$$

Now we calculate the partial derivatives $\frac{\partial \log_e z_j}{\partial x_i}$. If $i \neq j$, then

$$\frac{\partial[\log_e z_j]}{\partial x_i} = \frac{\partial[d_j x_j - \log_e y_j]}{\partial x_i} = -\frac{1}{y_j} \overbrace{\frac{\partial y_j}{\partial x_i}}^{-y_j y_i d_i} = y_i d_i$$

For $i = j$, the result is the same:

$$\frac{\partial[\log_e z_i]}{\partial x_i} = \frac{\partial[d_i x_i - \log_e y_i]}{\partial x_i} = d_i - \frac{1}{y_i} \overbrace{\frac{\partial y_i}{\partial x_i}}^{(1-y_i)y_i d_i} = d_i - \frac{(1-y_i)y_i d_i}{y_i} = y_i d_i$$

Therefore, for any $j \in [K]$, we have K equalities: $\frac{\partial[\log_e z_j]}{\partial x_i} = d_i y_i = \frac{\partial[\log_e z_1]}{\partial x_i}$, $i \in [K]$. This means that vector fields for each pair of functions $\log_e z_j$ and $\log_e z_1$ are identical. Integrating these fields yields the same function up to a constant c_j: $\log_e z_j = \log_e z_1 + c_j$, $j \in [K]$. Consequently, $z_j = z_1 \cdot e^{c_j}$, $j \in [K]$, and therefore $y_j = \frac{e^{d_j x_j}}{z_1 e^{c_j}} = \frac{e^{d_j x_j - c_j}}{z_1}$. From the *unity* condition, we can now determine the value of z_1:

$$1 = \sum_{k\in[K]} y_k = \sum_{k\in[K]} \frac{e^{d_k x_k - c_k}}{z_1} \longrightarrow z_1 = \sum_{k\in[K]} e^{d_k x_k - c_k} \longrightarrow y_j = \frac{e^{d_j x_j - c_j}}{\sum\limits_{k\in[K]} e^{d_k x_k - c_k}}.$$

□

Note that the above theorem excludes the functions `Sigmoid`, `Tanh` and `HardSigmoid`, `HardTanh` from the group of functions for which we can apply the SoftMax trick. Namely, in the vector version, all these functions, none of them can be considered as the special form of the generalized `SoftMax` function. It is obvious fact, but to give a formal reason, we observe that all those functions operate on each component of vector x independently, i.e., the result y_i depends only on argument x_i. In the generalized `SoftMax` function, y_i depends on all arguments $x_1, \ldots, x_n$.

Appendix B. `ISBE` Functionality in `PyTorch`

Appendix B.1. Testing Soft Options-Direct Way

`ISBE` functionality can be introduced to our training procedures in many ways.

1. The simplest way of replacing the call of the cross-entropy function is by calling $SoftMax$ and, after, subtracting target hot vectors or their soften versions, calling the backward for the net output:

```
for (labels,images) in tgen:
    outputs = net(images)
    soft_error = SoftMax(outputs) - labels
    optimizer.zero_grad()
    outputs.backward(soft_error)
```

```
        optimizer.step()
```

2. If we want to test more options and compare them with cross-entropy, the loop code will extend a bit:

```
for (labels,images) in tgen:
    outputs = net(images)
    if no_cross_entropy:
        if soft_option=="SoftMax":
            soft_error = SoftMax(outputs) - labels
        if soft_option=="tanh":
            soft_error = tanh(outputs) - (2.*labels-1.)
        elif soft_option=="hardtanh":
            soft_error = hardtanh(outputs) - (2.*labels-1.)
        elif # ...
             # next options
        optimizer.zero_grad()
        outputs.backward(soft_error)
    else:
        loss = loss_function(outputs, labels)
        optimizer.zero_grad()
        loss.backward()
    optimizer.step()
```

3. If we prefer to have a visually shorter loop, then by introducing the variable `soft_function` and extending the class `DataProvider` with matching target labels for a given soft option, we finally obtain a compact form:

```
for (labels,images) in tgen:
    outputs = net(images)
    if no_cross_entropy:
        soft_error = soft_function(outputs) - labels
        optimizer.zero_grad()
        outputs.backward(soft_error)
    else:
        loss = loss_function(outputs, labels)
        optimizer.zero_grad()
        loss.backward()
    optimizer.step()
```

4. However, if we want to register `ISBE` functionality as `torch.autograd.Function` then we have to follow the instruction of `PyTorch` on this kind registration. The effect is described in the next subsection.

Appendix B.2. `ISBE` *Functionality with Automatic Differentiation*

In order to make `ISBE_func` callable in both inference and backpropagation stage we have to define three static methods in extension of class `torch.autograd.Function`:

```
class ISBE_func(torch.autograd.Function):
  @staticmethod
  def forward(...)
   <body of forward>
  @staticmethod
  def setup_context(...)
   <body of setup_context>
  @staticmethod
  def backward(...)
   <body of backward>
```

The whole job is performed in the body of `forward` static method. Other two methods simply switch tensors:

1. Body of `forward`:

```
def forward(raw_scores, labels,
            options=dict(soft='SoftMax', num_classes=10, eps=1e-8)):
    K = options['num_classes']
    eps = options['eps']; soft_option = options['soft']
    one_hots =  torch.nn.functional.one_hot(\
                labels, num_classes=K)*(1-K*eps)+eps)
    if soft_option=='SoftMax':
        soft_scores = torch.nn.functional.SoftMax(raw_scores,dim=1)
        target_scores = one_hots
    elif soft_option=='sigmoid':
        soft_scores = torch.nn.functional.sigmoid(raw_scores)
        target_scores = one_hots
    elif soft_option=='hardsigmoid':
        soft_scores = torch.nn.functional.hardsigmoid(raw_scores)
        target_scores = one_hots
    elif soft_option=='tanh':
        soft_scores = torch.nn.functional.tanh(raw_scores)
        target_scores = 2.*one_hots-1.
    elif soft_option=='hardtanh':
        soft_scores = torch.nn.functional.hardtanh(raw_scores)
        target_scores = 2.*one_hots-1.
    soft_scores.requires_grad = False
    soft_errors = soft_scores - target_scores
    mse_soft = torch.mean(soft_errors**2)
    return mse_soft, soft_errors, soft_scores
```

2. Body of `setup_context`:

```
def setup_context(ctx, inputs, output):
    raw_scores, labels, options = inputs
    mse_soft, soft_errors, soft_scores = output
    ctx.set_materialize_grads(False)
    ctx.soft_errors = soft_errors
```

3. Body of `backward`:

```
def backward(ctx, grad_mse, grad_errors, grad_scores):
    return ctx.soft_errors, None, None
```

We cannot directly call the `forward` static function. We have to use `apply` method.

```
def isbe_func_(raw_scores, labels,
               options=dict(soft='SoftMax', num_classes=10, eps=1e-8)):
  return ISBE_func.apply(raw_scores, labels, options)
```

We could also simplify the use of options if there is a global object 'ex' which includes its reference:

```
isbe_func = lambda raw_scores,labels:\
  isbe_func_(raw_scores, labels, options=ex.loss_options)[0]
```

Instead of a method `backward` on `PyTorch` tensor we could use its wrapper `isbe_backward`:

```
isbe_backward = lambda soft_error: soft_error.backward()
```

Finally we can also hide the options for the `F.cross_entropy` function:

```
tones_ = torch.ones(ex.batch_size).to(ex.device)
cross_entropy_func = lambda x,t:\
  F.cross_entropy(x,t,reduction=ex.loss_options['reduction'],
                  label_smoothing=ex.loss_options['label_smoothing'])
ce_backward = lambda loss: loss.backward(tones_[:loss.size(0)])\
   if ex.loss_options['reduction']=='none' else loss.backward()
```

References

1. Schmidhuber, J. Annotated History of Modern AI and Deep Learning. *arXiv* **2022**, arXiv:2212.11279.
2. Rosenblatt, F. The Perceptron: A Probabilistic Model For Information Storage and Organization in the Brain. *Psychol. Rev.* **1958**, *65*, 386–408. [CrossRef] [PubMed]
3. Amari, S.I. A theory of adaptive pattern classifier. *IEEE Trans. Electron. Comput.* **1967**, *EC-16*, 279–307. [CrossRef]
4. Golden, R.M. *Mathematical Methods for Neural Network Analysis and Design*; The MIT Press: Cambridge, MA, USA, 1996.
5. Fergus, P.; Chalmers, C. *Applied Deep Learning—Tools, Techniques, and Implementation*; Springer: Cham, Switzerland, 2022.
6. Hinton, G. How to Represent Part-Whole Hierarchies in a Neural Network. *Neural Comput.* **2023**, *35*, 413–452. [CrossRef] [PubMed]
7. MacKay, D.J.C. *Information Theory, Inference, and Learning Algorithms*; Cambridge University Press: Cambridge, UK, 2003.
8. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
9. Werbos, P.J. Applications of advances in: Nonlinear sensitivity analysis. In *System Modeling and Optimization*; Drenick, R., Kozin, F., Eds.; Springer: Berlin/Heidelberg, Germany, 1982.
10. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning Representations by Back-propagating Errors. In *Neurocomputing: Foundations of Research*; MIT Press: Cambridge, MA, USA, 1988.
11. Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; Lerer, A. Automatic differentiation in: PyTorch. In Proceedings of the 31st Conference on Neural Information Processing Systems NIPS, Long Beach, CA, USA, 4–9 December 2017.
12. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
13. Sahin, H.M.; Miftahushudur, T.; Grieve, B.; Yin, H. Segmentation of weeds and crops using multispectral imaging and CRF-enhanced U-Net. *Comput. Electron. Agric.* **2023**, *211*, 107956. [CrossRef]
14. Yan, G.; Jing, H.; Li, H.; Guo, H.; He, S. Enhancing Building Segmentation in Remote Sensing Images: Advanced Multi-Scale Boundary Refinement with MBR-HRNet. *Remote Sens.* **2023**, *15*, 3766. [CrossRef]
15. Min, B.; Ross, H.; Sulem, E.; Veyseh, A.P.B.; Nguyen, T.H.; Sainz, O.; Agirre, E.; Heinz, I.; Roth, D. Recent Advances in Natural Language Processing via Large Pre-Trained Language Models: A Survey. *arXiv* **2021**, arXiv:2111.01243.
16. Majumdar, S.; Ginsburg, B. MatchboxNet: 1D Time-Channel Separable Convolutional Neural Network Architecture for Speech Commands Recognition. *arXiv* **2020**, arXiv:2004.08531v2.
17. Chung, J.S.; Nagrani, A.; Zisserman, A. VoxCeleb2: Deep Speaker Recognition. *arXiv* **2018**, arXiv:1806.05622.
18. Han, J.; Landini, F.; Rohdin, J.; Diez, M.; Burget, L.; Cao, Y.; Lu, H.; Cernocky, J. DiaCorrect: Error Correction Back-end For Speaker Diarization. *arXiv* **2023**, arXiv:2309.08377.
19. Chang, X.; Skarbek, W. Multi-Modal Residual Perceptron Network for Audio-Video Emotion Recognition. *Sensors* **2021**, *21*, 5452. [CrossRef]
20. Reis, D.; Kupec, J.; Hong, J.; Daoudi, A. Real-Time Flying Object Detection with YOLOv8. *arXiv* **2023**, arXiv:2305.09972.
21. Bridle, J.S. *Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition*; Soulié, F.F., Hérault, J., Eds.; Neurocomputing. NATO ASI Series; Springer: Berlin/Heidelberg, Germany, 1990; Volume 68, pp. 227–236.
22. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2006.
23. Mohri, M.; Rostazadeh, A.; Talwalkar, A. *Foundations of Machine Learning*; The MIT Press: Cambridge, MA, USA, 2012.
24. Cho, J.H.; Hariharan, B. On the Efficacy of Knowledge Distillation. *arXiv* **2019**, arXiv:1910.01348.
25. LeCun, Y.; Cortes, C.; Burges, C.J.C. THE MNIST DATABASE of Handwritten Digits. 1998. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 11 November 2013).
26. Bendersky, E. The SoftMax Function and Its Derivative. Available online: https://eli.thegreenplace.net/2016/the-softmax-function-and-its-derivative/ (accessed on 11 November 2013).
27. Golub, G.H.; Van Loan, C.F. *Matrix Computations*, 3rd ed.; Johns Hopkins University Press: Baltimore, MD, USA, 1996.
28. Liu, S.; Leiva, V.; Zhuang, D.; Ma, T.; Figueroa-Zúñiga, J.I. Matrix differential calculus with applications in the multivariate linear model and its diagnostics. *J. Multivar. Anal.* **2022**, *188*, 104849. [CrossRef]
29. Gao, B.; Lacra, P. On the Properties of the SoftMax Function with Application in Game Theory and Reinforcement Learning. *arXiv* **2018**, arXiv:1704.00805.

30. Skarbek, W. Symbolic Tensor Neural Networks for Digital Media—From Tensor Processing via BNF Graph Rules to CREAMS Applications. *Fundam. Inform.* **2019**, *168*, 89–184. [CrossRef]
31. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
32. Riley, K.F.; Hobson, M.P.; Bence, S.J. *Mathematical Methods for Physics and Engineering*; Cambridge University Press: Cambridge, UK, 2010.

entropy

MDPI

Article

Automatic Vertebral Rotation Angle Measurement of 3D Vertebrae Based on an Improved Transformer Network

Xing Huo [1], Hao Li [1] and Kun Shao [2,*]

[1] School of Mathematics, Hefei University of Technology, Hefei 230601, China; huoxing@hfut.edu.cn (X.H.); 2021111431@mail.hfut.edu.cn (H.L.)
[2] School of Software, Hefei University of Technology, Hefei 230601, China
* Correspondence: shaokun@hfut.edu.cn

Abstract: The measurement of vertebral rotation angles serves as a crucial parameter in spinal assessments, particularly in understanding conditions such as idiopathic scoliosis. Historically, these angles were calculated from 2D CT images. However, such 2D techniques fail to comprehensively capture the intricate three-dimensional deformities inherent in spinal curvatures. To overcome the limitations of manual measurements and 2D imaging, we introduce an entirely automated approach for quantifying vertebral rotation angles using a three-dimensional vertebral model. Our method involves refining a point cloud segmentation network based on a transformer architecture. This enhanced network segments the three-dimensional vertebral point cloud, allowing for accurate measurement of vertebral rotation angles. In contrast to conventional network methodologies, our approach exhibits notable improvements in segmenting vertebral datasets. To validate our approach, we compare our automated measurements with angles derived from prevalent manual labeling techniques. The analysis, conducted through Bland–Altman plots and the corresponding intraclass correlation coefficient results, indicates significant agreement between our automated measurement method and manual measurements. The observed high intraclass correlation coefficients (ranging from 0.980 to 0.993) further underscore the reliability of our automated measurement process. Consequently, our proposed method demonstrates substantial potential for clinical applications, showcasing its capacity to provide accurate and efficient vertebral rotation angle measurements.

Keywords: predictive models; deep learning; attention works; point cloud; automatic measurement

Citation: Huo, X.; Li, H.; Shao, K. Automatic Vertebral Rotation Angle Measurement of 3D Vertebrae Based on an Improved Transformer Network. *Entropy* **2024**, *26*, 97. https://doi.org/10.3390/e26020097

Academic Editor: Ercan Kuruoglu

Received: 27 November 2023
Revised: 17 January 2024
Accepted: 19 January 2024
Published: 23 January 2024

1. Introduction

Adolescent idiopathic scoliosis, characterized by complex three-dimensional deformities involving coronal, sagittal, and axial imbalances, demands precise measurement for comprehensive evaluation. Accurate assessment is crucial for gauging scoliosis severity, choosing the most appropriate treatment strategies, and effectively monitoring disease progression. Notably, vertebral rotation plays a pivotal role in the enigmatic pathogenesis of scoliosis [1]. Vrtovec [2] introduced a quantitative method automating the identification of the vertebral body's median plane, enhancing rotation evaluation. Wang [3] validated a novel three-dimensional ultrasound technique for quantifying vertebral rotation, highlighting its practicality. Recently, the orthopedic community has increased its focus on axial plane rotational deformities, reflecting heightened interest in this aspect [4,5]. This collective research underscores the growing importance of accurately measuring vertebral rotation, enhancing our understanding of scoliosis etiology and clinical management.

In cases of spinal curvature, CT images may obscure parts of vertebral bodies and pedicles, necessitating observations from multiple angles. Identifying individual pedicles on a curved spine using 2D images can be challenging [6,7]. However, a 3D vertebral model offers a promising solution, reducing potential CT imaging errors and providing a better understanding of spinal structures. Utilizing spatial angle assessments on these models can yield highly precise scoliosis angle measurements, which is crucial for guiding scoliosis treatment.

Previous methods relied on spinous process positions or pedicle shadows in 2D images, limiting angle estimations. In contrast, our proposed technique directly employs a 3D vertebral model, accurately establishing a vertebra's local coordinate system by incorporating data from end plates and pedicle positions. This innovation can enhance angle measurements and improve the precision of spinal assessments, with the primary aim of reducing errors resulting from vertebral asymmetry, leading to more reliable and accurate results.

The key contributions of our work are as follows:

(1) We propose a novel clinically relevant method for measuring vertebral rotation angles. Addressing the limitations of traditional two-dimensional imaging techniques, we introduce an automated angle measurement approach specifically designed for three-dimensional vertebral models. This represents a significant advancement over traditional two-dimensional imaging techniques;

(2) We propose a transformer-based point cloud segmentation network that incorporates distance distribution entropy into the corresponding point cloud downsampling algorithm. It is noteworthy that our approach achieves significant progress in predicting the upper and lower endplates of specific vertebrae.

2. Related Work

2.1. Point Cloud Processing

Convolutional neural networks (CNNs) have significantly expanded their role in recognizing and segmenting geometric data, achieving breakthroughs in image classification [8]. However, 3D point cloud data lacks the standardized alignment principles of images due to its direct embedding in three-dimensional space.

A pioneering step in this direction was taken by Charles R. Qi et al., who introduced the PointNet network architecture, later refined as PointNet++ [9,10]. PointNet was the first deep neural network capable of directly processing raw point clouds, performing tasks such as point cloud classification and semantic segmentation using inherent point cloud data. Subsequently, researchers addressed PointNet's limitations and developed methodologies for directly utilizing point cloud data in various point cloud analysis applications [11], emphasizing the interrelationships between points within the PointNet++ framework. Mao et al. [12] have proposed a new interpolation convolution operation, InterpConv, to address the learning and understanding challenges of point cloud features. To address the challenge of Multilayer Perceptrons (MLPs) inadequately capturing the geometric structure and contextual information of point clouds, Xie et al. [13] proposed GRNet. They regularized unordered point clouds into a 3D grid, ultimately achieving improved performance in point cloud completion tasks.

Recent years have seen the integration of convolution into point cloud data for point cloud segmentation [14–17]. Attention mechanisms [18], particularly self-attention mechanisms, have proven effective in capturing contextual information, making them suitable for point cloud processing tasks. The emergence of Point Transformer (PT) [19] and Point Cloud Transformer (PCT) [20] represents successful models for point cloud segmentation leveraging the transformer mechanism.

2.2. Vertebral Rotation Angle Measurement

The advent of deep learning technology has brought forth a novel wave of interdisciplinary applications across various engineering domains. In the realm of vertebral rotation angle measurement, neural network-based methodologies have garnered significant attention for automating the segmentation of vertebral bodies and pedicles. Noteworthy contributions include the work of Bakhous et al. [21], who introduced a CNN-based regression model aimed at enhancing vertebral pedicle localization and the estimation of vertebral rotation angles. Veena Logithasan et al. [22] developed a CNN-powered machine learning algorithm to autonomously compute the AVR (Apical Vertebral Rotation) on PAx radiographs using the Stokes method. Shahin Ebrahimi et al. [23] devised an automated pedicle detection system, employing image analysis, machine learning, and rapid manual landmark identification, serving as a quantitative VAR (Vertebral Apical Rotation) assessment tool for scoliosis patients. Zhang et al. [24] pioneered a computer-aided approach, integrating Hough transform and snake model techniques to semi-automatically gauge the Cobb angle and vertebral rotation on PAx radiographs. Devlin G. et al. [25] conducted spinal parameter measurements and correlation evaluations using standing PA radiographs, employing the Stokes method for vertebral rotation angle quantification. Quang N. Vo et al. [26] explored the reliability and accuracy of AVR measurements utilizing centerpoints from the vertebral body or transverse process on three-dimensional ultrasound images. Daniel Forsberg et al. [27] devised a fully automated approach for estimating AVR through images from computed tomography scans. An examination of recent research into vertebral rotation angle measurement methods reveals that a predominant focus remains on automated measurements derived from 2D CT images. However, these 2D approaches are susceptible to projection bias and often overlook the patient's transverse plane. These conventional 2D techniques fail to fully capture the intricacies of three-dimensional spinal curvature deformities, particularly in the context of vertebral axial rotation, which holds considerable significance.

2.3. EOS Imaging System

The EOS imaging system is a cutting-edge, low-radiation, and highly accurate method for assessing spinal curvature. Some studies [28–31] have used EOS images to create comprehensive 3D spine reconstructions, forming the basis for measuring vertebral rotation angles with specialized software. While EOS imaging has been used to reconstruct 3D vertebral models in certain studies, measuring vertebral rotation angles often involves time-consuming manual point marking. In contrast, our method allows for direct and automated determination of these angles from the 3D models. Importantly, our automated approach shows strong agreement with manual measurements, enhancing its suitability for clinical assessments.

3. Materials and Methods

3.1. Point Cloud Sampling Method

In the context of point cloud downsampling, selecting a subset of original points to represent the entire point cloud proves effective in reducing the overall number of points, thereby minimizing storage and processing costs. Common downsampling methods encompass random sampling, voxel downsampling, grid downsampling, adaptive sampling, and other techniques.

To attain a uniformly distributed point cloud, we opt for the Farthest Point Sampling (FPS) algorithm. However, given the uneven density in vertebral point cloud data, particularly in regions such as vertebral bodies and arches, the traditional FPS sampling algorithm may struggle to achieve uniform point sampling. In response to this challenge, we implement an enhanced version of the FPS algorithm, integrating distance distribution entropy for improved results. Distance distribution entropy provides a more comprehensive understanding of the distance distribution within the point cloud. By considering the distance distribution of points and their surroundings, it allows for a more intelligent selection of the farthest sampling points, thereby enhancing the informativeness of the sampling process.

For a given point cloud, the first step is to calculate the distances between each point and every other point, forming a distance matrix D. The calculation is as follows:

$$H = -\sum_{i}(P_i log(P_i + \epsilon)) \tag{1}$$

In this formula, firstly, the distance matrix D is flattened into a one-dimensional array, removing distances on the diagonal (i.e., distances from points to themselves), resulting in an array D' containing distances for all pairs of points. Next, the array D' is discretized into a histogram to obtain a frequency distribution P.

3.2. Local Information Extraction Module

The self-attention mechanism, initially proposed for assessing word correlations in different positions, is well-suited for connecting various positions within local point clouds. This adaptability is a significant factor in the success of transformers in handling 3D point clouds.

Once the 3D point cloud is divided into blocks using the KNN algorithm, the self-attention operator is employed to calculate the output features F_{SA} from the corresponding input features F_{in} within a local field F. The calculation is as follows:

$$F_{SA} = softmax(\frac{QK^{\top}}{\sqrt{d_a}})(V) \tag{2}$$

where Q, K and V are the query, key, and value metrics, and d_a is the dimension of the query and key vectors.

Due to rigid transformations, the absolute coordinates of the same point cloud can vary significantly. To address this, we introduce a position encoding function δ:

$$\delta = \phi(xyz) \tag{3}$$

where ϕ is an MLP with two layers and one ReLU nonlinearity.

$$F_{in} = \varphi(concat(f,\delta)) \tag{4}$$

where φ is a linear layer, and we do this by concatenating the original features and the transformed position encoding information as the final input features.

In the original model, the Self-Attention (SA) module is enhanced with the Relation Attention (RA) mechanism to improve point cloud feature representation. This is accomplished by using the Laplace matrix L (where $L = D - A$) to replace the adjacency matrix A in the graph convolution network, where D is the diagonal matrix. The deviation between the self-attention feature and the input feature is calculated through element-wise subtraction:

$$F_{RA} = F_{in} - F_{SA} \tag{5}$$

where F_{in} is the features of the input, F_{SA} is the features obtained by self attention transformation, and F_{RA} enhances the feature representation of the point cloud.

Then the characteristics of the final output F_{out} can be expressed as:

$$F_{out} = RA(F_{in}) = LBR(F_{RA}) + F_{in} \tag{6}$$

where LBR is a network that combines Linear, BatchNorm and ReLU layers. Figure 1 shows the local information extraction module.

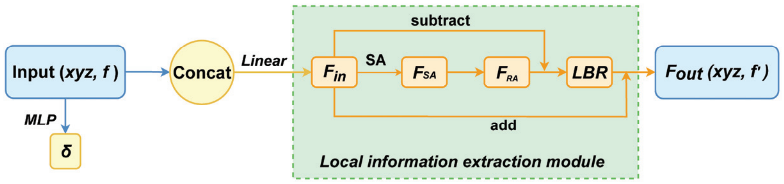


Figure 1. Relative attention (RA), xyz are the coordinates of the input clouds, f is the original feature of the input, LBR combines Linear, BatchNorm, and ReLU layers, and SA is the self attention.

3.3. Architecture Design

The Transformer's self-attention mechanism excels in point cloud tasks, outperforming PointNet models in benchmarks. Farthest Point Sampling (FPS) ensures comprehensive point cloud coverage and uniformity. In our experiments, FPS sampled points to predict vertebral body upper and lower plate locations using a trained network, reducing the point count. Upsampling for point cloud segmentation follows, as shown in Figure 2.

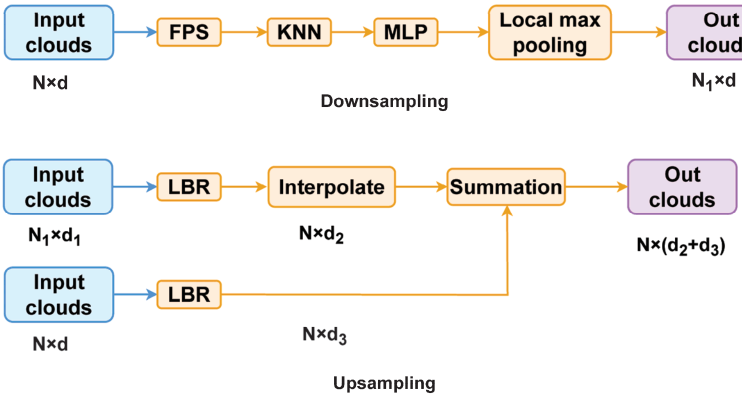


Figure 2. Downsampling and Upsampling. N_1 and N is the number of point clouds, d, d_1, $d_2$and d_3 is the corresponding dimension.

We adopt the U-Net framework [32], renowned for its excellence in medical image segmentation. For upsampling, we utilize the U-Net's deconvolution method. After each downsampling, interpolation points preserve previous feature information. Local-global features from skip connections are maintained to ensure invariance. This seamless fusion of local and global features helps mitigate limitations compared to simple concatenation.

Our network begins with input point clouds, expanding dimensionally to 64 through an MLP. It goes through four downsampling stages and four upsampling stages, with the Relation Attention (RA) module applied after each downsampling step.

The input embedding projects the point cloud into a higher-order feature space using an MLP. Downsampling utilizes local max-pooling to capture local-global features, preventing the information loss associated with direct global pooling. Upsampling employs trilinear interpolation, while skip connections aggregate information.

Our network design, similar to U-Net, incorporates skip connections to enable continuous integration of global and local features, enhancing discriminative feature generation. Figure 3 provides an illustration of our network's framework.

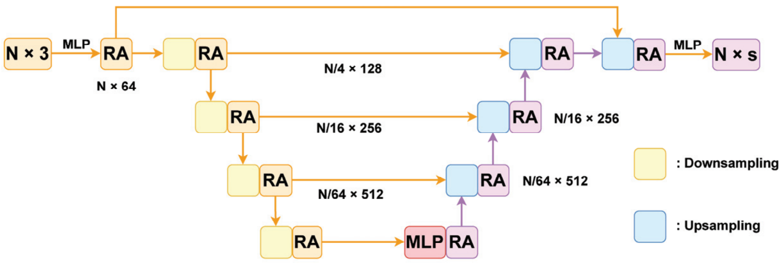


Figure 3. The network. N is the number of point clouds input to the network, RA is the local information extraction module, MLP is the multi-layer perceptron, and s is the number of point cloud categories.

3.4. The Measure of Vertebral Rotation Angle

The algorithmic steps for the automated measurement of vertebral rotation angles are as follows: First, import the vertebral model and perform point cloud sampling. Next, for the sampled vertebral point cloud, utilize a pre-trained neural network model to predict the corresponding point clouds for the vertebral endplates and pedicle roots. Finally, based on the obtained predicted point clouds, we conduct the measurement of the corresponding point cloud's rotation angles. Figure 4 shows the procedure of the algorithm to calculate the vertebral rotation angle.

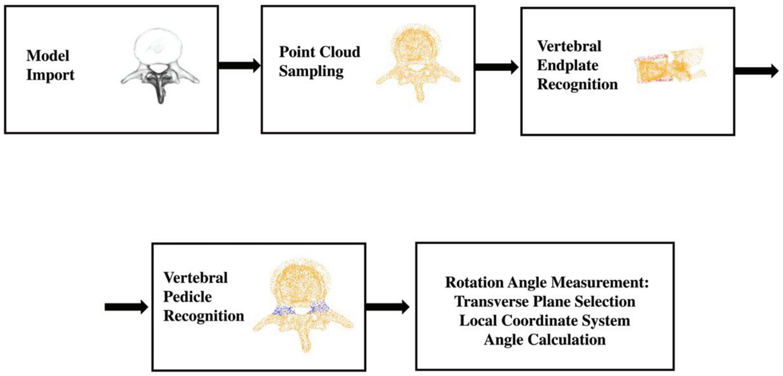

Model Import
Point Cloud Sampling
Vertebral Endplate Recognition
Vertebral Pedicle Recognition
Rotation Angle Measurement: Transverse Plane Selection Local Coordinate System Angle Calculation

Figure 4. Procedure of the algorithm to calculate the vertebral rotation angle.

3.4.1. Model Import and Point Cloud Sampling

By utilizing vtk's three-dimensional reconstruction functionality, we obtained a three-dimensional spine model based on CT images. Subsequently, we employed vtk's bounding box functionality for manual segmentation, obtaining the corresponding individual vertebra. The respective vertebra models can then be converted into point cloud-formatted data. For the acquired three-dimensional point cloud data, we applied a furthest point sampling algorithm in conjunction with distance distribution entropy, resulting in uniformly distributed point cloud data for the vertebrae.

3.4.2. Vertebral Endplate and Pedicle Recognition

We use our network to train on an existing dataset of vertebral endplate point clouds and then apply this trained model to predict new vertebral endplate point clouds. After generating these predicted point clouds, we divide them into two distinct sections based on their spatial distribution. We then determine the centers of these individual segments. Figure 5 displays the expected point clouds for the upper and lower endplates of the vertebrae. The point cloud highlighted in red represents the outcome of our network's prediction for the vertebral endplates, while the green spheres in the image represent the corresponding endplate centers.

We employ our network to train the generated pedicle dataset. For the anticipated pedicle point cloud, we apply the K-means clustering algorithm to partition this particular section of the point cloud into two distinct segments. Subsequently, the corresponding cluster centers are identified. In Figure 6, the highlighted red region represents the point cloud of the vertebral pedicles predicted by our model, while the green spheres symbolize the corresponding pedicle centers obtained through the K-Means clustering algorithm.

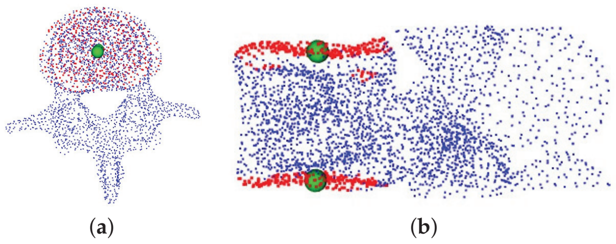


Figure 5. Recognition of vertebral endplate point clouds. (**a**) The top view of the vertebrae; (**b**) the corresponding side view.

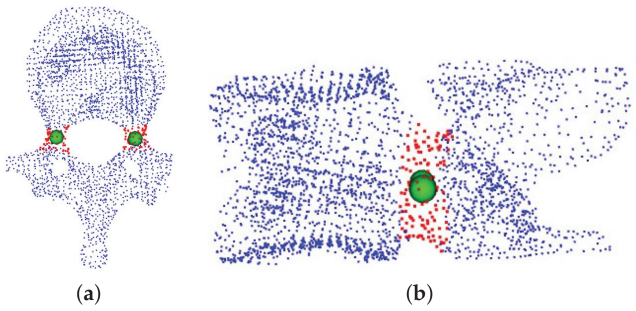


Figure 6. Recognition of the point clouds of vertebral pedicle. (**a**) The top view of the vertebrae; (**b**) the corresponding side view.

3.4.3. Vertebral Rotation Angle Measurement

To measure the vertebral rotation angle in the 3D vertebral model, we follow a specific methodology:

(1) Transverse Plane Selection: The vertebral centroid, positioned at the midpoint of the line connecting the midpoints of the upper and lower endplates of the vertebrae, serves as the center of the transverse plane. The normal vector of this plane is defined by the line connecting the midpoints of the upper and lower endplates. Figure 7a depicts the vertebra's transverse plane;

(2) Local Coordinate System: We establish a local coordinate system with the vertebra centroid as the origin point. In this coordinate system, the Y-axis is defined by the line connecting the midpoint of the pedicle and the vertebra centroid. The vector connecting the upper and lower endplates defines the z-axis, while the x-axis is computed using the cross product of the vectors representing the y- and z-axes. Figure 7b illustrates this sequential process;

(3) Angle Calculation: Figure 8 illustrates the method for measuring vertebral rotation angles. This angle is formed between the vector representing the projection of the local coordinate system's Y-axis onto the global coordinate system's Y-axis within the vertebra's transverse plane. Following this methodology, we accurately determine the vertebral rotation angle in the 3D vertebral model. The red and green lines represent the respective projected vectors of the local coordinate system and the global coordinate system's Y-axis on the transverse plane.

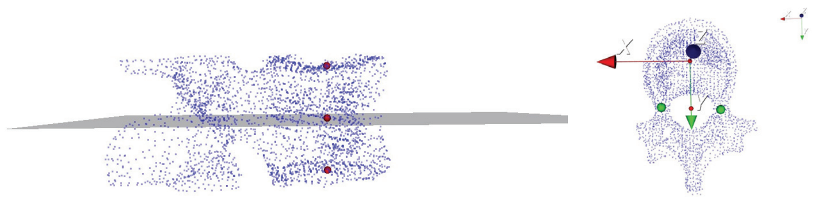

(**a**) (**b**)

Figure 7. (**a**) The transverse plane of the vertebrae; (**b**) the local coordinate system of the vertebra.

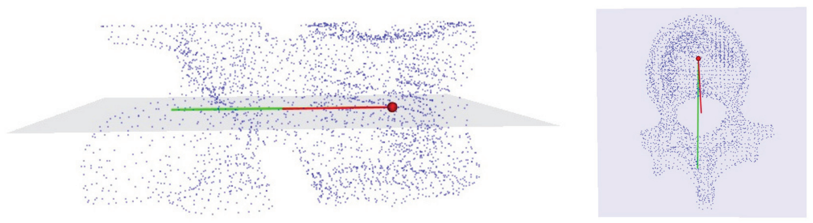

(**a**) (**b**)

Figure 8. (**a**) The measurement of the vertebral rotation angle; (**b**) the corresponding side views.

4. Experimental Results and Comparisons

In this section, we evaluate the network used in this paper, compare it with other networks, and apply it to the vertebral rotation angle measurement.

4.1. Data Preparation

The dataset used in our study is sourced from SpineWeb (dataset3), a freely accessible online resource. Our approach involves downloading the data to facilitate our research endeavors. Each CT image sequence is equipped with element spacing details, accessible directly from the file header. To construct the 3D vertebral model, we harness the power of VTK, an open-source toolkit renowned for its capabilities in three-dimensional reconstruction. This process involves the aggregation of sequential CT images of the vertebrae, enabling us to generate a comprehensive three-dimensional spine model.

Subsequently, we capitalize on VTK's functionalities to undertake manual segmentation on the model. This intricate segmentation process enables us to precisely isolate and select the specific vertebra of interest from the three-dimensional spine model. The resulting selection serves as the fundamental data source for our analysis, providing a robust foundation for our research efforts.

4.1.1. Generate Point Cloud Data for Vertebral Endplate and Pedicle

Regarding the obtained vertebral model, we perform manual labeling using the marking functionality in vtk. Specifically, we allocate a label of 1 to the point cloud data corresponding to the marked endplates. In contrast, for the remaining point cloud data, we assign a label of 0. The visual depiction of this procedure, illustrating the manual marking of the vertebrae model plane, is presented in Figure 9.

In Figure 9, we provide a comprehensive guide on how to execute the manual labeling process for the upper and lower endplate planes of the vertebrae. This process was facilitated using the versatile vtk open-source toolkit. The left side of Figure 9 illustrates the steps involved in manual labeling using vtk. Following this labeling, we proceed to attribute a label of 1 to the resultant point cloud and designate the label 0 for the remaining point cloud data, as depicted on the right side of Figure 9. This thorough process ensures the accurate distinction of the labeled endplates and sets the stage for subsequent analyses.

Figure 9. Manual marking of the vertebrae model plane.

Furthermore, we conduct the extraction of data from the vertebral pedicle within the vertebral bone model. This extraction process is primarily facilitated through the utilization of the bounding box feature provided by the VTK toolkit. The operational process of employing the VTK bounding box to effectively capture the vertebrae aligned with the pedicle is graphically depicted in Figure 10.

To elucidate, we designate a label of 1 for the point cloud data that corresponds to the intercepted pedicle of the vertebrae. Conversely, the remaining point cloud data is assigned a label of 0. This comprehensive labeling process not only precisely differentiates the pedicle regions but also sets the groundwork for subsequent analytical pursuits.

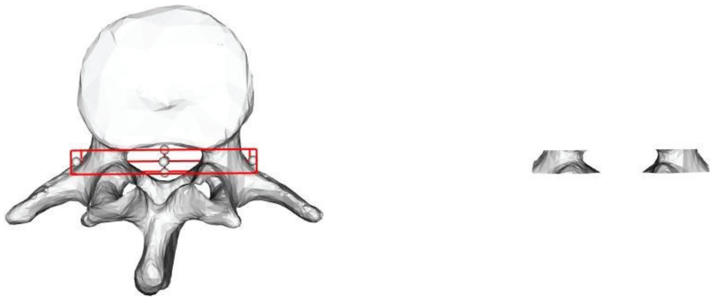


Figure 10. Manual extraction of the pedicle data from the vertebrae model.

4.1.2. Generation Point Clouds and Labels

The training data is generated through the labeling of point clouds corresponding to vertebral endplates, yielding the corresponding training data labels. The presence of multiple point clouds for each vertebral model, coupled with the varying point cloud counts across these models, poses a challenge for directly training subsequent deep learning networks. To address this issue, it becomes necessary to uniformly downsample the point cloud counts to a fixed value, given the varying numbers of point clouds for different vertebral models.

One effective approach for this uniform downsampling is the furthest point sampling algorithm (FPS), which maintains the shape of the sampled point cloud while achieving uniformity in point distribution. By employing FPS, we homogeneously collect labeled point cloud data, transforming it into a structured format comprising 3072 points.

To augment the number of vertebrae within the corresponding labeled dataset, we expand the dataset using two FPS samples drawn from the labeled dataset. This expansion strategy contributes to a more robust and diverse training dataset, thereby enhancing the effectiveness of subsequent deep learning processes.

4.2. Evaluating the Network

First, we evaluate the network using the publicly accessible ShapeNet dataset and conduct comparisons with other networks. The ShapeNet dataset has 16 categories and a total of 16,881 shapes annotated with 50 parts. The Shapenet dataset comprises 3D models from multiple categories, encompassing objects of various shapes and types. This allows for an examination of model performance across diverse object types in comparative studies of segmentation networks. Furthermore, it is widely used as a standard dataset in point cloud research.

We train using the Adam optimizer with a learning rate of 0.003, a decay rate of 0.5, a batch size of 4, and an epoch of 200.

During the training process, we utilize the cross-entropy loss function. We compare the obtained probability distribution from the Softmax function with the true labels and compute the cross-entropy loss. If the true labels are represented as a one-hot encoded vector y, where the i-th element y_i indicates whether the sample belongs to the i-th class, the cross-entropy loss is calculated as:

$$L(y, softmax(z)) = -\sum_{i=1}^{C} y_i log(softmax(z)_i) \tag{7}$$

where the raw output z is a C-dimensional vector, and C is the number of component categories in the point cloud.

IoU represents, in point cloud segmentation, the ratio of the intersection and sum of the true labels and predicted values for the class. mIoU is the mean intersection over union for each class in the dataset.

$$mIoU = \frac{1}{K} * \sum_{k=1}^{K} \frac{TP_k}{TP_k + FP_k + FN_k} * 100\% \tag{8}$$

where K represents the number of categories, TP_k represents the number of correctly predicted point clouds in category k, FP_k represents the number of point clouds that were misclassified as category k, and FN_k represents the number of point clouds where points of category k are misclassified as other categories.

We evaluate our segmentation results on the ShapeNet dataset using the mean intersection over union (mIoU) as our evaluation metric. Table 1 presents these results.

Table 1 reveals that our network's performance does not consistently surpass other point cloud segmentation networks across all ShapeNet dataset categories. However, our method excels in specific categories, particularly those featuring complex structures and smaller, indistinct components. Notably, our network demonstrates exceptional segmentation accuracy when dealing with objects characterized by complex structures, such as the Motor category. Furthermore, due to the complex shape of vertebrae, our network is well-suited for segmenting vertebral data.

Table 1. The result for the part segmentation on the ShapeNet. The metric is mIoU(%) on points.

	Mean	Airplane	Bag	Cap	Car	Chair	Earphone	Guitar	Knife	Lamp	Laptop	Motor	Mug	Pistol	Rocket	Skate Board	Table
PointNet	81.5	79.8	65.6	75.4	68.9	87.9	68.3	88.8	82.2	77.6	94.8	44.2	85.8	75.0	50.7	70.6	81.5
PointNet++	83.6	81.4	75.1	81.0	***76.7***	89.8	***78.2***	90.1	81.5	82.3	95.2	60.8	90.6	78.5	52.9	72.5	81.2
DGCNN	83.9	80.9	65.1	68.4	75.7	***90.3***	69.5	89.8	84.9	***84.5***	95.3	48.3	87.9	74.3	42.9	70.6	***82.9***
Point Transformer	81.4	78.3	74.3	78.4	69.8	88.9	75.4	89.4	83.5	81.7	94.7	60.0	80.2	74.6	49.9	68.5	78.4
Point Cloud Transformer	83.2	***82.0***	67.7	***82.2***	73.0	89.2	76.0	89.6	85.0	80.6	95.2	51.9	89.1	78.1	49.1	69.6	82.1
PointMLP	84.3	81.2	70.7	80.4	76.2	90.1	73.2	90.1	86.6	83.1	***95.5***	56.4	91.1	***80.7***	48.3	***74.6***	82.8
Ours	83.6	79.1	***75.2***	78.6	74.4	89.4	74.6	***90.2***	***86.7***	83.4	95.0	***61.1***	***91.4***	78.5	***54.6***	68.8	82.4

The bold and italicized font indicates the best segmentation performance compared to other networks for that category.

Table 2 illustrates whether the combination of distance distribution entropy with FPS influences the accuracy of predicting the vertebral dataset. From Table 2, it is evident that the results obtained after preprocessing vertebral point cloud data with distance distribution entropy are significantly superior to those obtained without using distance distribution entropy.

Table 2. The result for the part segmentation on the vertebral dataset. The metric is mIoU(%) on points.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
FPS	95.1	95.9	96.5	93.7	96.1	97.0	98.1	96.3	97.3	97.1
FPS + entropy	95.3	96.3	97.1	96.1	96.4	96.6	98.6.5	96.5	97.6	97.3

4.3. Pedicle Recognition Effect

Accurate angle measurement relies on establishing the Y-axis of the local coordinate system through two pedicle centers. Precise pedicle recognition is critical for minimizing potential errors, making the statistical analysis of pedicle recognition essential.

Table 3 presents a randomly selected subset of 10 vertebral model datasets from the complete dataset. Given the intricate spatial structure of the vertebrae, our network demonstrates superior segmentation accuracy compared to alternative networks in the majority of vertebral datasets.

Table 3. The result for the part segmentation on the vertebral dataset. The metric is mIoU(%) on points.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
PointNet	93.7	96.1	96.9	91.5	***97.6***	95.8	94.6	95.4	97.6	96.4
PointNet++	94.7	95.1	96.3	94.5	96.9	96.1	97.5	95.1	97.2	97.5
DGCNN	95.4	96.1	96.3	97.7	97.2	97.1	98.7	96.4	97.4	97.5
Point Transformer	95.3	95.4	97.1	93.4	96.8	96.4	97.2	96.3	97.3	96.7
Point Cloud Transformer	94.6	95.3	96.3	94.8	97.2	95.5	94.6	94.6	97.4	97.2
PointMLP	92.6	95.7	97.1	***98.3***	97.0	97.2	97.3	96.1	97.2	97.2
Ours	***95.6***	***96.2***	***97.3***	94.9	97.2	***97.2***	***98.9***	***96.7***	***97.8***	***97.6***

The bold and italicized numbers indicate the best segmentation performance for the corresponding vertebra.

Figure 11 vividly illustrates the clustering results for segmented vertebral body point clouds using different techniques. The blue point cloud in the figure represents the predicted vertebral pedicle point cloud, and the corresponding black spheres depict the clustering centers obtained through K-means clustering. By comparing it with the manually segmented vertebral pedicle point cloud, it can be observed that our proposed method outperforms other networks in predicting vertebral pedicle point clouds (the predicted blue point cloud is closer to the manually segmented point cloud). Our experiments reveal that K-means effectively groups various methods for obtaining vertebral pedicle point clouds into two clusters, closely aligned with actual pedicle centers. Notably, our method consistently produces clustering centers that closely match manual segmentation, emphasizing its effectiveness and accuracy in this critical analysis.

We conduct experiments on 10 randomly chosen vertebral bone datasets, calculating the displacement distances between their clustering centers and those obtained from manual segmentation. Table 4 displays the average offset distance. We use the distance (mm) as the evaluation metric. Our proposed network generates centroids that closely match manually derived centroids when using the K-means clustering algorithm. These segmentation results underscore the potential of our network in establishing the local coordinate system of vertebrae and computing vertebral rotation angles. This comprehensive assessment demonstrates the effectiveness and promise of our approach for improving vertebral rotation angle measurements, with potential applications in clinical evaluation and diagnosis.

Table 4. Comparison of the average offset distances between the clustering centers of multiple network-based pedicle segmentation results on the test set with respect to the reference cluster centers derived from manually labeled pedicles. We use the distance (mm) as the evaluation metric.

	PointNet	PointNet++	DGCNN	PT	PCT	PointMLP	Ours
Left Pedicle	3.621	1.823	3.947	2.479	1.906	2.075	1.773
Right Pedicle	2.738	1.960	4.520	2.059	2.007	2.560	1.188

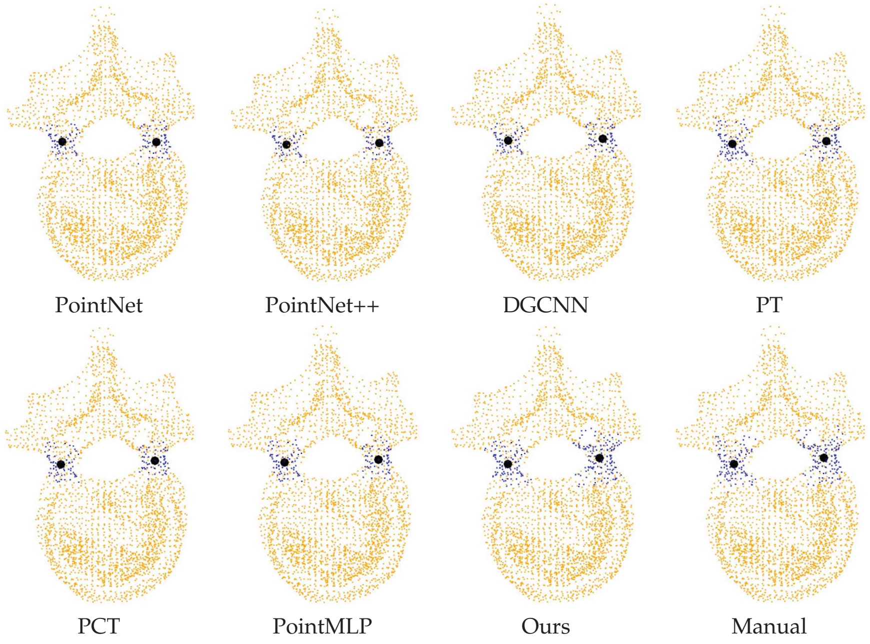


Figure 11. Demonstration of the clustering effect of the pedicle point clouds segmented in different ways.

4.4. Measurement Results

We initially select 10 vertebral models randomly and then conduct measurements using both manual and automated methods. For manual measurements, 3 observers measure each vertebra 10 times, and the results are averaged. Additionally, we employ our automated measurement program to measure each vertebra 10 times.

Manual measurements involve marking the four points of the vertebral endplate (top, bottom, left, and right) and determining the midpoint of these points to establish the endplate's center. We then extract the corresponding point cloud of the pedicle using an envelope box. The K-means algorithm is utilized to find the cluster centers of the left and right pedicles, and the corresponding angles are calculated.

To validate our proposed automated measurement method, we randomly selecte10 vertebral models and conduct both manual and automated measurements, each repeated 10 times. Figure 12 visualizes the measurement variability using Bland–Altman plots, which display a total of 100 points distributed across 10 distinct patterns. Each point represents a measurement pair, and points within the same pattern correspond to the same study subject. Importantly, points within the same pattern cluster closely together, indicating consistency within each subject. It is noteworthy that only 4 points (4% of the data) fall outside the limits of agreement (LoA).

This outcome suggests that there is no significant systematic difference when comparing differences between manual and automated measurements. The majority of measurements closely align, affirming the validity of our proposed automated measurement method.

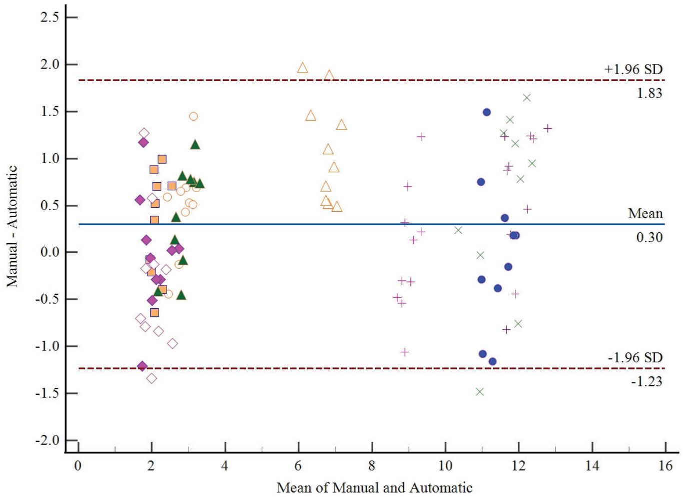


Figure 12. Bland–Altman plots (i.e., the difference between two measurements plotted against their mean) for measurements from manual and automatic measurements. Additional horizontal lines correspond to the mean difference of the measurements (dashed-dotted) and to the lines of agreement (dashed), i.e., the 95% CI for the difference of the measurements. The 10 different patterns in the figure represent 10 different vertebral models.

Table 5 shows the final results of the manual measurements as well as the average of the automatic measurements. The table presents the mean of 10 randomly selected vertebral models subjected to 10 manual measurements by 3 observers and the mean of 10 automated measurements. The analysis of the error in the automatic measurement of the angle is mainly due to the fact that the center of the upper and lower endplate point cloud of the predicted vertebrae is used in the generation of the vertebral medial surface, which may lead to deviations in the final angle measurement due to incomplete segmentation of the vertebral endplate point cloud. To address this issue we took multiple automated measurements to average the results of their automated measurements.

Table 5. Manual and automatic measurement results. The metric is °.

Model	Observe1	Observe2	Observe3	Automatic
1	3.57	2.69	3.11	2.62
2	12.85	12.33	12.12	11.70
3	11.31	10.85	11.87	11.40
4	12.38	12.78	10.53	11.35
5	8.95	8.81	9.23	8.99
6	7.44	7.47	7.08	6.20
7	2.44	2.64	1.92	2.01
8	2.46	3.57	3.20	2.67
9	2.20	2.30	1.67	2.09
10	2.34	1.94	1.36	2.19

To evaluate the accuracy of the automatic angle measurement, we use the evaluation metric ICC, and the model we use is a two-way mixed model of multiple measurements and absolute agreement (Table 6). Its calculation formula is as follows:

$$ICC = \frac{MS_R - MS_E}{MS_R + \frac{MS_C - MS_E}{n}} \tag{9}$$

where MS_R represents the mean square for rows, MS_E represents the mean square for error, MS_C represents the mean square for columns, and n represents the number of subjects.

Table 6. ICC using a two-way mixed model with absolute agreement, comparing the correlation in-between manual measurements and automatic measurements.

	ICC	**95% CI**
Automatic-Observe1	0.987	[0.904, 0.997]
Automatic-Observe2	0.985	[0.929, 0.996]
Automatic-Observe3	0.991	[0.967, 0.998]
Observe1-Observe2	0.993	[0.971, 0.998]
Observe1-Observe3	0.983	[0.933, 0.996]
Observe2-Observe3	0.980	[0.925, 0.995]

The mean differences and the 95% confidence intervals between the manual measurements were: $0.56° \pm 0.40°$, $-0.39° \pm 0.54°$, $0.33° \pm 0.63°$(Observe1-Observe2, Observe1-Observe3, Observe2-Observe3). The results between automatic and manual measurements were: $-0.47° \pm 0.47°$, $-0.42° \pm 0.47°$, $-0.09° \pm 0.43°$(Automatic-Observe1, Automatic-Observe2, Automatic-Observe3). The difference between automatic measurement and manual measurement is at the same level as the difference between manual measurements performed by different observers.

Using the intraclass correlation coefficients in Table 6, we can see that the ICC ranges between 0.980 and 0.993, which indicates that there is a high level of consistency between manual measurements and the automated measurements proposed by us.

4.5. Performance and Results Analysis

Comparative evaluations between our automated measurement method and manual measurements reveal a high level of agreement, supported by the Bland–Altman plots and ICC values. The strong agreement observed between automatic and manual measurements, as well as among different manual observers, suggests that our automated method has the potential to effectively replace manual measurements. However, Table 6 indicates a slight decrease in ICC values between automatic and manual measurements and among different manual measurements. This decrease is attributed to the estimated offset of the vertebral body center and the corresponding offset of the pedicle center.

Our local information extraction module, incorporating location codes into original input features and computing local features using our proposed relation attention mechanism, plays a pivotal role in our approach. The experimental results highlight our network's superiority in segmenting specific object parts, particularly excelling in segmenting vertebral point cloud data, including the upper and lower endplates of vertebrae and partial point clouds of the pedicle. Comparative analyses with other networks consistently demonstrate our network's superior performance in both endplate and pedicle segmentation.

However, our experiment has some limitations, one of which stems from the computational load introduced by incorporating a local point cloud information extraction module in our network. While this addition results in a certain improvement in training accuracy, it also increases the computational workload during the training process, thereby impacting the operational efficiency of the network. Therefore, in the future, we plan to explore modifications to this local information extraction strategy to enhance the operational efficiency of the network.

Another limitation stems from our approach to establish the symmetry plane of vertebrae using centroids based on upper and lower endplate point clouds. This approach may encounter challenges with incomplete segmentation of the vertebrae point cloud, such as partial endplate segmentation. In cases of uneven segmentation, our method, while improving endplate recognition to some extent, might produce incorrect centroid predictions due to non-uniform segmentation.

To address this limitation, future efforts should aim to enhance the vertebrae point cloud segmentation network by incorporating inherent features of vertebrae point clouds. This customized approach could lead to more accurate segmentation results, effectively addressing the unique characteristics of vertebrae models. Additionally, modifying the algorithm to incorporate insights from point cloud completion and other techniques could improve measurement accuracy for incomplete vertebrae point cloud models. These refinements will contribute to a more robust and comprehensive measurement methodology suitable for complex real-world scenarios in clinical applications.

5. Conclusions

In this paper, we successfully introduce an automated measurement method based on three-dimensional vertebral models, particularly suitable for evaluating spinal deformities in idiopathic scoliosis patients. Compared to traditional manual measurement methods, our automated approach demonstrates significant advantages in terms of accuracy and efficiency. Through comparisons with manual measurements, we validate the consistency of our automated measurement method among different observers and showcase its superior performance in vertebral model segmentation. Despite the significant success of our method, we acknowledge some limitations in the experiment. In future research, we focus on improving the local information extraction strategy to enhance the operational efficiency of the network. Additionally, efforts are directed towards further optimizing the vertebrae point cloud segmentation network to overcome challenges associated with incomplete segmentation. We believe these improvements will facilitate the application of our method in complex clinical scenarios, providing a more accurate and reliable tool for the assessment of spinal deformities.

Author Contributions: Conceptualization, X.H.; methodology, X.H. and K.S.; validation, H.L.; writing—original draft preparation, H.L.; writing—review and editing, H.L.; funding acquisition, X.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China under Grant 61872407.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Data is contained within the article

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

References

1. Adam, C.J.; Askin, G.N. Automatic measurement of vertebral rotation in idiopathic scoliosis. *Spine* **2006**, *31*, E80–E83. [CrossRef]
2. Vrtovec, T.; Pernuš, F.; Likar, B. A symmetry-based method for the determination of vertebral rotation in 3D. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008: 11th International Conference, New York, NY, USA, 6–10 September 2008; Proceedings, Part I 11; Springer: Berlin/Heidelberg, Germany, 2008; pp. 942–950. [CrossRef]
3. Wang, Q.; Li, M.; Lou, E.H.; Chu, W.C.; Lam, T.p.; Cheng, J.C.; Wong, M.s. Validity study of vertebral rotation measurement using 3-D ultrasound in adolescent idiopathic scoliosis. *Ultrasound Med. Biol.* **2016**, *42*, 1473–1481. [CrossRef]
4. Lau, K.K.L.; Kwan, K.Y.H.; Cheung, J.P.Y.; Chow, W.; Law, K.K.P.; Wong, A.Y.L.; Chow, D.H.K.; Cheung, K.M.C. Reliability of a three-dimensional spinal proprioception assessment for patients with adolescent idiopathic scoliosis. *Eur. Spine J.* **2022**, *31*, 3013–3019. [CrossRef] [PubMed]
5. van Royen, B.B. Understanding the Lenke Classification for Adolescent Idiopathic Scoliosis (AIS). *Curr. Probl. Diagn. Radiol.* **2023**, *52*, 233–236. [CrossRef] [PubMed]
6. Lee, J.; Kim, S.; Kim, Y.S.; Chung, W.K. Automated segmentation of the lumbar pedicle in CT images for spinal fusion surgery. *IEEE Trans. Biomed. Eng.* **2011**, *58*, 2051–2063. [CrossRef] [PubMed]

7. Kumar, S.; Nayak, K.P.; Hareesh, K. Semiautomatic method for segmenting pedicles in vertebral radiographs. *Procedia Technol.* **2012**, *6*, 39–48. [CrossRef]
8. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
9. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660. [CrossRef]
10. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv* **2017**, arXiv:1706.02413.
11. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph. (Tog)* **2019**, *38*, 1–12. [CrossRef]
12. Mao, J.; Wang, X.; Li, H. Interpolated convolutional networks for 3d point cloud understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27–28 October 2019; pp. 1578–1587.
13. Xie, H.; Yao, H.; Zhou, S.; Mao, J.; Zhang, S.; Sun, W. Grnet: Gridding residual network for dense point cloud completion. In Proceedings of the European Conference on Computer Vision. Springer, Glasgow, UK, 23–28 August 2020; pp. 365–381.
14. Ma, X.; Qin, C.; You, H.; Ran, H.; Fu, Y. Rethinking network design and local geometry in point cloud: A simple residual MLP framework. *arXiv* **2022**, arXiv:2202.07123.
15. Qian, G.; Li, Y.; Peng, H.; Mai, J.; Hammoud, H.; Elhoseiny, M.; Ghanem, B. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 23192–23204. [CrossRef]
16. Wang, J.; Chakraborty, R.; Stella, X.Y. Transformer for 3D Point Clouds. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 4419–4431. [CrossRef]
17. Zhu, G.; Zhou, Y.; Zhao, J.; Yao, R.; Zhang, M. Point cloud recognition based on lightweight embeddable attention module. *Neurocomputing* **2022**, *472*, 138–148. [CrossRef]
18. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
19. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 16259–16268. [CrossRef]
20. Guo, M.H.; Cai, J.X.; Liu, Z.N.; Mu, T.J.; Martin, R.R.; Hu, S.M. Pct: Point cloud transformer. *Comput. Vis. Media* **2021**, *7*, 187–199. [CrossRef]
21. Bakhous, C.; Aubert, B.; Vazquez, C.; Cresson, T.; Parent, S.; De Guise, J. Automatic pedicles detection using convolutional neural network in a 3D spine reconstruction from biplanar radiographs. In Proceedings of the Medical Imaging 2018: Computer-Aided Diagnosis, SPIE, Bellingham, WA, USA, 7 February 2018; Volume 10575, pp. 143–151. [CrossRef]
22. Logithasan, V.; Wong, J.; Reformat, M.; Lou, E. Using machine learning to automatically measure axial vertebral rotation on radiographs in adolescents with idiopathic scoliosis. *Med. Eng. Phys.* **2022**, *107*, 103848. [CrossRef] [PubMed]
23. Ebrahimi, S.; Gajny, L.; Vergari, C.; Angelini, E.D.; Skalli, W. Vertebral rotation estimation from frontal X-rays using a quasi-automated pedicle detection method. *Eur. Spine J.* **2019**, *28*, 3026–3034. [CrossRef]
24. Zhang, J.; Lou, E.; Hill, D.L.; Raso, J.V.; Wang, Y.; Le, L.H.; Shi, X. Computer-aided assessment of scoliosis on posteroanterior radiographs. *Med. Biol. Eng. Comput.* **2010**, *48*, 185–195. [CrossRef]
25. Morrison, D.G.; Chan, A.; Hill, D.; Parent, E.C.; Lou, E.H. Correlation between Cobb angle, spinous process angle (SPA) and apical vertebrae rotation (AVR) on posteroanterior radiographs in adolescent idiopathic scoliosis (AIS). *Eur. Spine J.* **2015**, *24*, 306–312. [CrossRef]
26. Vo, Q.N.; Lou, E.H.; Le, L.H. Measurement of axial vertebral rotation using three-dimensional ultrasound images. *Scoliosis* **2015**, *10*, S7. [CrossRef]
27. Forsberg, D.; Lundström, C.; Andersson, M.; Vavruch, L.; Tropp, H.; Knutsson, H. Fully automatic measurements of axial vertebral rotation for assessment of spinal deformity in idiopathic scoliosis. *Phys. Med. Biol.* **2013**, *58*, 1775. [CrossRef] [PubMed]
28. Newton, P.O.; Fujimori, T.; Doan, J.; Reighard, F.G.; Bastrom, T.P.; Misaghi, A. Defining the "three-dimensional sagittal plane" in thoracic adolescent idiopathic scoliosis. *J. Bone Jt. Surg.* **2015**, *97*, 1694–1701. [CrossRef] [PubMed]
29. Ilharreborde, B.; Steffen, J.S.; Nectoux, E.; Vital, J.M.; Mazda, K.; Skalli, W.; Obeid, I. Angle measurement reproducibility using EOS three-dimensional reconstructions in adolescent idiopathic scoliosis treated by posterior instrumentation. *Spine* **2011**, *36*, E1306. [CrossRef] [PubMed]
30. Newton, P.O.; Farnsworth, C.L.; Parvaresh, K.C. Imaging and three-dimensional analysis of adolescent idiopathic scoliosis. In *Seminars in Spine Surgery*; Elsevier: Amsterdam, The Netherlands, 2015; Volume 27, pp. 21–26.
31. József, K.; Schlégl, Á.T.; Burkus, M.; Márkus, I.; OŚullivan, I.; Than, P.; Csapó, M.T. Maximal axial vertebral rotation in adolescent idiopathic scoliosis: is the apical vertebra the most rotated? *Glob. Spine J.* **2022**, *12*, 244–248. [CrossRef]
32. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015; Proceedings, Part III 18; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241. [CrossRef]

entropy

Article

A Good View for Graph Contrastive Learning

Xueyuan Chen [1] and Shangzhe Li [2,*]

[1] State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China; xueyuanchen@buaa.edu.cn
[2] School of Statistics and Mathematics, Central University of Finance and Economics, Beijing 100081, China
[*] Correspondence: shangzheli@cufe.edu.cn

Abstract: Due to the success observed in deep neural networks with contrastive learning, there has been a notable surge in research interest in graph contrastive learning, primarily attributed to its superior performance in graphs with limited labeled data. Within contrastive learning, the selection of a "view" dictates the information captured by the representation, thereby influencing the model's performance. However, assessing the quality of information in these views poses challenges, and determining what constitutes a good view remains unclear. This paper addresses this issue by establishing the definition of a good view through the application of graph information bottleneck and structural entropy theories. Based on theoretical insights, we introduce CtrlGCL, a novel method for achieving a beneficial view in graph contrastive learning through coding tree representation learning. Extensive experiments were conducted to ascertain the effectiveness of the proposed view in unsupervised and semi-supervised learning. In particular, our approach, via CtrlGCL-H, yields an average accuracy enhancement of 1.06% under unsupervised learning when compared to GCL. This improvement underscores the efficacy of our proposed method.

Keywords: graph contrastive learning; coding tree representation; structural entropy

Citation: Chen, X.; Li, S. A Good View for Graph Contrastive Learning. *Entropy* **2024**, *26*, 208. https://doi.org/10.3390/e26030208

Academic Editor: Yanchun Liang

Received: 15 January 2024
Revised: 25 February 2024
Accepted: 26 February 2024
Published: 27 February 2024

1. Introduction

Contrastive learning has demonstrated its effectiveness in various domains, including computer vision [1–3], natural language processing [4,5], and graph representation learning [6]. Specifically, in the context of graph representation learning, graph contrastive learning (GCL) [7] has proven to be a valuable approach. GCL boosts the performance of downstream tasks by pre-training a Graph Neural Network (GNN) on extensive datasets, often characterized by limited or absent annotations. This method has evolved into a practical self-supervised learning technique for effectively capturing graph representations.

In the realm of graph contrastive learning, two key modules have been delineated: graph augmentation and contrastive learning methodologies [8]. Similar to contrastive learning methods in other domains, those designed for graphs aim to enhance agreement among positive examples while minimizing it among negative samples. Graph augmentation employs diverse strategies such as node dropping, edge perturbation, attribute masking, and subgraph operations to generate augmented views [7]. Researchers have highlighted the pivotal role of view quality in the performance of contrastive learning models [9] and have focused on constructing effective views for graphs through data augmentation [10,11]. Unlike images, generating high-quality contrastive samples for graphs is challenging due to the intricate structural information embedded in graphical data [12]. This raises the following fundamental questions regarding how to address these challenges:

(1) *What defines a good view?*
(2) *What information should a good view include or exclude?*
(3) *How can a good view be generated?*

Recently, the information bottleneck (IB) theory has been applied to learn graph representation [13]. This has inspired the proposition that a good view for graph representation should possess minimal yet sufficient information, i.e., the essential information. Consequently, a metric for quantifying the quality of information embedded in graphs is indispensable. Taking cues from information theory, particularly the quantification of information in communication systems [14], researchers have grappled with the formidable task of measuring graph structural information, considered one of the "three great challenges for half-century-old computer science" [15]. Recently, the concept of structural entropy for graphs has been introduced to measure the uncertainty of graph structures, thus addressing this challenge [16]. This theory posits that minimizing uncertainty in a graph, or reducing its structural entropy, unveils the essential structure of the graph. In essence, a good view aims to minimize structural uncertainty, providing minimal yet sufficient information, and maximize benefits in graph contrastive learning with the least cost.

In this research, we present a novel approach named CtrlGCL (refer to Figure 1) designed for graph contrastive learning, with a primary focus on the concept of a "good view" as defined earlier. Our methodology employs an optimization algorithm to decode essential structures by minimizing structural entropy. This decoding process generates coding trees, which represent essential structures corresponding to the given graphs. Subsequently, drawing inspiration from the message-passing mechanism inherent in Graph Neural Networks (GNNs), we propose an encoder tailored for learning representations from these coding trees, effectively capturing essential information. In comparison to existing effective views in previous studies, we conducted comprehensive experiments covering both semi-supervised and unsupervised learning across various graph classification benchmarks. The results demonstrate superior performance compared to state-of-the-art (SOTA) methods. The contributions are as follows:

- We are the first, to the best of our knowledge, to formulate a definition for a "good view" in the context of graph contrastive learning, grounded in the theories of graph information bottleneck and structural entropy.
- Drawing on these theoretical insights, we introduce CtrlGCL as a method to actualize the concept of a good view for graph contrastive learning, employing coding tree representation learning.
- Our proposed methodology for constructing good views was comprehensively assessed across various benchmarks, encompassing unsupervised and semi-supervised learning scenarios. The results consistently showcase its superior performance compared to state-of-the-art methods, underscoring the efficacy of our approach.

  In this article, the initial section provides a comprehensive overview of the background and outlines our specific contributions. The subsequent section delves into the existing research on graph contrast learning and structural entropy. Following that, the third section elucidates our theory and delineates the instantiation of our model. Moving forward, the fourth and fifth sections expound upon the experimental setup and present the obtained results. Finally, the concluding section encapsulates the essence of our study, providing a succinct summary of our findings.

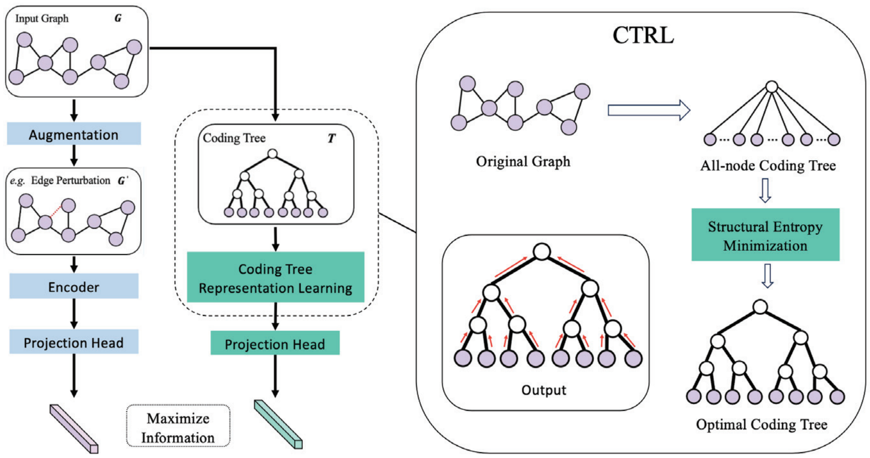


Figure 1. Framework. Our framework with the decoded essential graph structure for contrastive learning consists of two blocks. We adopt a view from previous works regarding graph contrastive learning. As for our good view, the original graphs are taken for coding tree transformation, and the representation for contrastive loss estimation can be obtained through the coding tree encoder.

2. Related Work

2.1. Graph Contrastive Learning

In the wake of the success achieved by contrastive learning in Convolutional Neural Networks (CNNs) for unsupervised image representation learning [2,9], the application of contrastive learning for graph representation learning has gained traction due to the scarcity of labels in real-world network data [7,10–12,17]. However, unlike image data augmentation, which does not demand extensive domain knowledge, augmentation in graph data is more intricate and challenging to analyze, posing difficulties in generating high-quality contrast samples [12,17,18]. Thus, the investigation of the contrastive view becomes a pivotal aspect of graph contrastive learning. Initially, contrastive pairs were constructed from different graph components, forming diverse contrastive modes [6,19–21]. More recently, inspired by the heuristics view design in computer vision, GCL introduced four types of views with random augmentation [7]. However, the optimal view combinations needed extensive evaluation. Subsequent works, like JOAO, proposed a search strategy based on the Min–Max principle for efficient view selection [11]. Similarly, AD-GCL aimed to produce graph views through learnable edge dropping [10]. Furthermore, LP-Info introduced a view-producing model via graph generation to avoid prefabricated data augmentations that required domain knowledge [17]. In addition to these foundational studies on GCL, an increasing number of researchers have employed the GCL methodology for recommendation systems. Specifically, CGI [22] adheres to the design principles of AD-GCL, learning to determine whether to drop an edge or node under the guidance of the information bottleneck principle. On the other hand, LightGCL [23] opts for graph reconstruction, leveraging singular-value decomposition for contrastive augmentation.

Despite the effectiveness of these graph views on various tasks, the data augmentations proposed, such as random perturbation, in existing methods may introduce structural damage and noisy information [7,10]. Similarly, learnable views through graph generation (i.e., LP-Info and LightGCL), based on different experimental settings, may not prevent artificially introduced noise. Additionally, methods like AD-GCL and CGI, relying on forced edge dropping, might suffer from graph structure damage and yield an undesirable performance in various regularizing settings [10]. In contrast, CtrlGCL provides theoretical guiding principles for contrastive view generation via an optimization algorithm that avoids random corruption and artificially introduced noisy information.

2.2. Structural Entropy

The need to measure information in communication networks gave rise to information entropy [14]. Several metrics have been developed for quantifying information in graphs. $p(G)$ may be used to evaluate the entropy of graphs worldwide [24]. Different methods attempt to quantify the structural entropy of nodes in a signal network. Based on distance, the first example of local graph entropy measurement was presented [25]. Subsequently, numerous research projects were undertaken in an effort to quantify a graph's structural information from various angles. These projects included Gibbs entropy [26], parametric graph entropy [27], and von Neumann entropy [28]. However, these definitions all destructure the graph into an unstructured probability distribution and then apply Shannon entropy to define the information of the graph. Therefore, these metrics do not suit the measurement of structural information, which is crucial for graphs and the key to the success of GNNs. In addition, these graph entropy definitions are only statistical mechanics approaches, providing an approach to comparing the different models of networks rather than an approach to figure out the minimal structural entropy of a given graph.

In more recent work, structural entropy was introduced and applied to evaluate the hierarchical structure complexity in a graph [16]. It was based on coding trees. Structural entropy was further established and used for decoding the fundamental graph structure, with an emphasis on measuring graph information through fixed hierarchical structures [29].

3. Materials and Methods

3.1. Preliminaries

Here, we introduce some preliminary concepts and notations. In this study, given a set of graphs $\mathbb{G} = G_1, G_2, \cdots, G_M$, every graph can be written as $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of nodes and edges, respectively. The graph G may have node attributes $\mathcal{X}_{\mathcal{V}} = X_v | v \in \mathcal{V}$.

3.1.1. Graph Representation Learning

Graph Neural Networks (GNNs) with a message-passing method were used as encoders in this work. The purpose of GNNs is to learn a vector $h_G \in \mathbb{R}$ for the whole graph G and an embedding vector $h_v \in \mathbb{R}$ for each node. The initial node representation h_v is updated iteratively using the GNN, starting at $h_v^{(0)} = X_v$. When an L-layer GNN is used, each node representation's update takes into account data from the nodes that are nearby within L hops. According to Gilmer et al. [30], the L-layer of a GNN may be written as follows:

$$h_v^{(l)} = f_U^l(h_v^{(l-1)}, f_M^{(l)}((h_v^{(l-1)}, h_u^{(l-1)})|u \in N(v))), \tag{1}$$

where h is the node representation of v in the L-th layer, $N(v)$ is the neighborhood node set for node v, f_U^l is the update function in the L-th layer, and f_M^l is the trainable message-passing function in the L-th layer. The node representation h_v is similar to a subgraph in that it is a summary of the nearby nodes. As a result, the whole-graph representation is formalized as follows after L iterations:

$$h_G = f_R(h_v | v \in \mathcal{V}), \tag{2}$$

where the readout function that pools the final collection of node representations is denoted by f_R.

3.1.2. The Mutual Information Maximization

Graph contrastive learning operates under the mutual information maximization (InfoMax) principle [20], where the objective is to maximize the degree of correspondence between a graph's representations and various augmented perspectives. The goal of the graph representation h_G is to capture the unique characteristics of the graph G such that the representation can effectively distinguish this graph from others. The following is how mutual information maximization aims to be expressed:

$$\text{InfoMax:} \quad \max I(G; h_G), \text{ where } G \sim \mathbb{P}_G, \tag{3}$$

where $I(\cdot)$ signifies the mutual information between two random variables, and $\mathbb{P}_G$ represents the distribution defined over the graph G.

3.1.3. Methodology

In this section, we begin by outlining our theoretical reasoning and then attempt to provide a definition of a good perspective. We next offer a particular instantiation of the good view tailored for graph contrastive learning, building upon structural information theory.

3.2. The Essential Structure with Minimal Structural Uncertainty

Given the challenge of limited labeled data in real-world graph datasets, obtaining meaningful representations and establishing effective pre-training is contingent upon self-supervised models delving into the intrinsic information of graphs [31]. The choice of "views" in self-supervised learning becomes pivotal as it dictates the information encapsulated by the model's representation [9]. While there have been prior works addressing the design of effective views for graph contrastive learning from various perspectives [7,10], none of them offer a clear definition of a good view aligned with the fundamental objective stated above. Our investigation seeks to fill this gap by providing a precise definition of what constitutes a good view in the context of graph contrastive learning.

In the realm of computer vision, researchers have provided an empirical solution, suggesting that a good view involves compressing the mutual information between views while preserving information relevant to downstream tasks [9]. The notion of the information bottleneck, or more precisely the graph information bottleneck (GIB), has been presented in relation to Graph Neural Networks (GNNs) [13]. A similar approach is put out by GIB, which drives our investigation into what makes a good perspective for graph contrastive learning. Models can obtain minimum yet adequate information for a particular task by simultaneously minimizing mutual information between the input and the output (i.e., $\min I(G; f(G))$) and maximizing it between the model's output and the target (i.e., $\max I(f(G); Y)$). The following is how the GIB aim is stated:

$$\text{GIB:} \quad \max_f I(f(G); Y) - \beta I(G; f(G)), \tag{4}$$

where β is a positive constant and $(G, Y) \sim \mathbb{P}_{\mathcal{G} \times \mathcal{Y}}$. We suggest that, under the framework of GIB, an optimal perspective for graph contrastive learning should capture the least amount of information necessary for tasks that come after, hence optimizing gains at the lowest possible expense.

It is undoubtedly important to note that the first part of GIB needs target-specific data for the job at hand (that is, Y), which presents difficulties for the self-supervised training framework. This challenge, however, directs our attention to the latter portion of GIB, which is independent of such target-related data. This feature clarifies the investigation of what, in the context of graph contrastive learning, makes a good view. Moreover, the goal of decreasing the mutual information between the input graph and the learnt representation (that is, $\min I(G; f(G))$) emphasizes the essential information that graph contrastive learning ought to include. This goal supports the notion that an excellent perspective should emphasize gathering pertinent details while reducing redundancy, providing insightful information for efficient graph representation learning.

To delve deeper into the essential information of graphs, we establish a property that a good view should possess:

Definition 1. *A good view is intended to be a substructure of the corresponding graph to mitigate the introduction of artificially induced noise.*

In computer vision, important information is frequently obtained via random disturbance of the data, and the generated noise is known to support robust representation learning [9]. Graph augmentation is more difficult to interpret and less intuitive than data augmentation on photos, which does not require in-depth topic expertise. It is challenging to produce high-quality contrastive samples for graphs because of this complexity [12,17]. Thus, we argue that random perturbation should not be used to introduce fake noise in a decent representation of graphs.

Given a normal graph G and its G^* view, the mutual information between G and G^* may be written as follows:

$$I(G^*; G) = \mathcal{H}(G^*) - \mathcal{H}(G^*|G), \tag{5}$$

where the entropy of G^* is $\mathcal{H}(G^*)$, and the conditional entropy of G^* conditioned on G is $\mathcal{H}(G^*|G)$ (for simplicity, we ignore the graph encoder f without losing generality). According to the definition of Shannon entropy [14], the uncertainties of G^* and G are also represented by the variables $\mathcal{H}(G^*)$ and $\mathcal{H}(G)$, respectively.

Furthermore, we may deduce that $\mathcal{H}(G^*|G) = 0$ since, in accordance with Definition 1, the information encoded in G^* is a subset of the information in G. Thus, it is possible to reduce the mutual information to the following:

$$I(G^*; G) = \mathcal{H}(G^*). \tag{6}$$

Consequently, to capture the essential information of the input graph, we need to minimize the uncertainty of graph G, expressed as $\min, \mathcal{H}(G^*)$. Here, we provide the definition of a good view for graph contrastive learning (GCL).

Definition 2. *The good view of a graph should have minimal structural uncertainty.*

Shannon entropy is a useful measure for assessing structural information in graphs, but it is not appropriate for our purposes. Brooks posed the dilemma of how to characterize a graph's underlying data in a way that makes it possible to understand its fundamental structure [15]. Shannon likewise considered whether communication graph analysis could be aided by a structural theory of information [32].

To measure the uncertainty of a graph's structure, the notion of structural entropy was recently proposed and described on graphs [16]. This structural information theory states that a coding tree encodes a graph. The structural entropy of a graph $G = (\mathcal{V}, \mathcal{E})$ on its coding tree T is defined as

$$\mathcal{H}^T(G) = -\sum_{v_\tau \in T} \frac{g_{v_\tau}}{vol(\mathcal{V})} \log \frac{vol(v_\tau)}{vol(v_\tau^+)}, \tag{7}$$

where g_{v_t} denotes the number of edges with an endpoint in the leaf node partition of v_t, v_t^+ is the parent of v_t, and $vol(\mathcal{V})$ and $vol(v_t)$ are the sums of degrees of leaf nodes in $\mathcal{V}$ and v_t, respectively. Moreover, v_t is a nonroot node in T and can also be viewed as a node subset $\subset \mathcal{V}$ according to its leaf node partition in T.

The objective is to find the ideal coding tree T with the smallest entropy, or $\min_T \mathcal{H}^T(G)$, in order to interpret the fundamental structure of graph G with the least amount of structural uncertainty. A coding tree with a matching constant height is preferable because real-world networks frequently have a natural structure with a defined hierarchy. Here, the ideal coding tree with a height of k is decoded using the k-dimensional structural entropy:

$$\mathcal{H}^k(G) = \min_{\forall T:\text{Height}(T)=k} \mathcal{H}^T(G). \tag{8}$$

An Instantiation of Essential Structure Decoding and Representation

In this subsection, we will initially present a practical instantiation for decoding the essential structure to minimize structural uncertainty. Following this, we will introduce a novel Graph Convolutional Network (GCN) for the coding tree representation based on Graph Neural Networks (GNNs).

We seek a method to decode the coding tree of height k from a given graph, guided by the notion of k-dimensional structural entropy. A coding tree T can be created for a graph $G = (\mathcal{V}, \mathcal{E})$, with v_r serving as the tree's root node and $\mathcal{V}$ serving as its leaf nodes. Two functions for the coding tree T are defined as follows:

Definition 3. *Let T be any coding tree for the graph $G = (\mathcal{V}, \mathcal{E})$, where the leaf nodes are $\mathcal{V}$ and the root node is v_r. In T, let $v_i \in v_r.children$ and $v_j \in v_r.children$ be any two nodes (v_i, v_j). Defining a function $MERGE_T(v_i, v_j)$ for T that inserts a new node v_ε between (v_i, v_j) and v_r*

$$v_\varepsilon.children \leftarrow v_i; \tag{9}$$

$$v_\varepsilon.children \leftarrow v_j; \tag{10}$$

$$v_r.children \leftarrow v_\varepsilon; \tag{11}$$

Definition 4. *In accordance with the configuration described in Definition 3, given any two nodes (v_i, v_j), where $v_i \in v_j.children$. Create the function $delete_T(v_i)$ for T in order to merge $v_i.children$ with $v_j.children$ and delete v_i from T:*

$$v_j.children \leftarrow v_i.children; \tag{12}$$

Algorithm 1 provides a greedy algorithm based on the two provided functions that computes the coding tree with a given height k using structural entropy minimization. More specifically, starting from the bottom, a full-height binary coding tree is created. In this step, the goal is to maximize the decrease in structural entropy by merging two child nodes of the root to produce a new division in each iteration. In the second step, we must eliminate nodes from the previous full-height binary coding tree in order to compress it to meet a set number of graph coarsenings. Each time, we take an inner-node from T, and after removing it, T has the lowest structural entropy. With the help of structural entropy, we have already created a coding tree with a certain height k at the conclusion of the second step. When implementing hierarchical pooling based on such a coding tree, there can be nodes that, due to cross-layer linkages, do not have an immediate successor in the following layer. This will result in nodes being missing. As a result, in order to maintain the integrity of information transfer across layers and to avoid interfering with G's structural entropy on the coding tree T, we must complete the third step. Ultimately, $T = (\mathcal{V}^T, \mathcal{E}^T)$, $\mathcal{V}^T = (\mathcal{V}_0^T, \ldots, \mathcal{V}_k^T)$, and $\mathcal{V}_0^T = \mathcal{V}$ may be used to create a coding tree T for the provided graph G. Furthermore, it is possible to obtain the cluster assignment matrices from $\mathcal{E}^T$, that is, $\mathbb{S} = (\mathbf{S}_1, \ldots, \mathbf{S}_k)$.

Complexity analysis of Algorithm 1. With h_{max} representing the height of the coding tree T following the first step, the runtime complexity of Algorithm 1 is $O(2n + h_{max}(m \log n + n))$. During the structural entropy reduction process, h_{max} will be about $\log n$, since coding tree T tends to be balanced. Algorithm 1's runtime roughly grows linearly with the number of edges, as a network often has more edges than nodes, namely $m \gg n$. The algorithm under consideration maintains two data structures: a coding tree and a graph. The space complexity of the algorithm is $O(m + n)$, where n denotes the number of nodes and m represents the number of edges. Specifically, the graph requires $O(n + m)$ space. The coding tree, on the other hand, necessitates $O(n)$ space, given that the number of nodes in the coding tree is less than or equal to $2n$.

Algorithm 1 Coding tree with height k via structural entropy minimization.

```
Input: a graph $G = (\mathcal{V}, \mathcal{E})$, a positive integer $k > 1$
Output: a coding tree $T$ with height $k$
 1: Generate a coding tree $T$ with a root node $v_r$ and all nodes in $\mathcal{V}$ as leaf nodes;
 2: // Stage 1: Bottom to top construction;
 3: while $|v_r.children| > 2$ do
 4:     Select $v_i$ and $v_j$ from $v_r.children$, conditioned on
        $argmax_{(v_i,v_j)}\{\mathcal{H}^T(G) - \mathcal{H}^{T_{\mathrm{MERGE}(v_i,v_j)}}(G)\}$;
 5:     MERGE$(v_i, v_j)$;
 6: end while
 7: // Stage 2: Compress $T$ to the certain height $k$;
 8: while Height$(T) > k$ do
 9:     Select $v_i$ from $T$, conditioned on
        $argmin_{v_i}\{\mathcal{H}^{T_{\mathrm{REMOVE}(v_i)}}(G) - \mathcal{H}^T(G)|$
                 $v_i \neq v_r \,\&\, v_i \notin \mathcal{V}\}$;
10:     REMOVE$(v_i)$;
11: end while
12: // Stage 3: Fill $T$ to avoid cross-layer links;
13: for $v_i \in T$ do
14:     if $|\mathrm{Height}(v_i.parent) - \mathrm{Height}(v_i)| > 1$ then
15:         insert a new node $v_\varepsilon$ between $v_i$ and $v_j$;
16:     end if
17: end for
18: return $T$;
```

Coding tree representation learning. The coding tree functions act as a compact representation of the original graph structure, preserving its essential information while minimizing redundancy and avoiding noise introduced during augmentation. To seamlessly integrate the coding tree into the graph contrastive learning architecture, a novel encoder is introduced. In order to capture the hierarchical structure of nodes within the coding tree, a tree positional encoding mechanism is employed. This mechanism enables the model to distinguish nodes located at different depths. The positional embedding p^i for nodes of height i is defined as follows:

$$p^i = \mathrm{PositionEncoder}(i), \tag{13}$$

where PositionEncoder(i) generates unique embeddings for each layer of the coding tree. In practical experiments, the implementation of PositionEncoder utilizes the embedding layer provided by PyTorch.

The encoder, a novel recursive neural network, propagates information iteratively from the bottom to the top. The process begins with the leaf nodes, and as iterations progress, the model gradually learns representations for each non-leaf node by aggregating the representations of its descendants. This iterative process culminates in the derivation of the representation for the root node. We utilize a Gated Recurrent Unit (GRU) [33] as the aggregate function. Consequently, the representation of a non-leaf node with height i in the coding tree is computed as

$$\hat{r}_v^i = \textstyle\sum_{u \in \mathcal{C}(v)} r_u^{(i-1)}, \tag{14}$$

$$r_v^i = \mathrm{GRU}\Big(p^i, \;\; \hat{r}_v^i\Big), \tag{15}$$

where r_v^i is the hidden representation of node v, $p^{(i)}$ is taken as the input, and $\hat{r}_v^i$ is the aggregated information from the children of node v, which represents the hidden state. More specifically, r_v^i is given by

$$s_v^i = \sigma\Big(\mathbf{W}_s p^i + \mathbf{U}_s \hat{r}_v^i\Big), \tag{16}$$

$$z_v^i = \sigma\Big(\mathbf{W}_z p^i + \mathbf{U}_z \hat{r}_v^i\Big), \tag{17}$$

$$\tilde{r}_v^i = \tanh\Big(\mathbf{W}_r p^i + \mathbf{U}_r (s_v^i \odot \hat{r}_v^i)\Big), \tag{18}$$

$$r_v^i = (1 - z_v^i) \odot \tilde{r}_v^i + z_v^i \odot \hat{r}_v^i, \tag{19}$$

where σ is the logistic sigmoid function, $\odot$ denotes element-wise multiplication, $\mathbf{W}_*$ and $\mathbf{U}_*$ refer to weight matrices used for linear transformations of vectors that control how the input and hidden state are combined to produce the new state r_v^i.

Complexity analysis of coding tree learning. The runtime complexity is $O(nd^2)$. This process involves a propagation step, which traverses the tree from the leaf nodes to the root, taking $O(n)$ time. At each node, the GRU executes update and reset operations, both of which involve matrix multiplications. Assuming the dimension of the hidden feature to be d, these operations require $O(d^2)$ time. The space complexity of the learning process is $O(nd + d^2)$, determined by the storage of the node features and the parameters of the GRU. Each node in the tree possesses a d-dimensional feature vector, hence storing the features for all tree nodes requires $O(nd)$ space. The GRU parameters, which are of size $O(d^2)$, also contribute to the space complexity.

For the contrastive loss calculation, x_{root}^k (i.e., the feature vector of the root node) can be used to represent the entire coding tree. However, recognizing the distinct functionality of the natural hierarchy, we incorporate the embedded information from each iteration through skip connections. Specifically, we learn the coding tree with concatenated layer representations:

$$r_T = [\text{POOL}(\{r_v^0 | v \in V_T^0\}) \ ; \ \text{POOL}(\{r_v^1 | v \in V_T^1\}) \ ; \ \ldots \ ; \ x_{root}^k)], \tag{20}$$

where r_v^i is the hidden representation and k is the height of tree T. In particular, POOL in Equation (20) will be implemented by the widely used pooling approaches, such as summation or averaging.

4. Experiment Setup

In this section, we dedicate ourselves to evaluating CtrlGCL through extensive experiments. We begin by describing the experimental setup for graph classification, covering both semi-supervised and unsupervised learning scenarios. Subsequently, we validate the effectiveness of the proposed good view against state-of-the-art (SOTA) competitors, contrasting with pre-defined rules for graph augmentation.

Given that our good view is orthogonal to previous works on graph augmentations, we conduct additional analyses to demonstrate the collaborative capabilities of CtrlGCL with existing approaches. This comprehensive evaluation aims to showcase the versatility and effectiveness of CtrlGCL across various experimental settings and in collaboration with diverse graph augmentation strategies.

4.1. Datasets

Various benchmarks for view validation are adopted from TUDatasets [34]. Specifically, we utilized six datasets for social networks, including IMDB-BINARY, IMDB-MULTI, COLLAB, REDDIT-MULTI-5K, REDDIT-BINARY, and GITHUB; two datasets for small molecules, including NCI1 and MUTAG; and two datasets for bioinformatics, including PROTEINS and DD.

We performed trials for several graph property prediction tasks on a large variety of datasets from different disciplines. We offer thorough explanations of each of the ten benchmark datasets utilized in this investigation. The statistics for these datasets are summarized in Table 1.

Social Network. IMDB-BINARY and -MULTI are products of a movie set working together. Actors or actresses are represented as nodes in these two databases, while their collaboration in a certain film is represented by edges. Every graph has a label that relates to the genre of the particular film that it is connected to. Comparably, COLLAB is a scientific domain collaboration dataset made up of three public collaboration datasets: condensed matter physics, high-energy physics, and astronomy. For the graphs in this benchmark, researchers from different fields have created different ego networks. The study field that each graph's nodes belong to is indicated by its label. The balanced datasets REDDIT-BINARY or -MULTTI-5K have graphs that each represent an online discussion, and the nodes stand for users. If two nodes reply to one other's comments, then an edge is formed between them. Sorting each graph into the appropriate community or subreddit is the current work at hand.

Bioinformatics. Protein structure diagrams are included in DD. Every node symbolizes an amino acid, and edges arise when two nodes are separated by less than 6 A°. If a protein is an enzyme or not, it is indicated on the label. A dataset known as PROTEINS has secondary structural elements (SSEs) as its nodes. If two nodes are next to one another in a 3D space or in the provided amino acid sequence, then an edge exists between them. Three discrete labels that stand for helices, sheets, or turns are found in the dataset. The NCI1 dataset comes from the field of chemical informatics, where each input graph is used as a representation of a compound: each vertex represents an atom of a molecule, and the edges between vertices represent bonds between atoms. This dataset is related to anti-cancer screening, where chemicals are evaluated as positive or negative for cellular lung cancer. A total of 37 distinct labels make up this dataset. Seven different graph types are found in MUTAG, which are formed from 188 different carcinogenic aromatic and heteroaromatic nitro chemicals. Ten datasets were used, and their characteristics are compiled in Table 1.

Table 1. Statistics for the datasets from TUDataset.

Dataset	#Graphs	#Classes	Avg. #Nodes	Avg. #Edges
REDDIT-BINARY	2000	2	429.63	497.75
COLLAB	5000	3	74.49	2457.78
REDDIT-MULTI-5K	4999	5	508.52	594.87
IMDB-MULTI	1500	3	13.00	65.94
IMDB-BINARY	1000	2	19.77	96.53
GITHUB	12,725	2	113.79	234.64
MUTAG	188	2	17.93	19.79
NCI1	4110	2	29.87	32.30
DD	1178	2	284.32	715.66
PROTEINS	1113	2	39.06	72.82

4.2. Configuration

Our two-block contrastive learning framework with the decoded basic graph structure is illustrated in Figure 1. In the block pertaining to graph augmentations, we utilize the identical GNN architectures with their original hyper-parameters under different experiment circumstances, as per the methodology used in GraphCL (the first technique for graph contrastive learning) [7]. To be more precise, we utilized GIN with 32 hidden units and 3 layers for unsupervised representation learning and ResGCN with 128 hidden units and 5 layers for semi-supervised learning. Furthermore, graphs with a default augmentation strength of 0.2 were subjected to the same data augmentations.

The number of tree encoder layers for coding tree representation learning was determined by the tree height, which ranges from two to five. There were two layers in each iteration of the MLP (Multilayer Perceptron). In order to preserve uniformity with GraphCL across the different experiment configurations, the encoder's hidden dimensions are specified with the corresponding setting. The optimal hyper-parameter combination was determined based on the performance on the validation sets.

Semi-Supervised Learning. We performed five trials for each dataset, with a 10% label rate, meaning that each experiment corresponds to a 10-fold assessment, as described in [7]. For every experiment, we present the accuracy (%) mean, and standard deviation. A grid search was used to set the epoch number to $\{20, 40, 60, 80, 100\}$ and the learning rate to $\{0.01, 0.001, 0.0001\}$ for pre-training. We fine-tuned using the same parameters as described in [7]: learning rate of 0.001, batch size of 128, hidden dimension of 128, and 100 epochs of training for the pre-trained models.

Unsupervised Learning. Each experiment was carried out five times, and as shown in [20], each experiment corresponds to a 10-fold assessment. For every experiment, we present the accuracy (%) mean, and standard deviation. Models were tested every 10 epochs and trained for 20 epochs in order to learn the graph representations. The batch size was 128 and the hidden dimension was 32.

Data Augmentations on Graphs. There are four common categories of data augmentations for graph-structured data, which correspond to the data augmentations utilized in GraphCL [7].

Edge Perturbation. Here, the connectivities in G are disturbed by arbitrarily adding or removing a specific percentage of edges. This operation is predicated on the notion that the semantic meaning of G is relatively resilient to variations in edge connection patterns. To add or remove each edge, we further adhered to an independent, identically distributed (i.i.d.) uniform distribution.

Node Dropping. Node dropping, given the graph G, randomly discards a subset of vertices and their connections. This procedure makes the assumption that the semantic meaning of G is unaffected by the missing portion of vertices. The default independent and identically distributed uniform distribution governs each node's dropping probability.

Attribute Masking. With attribute masking, models are prompted to retrieve masked vertex attributes by utilizing their context, that is, the remaining attributes. The underlying premise of this operation is that the model's predictions are not substantially impacted by the missing partial vertex information.

Subgraphs. Using a random walk, this procedure samples a subgraph from G. It makes the assumption that G's partial local structure can effectively retain its semantics.

4.3. Learning Protocols

We employed the necessary learning techniques to enable fair comparison with state-of-the-art (SOTA) efforts. All data were utilized for model pre-training in unsupervised representation learning [20], after which the learnt graph embeddings were fed into an SVM classifier for 10-fold cross-validation. Two learning settings were used for semi-supervised learning [7]. Only the training dataset was used for pre-training when the datasets had a public training/validation/test split. Ten percent of the training data was used for fine-tuning, and the validation/test sets yielded the final assessment findings. All samples were used for pre-training on datasets without these splits, and assessment and fine-tuning were carried out across ten assessments.

4.4. The Compared Methods

In the two-block design of CtrlGCL, there were three types of results: (1) only graph embedding, termed CtrlGCL-G; (2) only coding tree embedding, termed CtrlGCL-T; (3) and the hybrid of graph embedding and coding tree embedding, termed CtrlGCL-H.

In unsupervised learning, we adopted eight baselines that fall into three categories. We used the published hyper-parameters of these methods. The first set included three state-of-the-art (SOTA) kernel-based methods: GL [35], WL [36], and DGK [37]. The second set comprised four heuristic self-supervised methods: node2vec [38], sub2vec [39], graph2vec [40], and InfoGraph [20]. GraphCL, the last technique in this group, uses the same pre-established augmentation rules on graphs for unsupervised learning [7]. The default augmentation ratio was 0.2 (dropping, perturbation, masking, and subgraph). In addition to the individual use of particular data augmentation, GraphCL uses augmentation pools for contrastive learning. Specifically, biological molecules are treated using node dropping and subgraphs; all augmentations are applied to dense social networks; and for sparse social networks, all except attribute masking are employed.

Under semi-supervised learning, we considered five baselines:

(1) A naive GCN without pre-training [7], which is directly trained with 10% labeled data from random initialization.
(2) GAE [41], a predictive method based on edge-based reconstruction in the pre-training phase.
(3) Infomax [6], a node-embedding method with global–local representation consistency.
(4) ContextPred [19], a method using sub-structure information preserving.
(5) GraphCL [7], the first graph contrastive learning method with data augmentations.

5. Results

5.1. Unsupervised Learning

In the context of unsupervised learning, Table 2 summarizes the classification accuracy of CtrlGCL and the comparative approaches. When considering the baselines, the results show a notable boost in performance with the addition of a good view. When the last column for average rank is taken into account, the three CtrlGCL variations have the highest ranks. Significantly, our techniques outperform the competing approaches on all seven benchmarks, with the exception of NCI1. In addition, in the case without kernel-based techniques, CtrlGCL-G consistently achieves the maximum accuracy. Our findings indicate that, in unsupervised learning scenarios, our techniques regularly beat the most advanced approaches.

In addition to the overall superior performance of CtrlGCL, we delved deeper into the specific performance of each variant. As indicated by the marker * in Table 2, we identified the best performances among the three variants. Among the four datasets in social networks, CtrlGCL-T achieves the highest accuracies on COLLAB and IMDB-BINARY, while CtrlGCL-H outperforms on the REDDIT datasets. Notably, the edge density (average edges divided by average nodes) of COLLAB and IMDB-BINARY is much higher than that of the REDDIT datasets, suggesting higher structural uncertainty in COLLAB and IMDB-BINARY. Therefore, the performance of CtrlGCL-T/H on social networks confirms the effectiveness of the proposed good view in minimizing structural uncertainty. Additionally, for datasets with higher structural uncertainty, the proposed good view provides high-quality graph representation, while for datasets with lower structural uncertainty, the proposed method presents sufficient information for performance improvement. In bioinformatics datasets, a similar phenomenon can be observed on the PROTEINS dataset, which also has higher edge density. However, different results are shown in DD, where the non-structural properties of this type of protein may explain the variation. For the other two datasets with lower edge density, CtrlGCL-G, which employs graph embedding, shows the best performance, implying lower structural uncertainty in these two sets.

Table 2. The average classification accuracies (%), along with their standard deviations (±Std.), obtained from five separate runs of the compared methods via unsupervised representation learning. The **bold** text highlights the best overall performances among all methods. The marker * indicates the best performance among the three variations of CtrlGCL. The term A.R. stands for average rank, which is used to assess the relative performance of each method. The results for the baselines were obtained from previously published works.

	NCI1	**PROTEINS**	**DD**	**MUTAG**	**COLLAB**	**RED-B**	**RED-M5K**	**IMDB-B**	**A.R.**
Avg. #Nodes	29.87	39.06	284.32	17.93	74.49	429.63	508.52	19.77	
Avg. #Edges	32.30	72.82	715.66	19.79	2457.78	497.75	594.87	86.53	
GL	-	-	-	81.66 ± 2.11	-	77.34 ± 0.18	41.01 ± 0.17	65.87 ± 0.98	8.3
WL	80.01 ± 0.50	72.92 ± 0.56	-	80.72 ± 3.00	-	68.82 ± 0.41	46.06 ± 0.21	72.30 ± 3.44	6.7
DGK	**80.31 ± 0.46**	73.30 ± 0.82	-	87.44 ± 2.72	-	78.04 ± 0.39	41.27 ± 0.18	66.96 ± 0.56	5.7
node2vec	54.89 ± 1.61	57.49 ± 3.57	-	72.63 ± 10.20	-	-	-	-	9.3
sub2vec	52.84 ± 1.47	53.03 ± 5.55	-	61.05 ± 15.80	-	71.48 ± 0.41	36.69 ± 0.42	55.26 ± 1.54	10
graph2vec	73.22 ± 1.81	73.30 ± 2.05	-	83.15 ± 9.25	-	75.78 ± 1.03	47.86 ± 0.26	71.10 ± 0.54	7.0
InfoGraph	76.20 ± 1.06	74.44 ± 0.31	72.85 ± 1.78	89.01 ± 1.13	70.65 ± 1.13	82.50 ± 1.42	53.46 ± 1.03	73.03 ± 0.87	4.3
GraphCL	77.87 ± 0.41	74.39 ± 0.45	78.62 ± 0.40	86.80 ± 1.34	71.36 ± 1.15	89.53 ± 0.84	55.99 ± 0.28	71.14 ± 0.44	4.0
CtrlGCL-G	79.00 ± 0.72*	75.79 ± 0.27	78.15 ± 0.56	**90.21 ± 0.66 ***	70.73 ± 0.65	89.85 ± 0.56	55.27 ± 0.32	72.30 ± 0.24	2.9
CtrlGCL-T	74.92 ± 0.53	**76.01 ± 0.42 ***	77.34 ± 1.03	88.50 ± 1.30	**74.12 ± 0.47 ***	88.67 ± 0.60	52.26 ± 0.69	**73.58 ± 0.44 ***	3.4
CtrlGCL-H	78.86 ± 0.38	75.85 ± 0.46	**78.76 ± 0.57 ***	90.17 ± 0.97	71.44 ± 0.45	**90.21 ± 0.65 ***	**56.13 ± 0.30 ***	72.78 ± 0.64	2.0

5.2. Semi-Supervised Learning

Under semi-supervised learning, the accuracies of our models and the compared methods are presented in Table 3. Notably, GtrlGCL is better than these state-of-the-art (SOTA) methods across all benchmarks. With the exception of CtrlGCL-T's poor performance, CtrlGCL-G and CtrlGCL-H rank as the top two methods overall. Specifically, CtrlGCL-G achieves the highest accuracy on three out of the seven benchmarks, while CtrlGCL-H holds this position in four out of the seven benchmarks. These results demonstrate the effectiveness of our methods under semi-supervised learning. The results from Table 3 show that GtrlGCL-H, which incorporates both graph embedding and coding tree embedding, achieves the highest mean rank and significant performance improvement compared to CtrlGCL-G. This validates the effectiveness of our proposed good view in semi-supervised learning. However, the poor performance of GtrlGCL-T is a concern. Nonetheless, the overall results demonstrate the value of our methods in semi-supervised learning, with CtrlGCL-G and CtrlGCL-H ranking highly.

Table 3. The comparison approaches' average accuracies (%) and standard deviations (±Std) during semi-supervised learning with 10% labels. The strategy that performed the best overall is highlighted in the **bold** text. Average rank, or A.R., is used to evaluate the relative effectiveness of each approach. The baseline results are from previously released publications.

	NCI1	**PROTEINS**	**DD**	**COLLAB**	**RED-B**	**RED-M5K**	**GITHUB**	**A.R.**
No Pre-Train	73.72 ± 0.24	70.40 ± 1.51	73.56 ± 0.41	73.71 ± 0.27	86.63 ± 0.27	51.33 ± 0.44	60.87 ± 0.17	7.0
GAE	74.36 ± 0.24	70.51 ± 0.17	74.54 ± 0.68	75.09 ± 0.19	87.69 ± 0.40	53.58 ± 0.13	63.89 ± 0.52	5.0
Infomax	74.86 ± 0.26	72.27 ± 0.40	75.78 ± 0.34	73.76 ± 0.29	88.66 ± 0.95	53.61 ± 0.31	65.21 ± 0.88	4.0
ContextPred	73.00 ± 0.30	70.23 ± 0.63	74.66 ± 0.51	73.69 ± 0.37	84.76 ± 0.52	51.23 ± 0.84	62.35 ± 0.73	7.3
GraphCL	74.63 ± 0.25	74.17 ± 0.34	76.17 ± 1.37	74.23 ± 0.21	89.11 ± 0.19	52.55 ± 0.45	65.81 ± 0.79	3.3
CtrlGCL-G	74.72 ± 0.26	**74.65 ± 0.54**	**76.33 ± 0.43**	74.26 ± 0.27	**89.40 ± 0.23**	52.93 ± 0.37	65.92 ± 0.64	2.1
CtrlGCL-T	71.80 ± 0.35	73.31 ± 0.47	75.63 ± 0.58	73.36 ± 0.35	88.70 ± 0.15	52.11 ± 0.34	65.39 ± 0.58	5.6
CtrlGCL-H	**75.09 ± 0.22**	73.85 ± 0.53	75.82 ± 0.65	**75.18 ± 0.22**	89.35 ± 0.27	**53.73 ± 0.28**	**66.01 ± 0.66**	1.7

5.3. Orthogonal to Graph Augmentations

In this section, we evaluate the collaborative potential of CtrlGCL by integrating it with four established graph augmentation techniques: AD-GCL [10], JOAO [11], AutoGCL [42], and RGCL [8]. These approaches introduce innovative strategies for generating augmented views, and we explore their synergy with CtrlGCL in an unsupervised learning setting.

AD-GCL [10]. We used the same configurations as before in the cooperative experiment with AD-GCL, and we swapped out its anchor view with the proposed good view from CtrlGCL. The approaches were assessed using a linear classifier after they had been trained using the relevant self-supervised goal. We adhered to the linear assessment methodology presented in AD-GCL [10]. Specifically, once the encoder provides representations, a Logistic (+L2) classifier is trained on top and evaluated for classification tasks. The classifier was implemented using Scikit-learn [43] or LibLinear [44] solvers. Finally, the lone hyper-parameter of the downstream linear model, that is, the L2 regularization strength, is grid searched among {0.001, 0.01, 0.1, 1, 10, 100, 1000} on the validation set for every single representation evaluation. Accuracy (%) was selected as the test parameter in accordance with the usual procedure. Every AD-GCL experiment was ran ten times using a different set of random seeds. For every dataset, we provided the mean and standard deviation of the associated test measure.

The encoder utilized in the joint experiment with AD-GCL was the GIN encoder [45]. To guarantee a fair comparison, the encoder was fixed and not adjusted while performing self-supervised learning (i.e., embedding dimension, number of layers, pooling type) for all the approaches. This decision was made with the intention of completely attributing any performance disparity to the self-supervised goal and excluding the encoder design. The GIN encoder was configured with the following unique hyper-parameters: a batch size of 32, a hidden dimension of 32, five GIN layers, summation as the graph readout function, and a dropout set at 0.5. Adam was used for optimization, and the learning rates in AD-GCL were adjusted to be within $\{0.01, 0.005, 0.001\}$ for both the encoder and the augmenter. Since asymmetric learning rates for the augmenter and encoder tend to render the training non-stable, the learning rate was set to 0.001 for all datasets and experiments were carried out to ensure stability [10]. Using the validation set, the number of training epochs was selected as $\{20, 50, 80, 100, 150\}$.

JOAO(v2) [11]. We used the same experimental setup as the original study in our collaboration with JOAO, but we made the following significant change: we swapped out one of the two views with the "good view" suggested in CtrlGCL. We were able to assess the effectiveness of our suggested view selection technique as a result. In this experiment, GIN was also adopted as the basic graph encoder [45], while non-linear SVM was employed for evaluation as GraphCL. To strike a compromise between the contrastive loss and view distance, the hyper-parameter γ introduced in JOAO was adjusted within the range $\{0.01, 0.1, 1\}$. Notably, because multiple projection heads were used, JOAOv2 was pre-trained twice as many epochs than JOAO, despite our 20 epochs of pre-training JOAO. With this modification, we were able to evaluate the relative performance of the two approaches and make a direct comparison.

AutoGCL [42]. We adopted the naive training strategy proposed in AutoGCL to make a fair comparison. Specifically, we retained one of the two graph generators and assigned our proposed anchor view to the blank position. In particular, AutoGCL extends the layer number of the graph encoder from 3 to 5 and the hidden size from 32 to 128. Moreover, AutoGCL was pre-trained with 30 epochs rather than 20 epochs.

RGCL [8]. In cooperation with RGCL [8], we faithfully followed the experiment settings revealed in their codes while replacing one of the two rationale-augmented views with SEGA. Note that, the tuned hyper-parameters in RGCL include the learning rate, sampling ratio ρ, loss temperature τ, and loss balance λ. In particular, RGCL was pre-trained on 40 epochs in total and evaluated every 5 epochs.

The unsupervised learning classification accuracies (%) of CtrlGCL in collaboration with the four methods for augmentations are presented in Table 4. The last column displays the average accuracies (%) over all datasets, and the three versions of CtrlGCL always suppress their corresponding partner view, highlighting the efficacy of the proposed good view in minimizing structural uncertainty. Specifically, when combined with AD-GCL-

FIX, CtrlGCL-T achieved the highest accuracies on three out of the nine datasets, while CtrlGCL-H outperformed on the remaining six benchmarks. Despite a few setbacks in the collaboration with JOAO, the overall superior performance underscores the success of CtrlGCL in these extensive experiments. Furthermore, when paired with AutoGCL, CtrlGCL-T attained the highest accuracies on two out of the eight datasets, while CtrlGCL-H excelled on five out of the eight benchmarks. In the case of RGCL, the collaboration yielded the highest results on all eight datasets. To elaborate, CtrlGCL-G outperformed on the MUTAG dataset, CtrlGCL-T excelled on the RED-B and IMDB-B datasets, and CtrlGCL-H achieved the best results on the remaining five datasets.

Table 4. The average accuracy (%) $\pm$ standard deviation (over five times) of various methods used in unsupervised learning. Boldface type highlights the best performances for each individual dataset. A.A. signifies the average accuracy across all datasets. The results of AD-GCL-FIX, JOAO(v2), AutoGCL, and RGCL were obtained from their respective papers.

View1	View2	NCI1	PROTEINS	DD	MUTAG	COLLAB	RED-B	RED-M5K	IMDB-B	IMDB-M	A.A.
AD-GCL-FIX		69.57 ± 0.51	73.59 ± 0.65	74.49 ± 0.52	89.25 ± 1.45	73.71 ± 0.27	85.52 ± 0.79	53.00 ± 0.82	71.57 ± 1.01	49.04 ± 0.53	71.05
CtrlGCL-G	AD-GCL-Fix	69.93 ± 0.73	73.76 ± 0.57	74.62 ± 0.43	89.63 ± 1.54	73.77 ± 0.56	85.75 ± 0.67	53.88 ± 0.63	71.76 ± 0.57	49.44 ± 0.98	71.39
CtrlGCL-T	AD-GCL-Fix	69.78 ± 0.32	$\mathbf{74.61 \pm 0.81}$	75.55 ± 0.51	88.20 ± 1.20	73.97 ± 0.55	86.73 ± 0.55	53.52 ± 0.31	$\mathbf{72.32 \pm 0.49}$	$\mathbf{50.83 \pm 0.34}$	71.72
CtrlGCL-H	AD-GCL-Fix	$\mathbf{70.38 \pm 0.76}$	74.57 ± 0.50	$\mathbf{75.84 \pm 0.64}$	$\mathbf{89.89 \pm 0.69}$	$\mathbf{75.03 \pm 0.36}$	$\mathbf{87.74 \pm 0.39}$	$\mathbf{54.29 \pm 0.54}$	72.28 ± 1.40	50.03 ± 0.81	72.23
JOAO		$\mathbf{78.07 \pm 0.47}$	74.55 ± 0.41	77.32 ± 0.54	87.35 ± 1.02	69.50 ± 0.36	85.29 ± 1.35	55.74 ± 0.63	70.21 ± 3.08		74.75
CtrlGCL-G	JOAO	75.99 ± 0.59	74.95 ± 0.42	77.70 ± 0.85	87.12 ± 2.46	69.58 ± 0.27	86.57 ± 1.22	54.69 ± 0.73	71.46 ± 0.17		74.88
CtrlGCL-T	JOAO	73.32 ± 0.37	$\mathbf{75.44 \pm 0.54}$	76.29 ± 0.55	85.13 ± 1.79	$\mathbf{72.82 \pm 0.35}$	86.09 ± 0.94	54.63 ± 0.64	$\mathbf{71.74 \pm 1.26}$		74.43
CtrlGCL-H	JOAO	76.19 ± 0.77	75.18 ± 0.63	$\mathbf{78.27 \pm 1.32}$	$\mathbf{87.70 \pm 1.31}$	71.80 ± 0.33	$\mathbf{86.79 \pm 1.31}$	$\mathbf{56.17 \pm 0.67}$	71.66 ± 0.42		**75.47**
JOAOv2		$\mathbf{78.36 \pm 0.53}$	74.07 ± 1.10	77.40 ± 1.15	87.67 ± 0.79	69.33 ± 0.34	86.42 ± 1.45	56.03 ± 0.27	70.83 ± 0.25		75.01
CtrlGCL-G	JOAOv2	77.81 ± 0.73	75.22 ± 0.94	77.84 ± 0.84	87.82 ± 2.17	69.34 ± 0.31	87.97 ± 0.80	56.11 ± 0.33	71.68 ± 0.74		75.47
CtrlGCL-T	JOAOv2	77.37 ± 0.31	$\mathbf{75.94 \pm 0.88}$	77.67 ± 0.48	86.77 ± 1.08	$\mathbf{72.76 \pm 0.27}$	87.32 ± 0.60	55.49 ± 0.32	$\mathbf{72.52 \pm 0.79}$		75.73
CtrlGCL-H	JOAOv2	78.04 ± 0.19	75.25 ± 0.41	$\mathbf{78.37 \pm 1.26}$	$\mathbf{88.53 \pm 2.45}$	70.18 ± 0.34	$\mathbf{87.98 \pm 0.29}$	$\mathbf{56.15 \pm 0.29}$	72.50 ± 0.94		**75.87**
AutoGCL		$\mathbf{82.00 \pm 0.29}$	75.80 ± 0.36	77.57 ± 0.60	88.64 ± 1.08	70.12 ± 0.68	88.58 ± 1.49	56.75 ± 0.18	73.30 ± 0.40		76.60
CtrlGCL-G	AutoGCL	81.77 ± 0.32	75.63 ± 0.77	77.94 ± 0.85	88.84 ± 1.34	71.98 ± 0.83	88.75 ± 1.03	56.93 ± 0.28	73.87 ± 0.68		76.93
CtrlGCL-T	AutoGCL	81.04 ± 0.41	$\mathbf{76.43 \pm 0.67}$	78.29 ± 0.59	88.63 ± 1.29	72.49 ± 0.47	89.59 ± 1.48	57.27 ± 0.75	$\mathbf{73.95 \pm 0.87}$		77.21
CtrlGCL-H	AutoGCL	81.84 ± 0.53	76.38 ± 0.54	$\mathbf{78.31 \pm 1.37}$	$\mathbf{89.03 \pm 1.01}$	$\mathbf{72.68 \pm 0.23}$	$\mathbf{89.88 \pm 1.21}$	$\mathbf{57.43 \pm 0.37}$	73.94 ± 0.99		**77.44**
RGCL		78.14 ± 1.08	75.03 ± 0.43	78.86 ± 0.48	87.66 ± 1.01	70.92 ± 0.65	90.34 ± 0.58	56.38 ± 0.40	71.85 ± 0.84		76.15
CtrlGCL-G	RGCL	79.28 ± 0.94	75.28 ± 0.72	79.45 ± 0.63	$\mathbf{88.87 \pm 1.46}$	72.73 ± 0.55	90.47 ± 0.77	56.58 ± 0.41	72.19 ± 0.67		76.86
CtrlGCL-T	RGCL	78.95 ± 1.53	75.87 ± 0.45	79.12 ± 0.88	87.50 ± 1.75	73.11 ± 0.24	$\mathbf{90.90 \pm 0.62}$	56.82 ± 0.29	$\mathbf{72.75 \pm 0.66}$		76.88
CtrlGCL-H	RGCL	$\mathbf{79.42 \pm 0.82}$	$\mathbf{76.21 \pm 0.46}$	$\mathbf{79.54 \pm 1.14}$	88.79 ± 1.87	$\mathbf{73.14 \pm 0.37}$	90.75 ± 0.84	$\mathbf{57.28 \pm 0.42}$	72.61 ± 0.94		**77.22**

5.4. Memory Efficiency

To evaluate the scalability of the proposed approach, we conducted an in-depth analysis of the GPU memory efficiency of CtrlGCL on Erdos–Renyi graphs [46]. In line with the methodology employed in prior research [47], we generated Erdos–Renyi graphs by modulating the number of nodes n while maintaining the edge size m at twice the number of nodes $m = 2n$. As depicted in Figure 2, our CtrlGCL demonstrates high memory efficiency, attributable to the computational efficiency of the tree encoder. This characteristic renders it particularly practical for large-scale graph applications. Notably, a comparison of memory usage across different tree heights with the same graph size reveals that the GPU memory consumption remains relatively constant, further underscoring the scalability of our proposed approach.

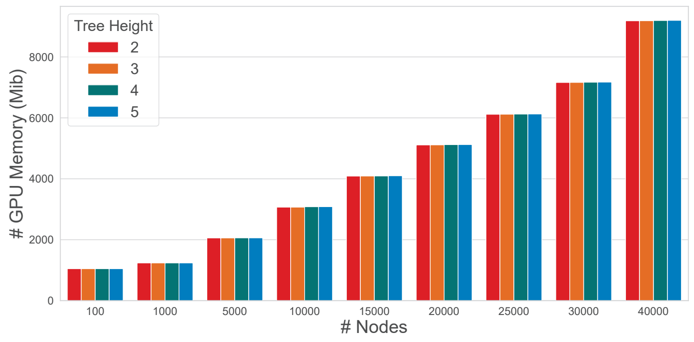


Figure 2. GPU memory efficiency of CtrlGCL with varying input graph sizes.

6. Conclusions

In this study, our focus is on exploring a good view for graph contrastive learning. Upon leveraging insights from the structural information and graph information bottleneck theory, we proposed the definition that a good view should possess minimal structural uncertainty for a graph. Taking this concept further, we introduced CtrlGCL, a practical implementation for graph contrastive learning through coding tree representations. Our approach utilized an optimization algorithm driven by structural entropy to approximate the minimization of structural uncertainty, resulting in coding trees that encapsulate essential graph information. The encoder, designed with the convolution mechanism of GNNs, was tailored for learning representations from coding trees. The effectiveness of our proposed approach was extensively validated across various benchmarks in both unsupervised and semi-supervised learning. This validation was reflected in the average ranking and average accuracy, demonstrating superior performance compared to other state-of-the-art methods. Specifically, our approach, implemented via GtrlGCL-H, yielded an average accuracy enhancement of 1.06% in the context of unsupervised learning when compared to GraphCL. In the semi-supervised learning scenario, CtrlGCL-G outperformed GraphCL, with an increase of 0.22%. Notably, in orthogonal experiments, almost all versions of CtrlGCL-H surpassed the corresponding baselines by more than 1% for average accuracy under unsupervised learning.

Despite the superiority of the proposed "good view" for graph contrastive learning based on structural entropy, the current definition of structural entropy only considers the structural information. This limitation may affect tasks such as node classification and link prediction that heavily rely on node features, potentially limiting the benefits of the proposed good view. Our future research direction entails enhancing our methods through the refinement of the structural entropy theory or by exploring the amalgamation of multiple entropy measures. The emphasis on minimizing uncertainty in node features suggests promising avenues for future research, exploration, and improvement. We look forward to continuing our work in this exciting field.

Author Contributions: Methodology, X.C.; Writing—original draft, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC (Grant No. 61932002).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

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

References

1. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 9729–9738.
2. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020; pp. 1597–1607.
3. Rezaeifar, S.; Voloshynovskiy, S.; Asgari Jirhandeh, M.; Kinakh, V. Privacy-Preserving Image Template Sharing Using Contrastive Learning. *Entropy* **2022**, *24*, 643. [CrossRef]
4. Gao, T.; Yao, X.; Chen, D. Simcse: Simple contrastive learning of sentence embeddings. *arXiv* **2021**, arXiv:2104.08821.
5. Albelwi, S. Survey on self-supervised learning: Auxiliary pretext tasks and contrastive learning methods in imaging. *Entropy* **2022**, *24*, 551. [CrossRef] [PubMed]
6. Velickovic, P.; Fedus, W.; Hamilton, W.L.; Liò, P.; Bengio, Y.; Hjelm, R.D. Deep Graph Infomax. *ICLR* **2019**, *2*, 4.
7. You, Y.; Chen, T.; Sui, Y.; Chen, T.; Wang, Z.; Shen, Y. Graph contrastive learning with augmentations. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 5812–5823.
8. Li, S.; Wang, X.; Zhang, A.; Wu, Y.; He, X.; Chua, T.S. Let Invariant Rationale Discovery Inspire Graph Contrastive Learning. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 13052–13065.
9. Tian, Y.; Sun, C.; Poole, B.; Krishnan, D.; Schmid, C.; Isola, P. What makes for good views for contrastive learning? *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 6827–6839.
10. Suresh, S.; Li, P.; Hao, C.; Neville, J. Adversarial graph augmentation to improve graph contrastive learning. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 15920–15933.
11. You, Y.; Chen, T.; Shen, Y.; Wang, Z. Graph contrastive learning automated. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021; pp. 12121–12132.
12. Feng, S.; Jing, B.; Zhu, Y.; Tong, H. Adversarial graph contrastive learning with information regularization. In Proceedings of the the ACM Web Conference 2022, Virtual, 25–29 April 2022; pp. 1362–1371.
13. Wu, T.; Ren, H.; Li, P.; Leskovec, J. Graph information bottleneck. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 20437–20448.
14. Shannon, C.E. A mathematical theory of communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
15. Brooks, F.P., Jr. Three great challenges for half-century-old computer science. *J. ACM* **2003**, *50*, 25–26. [CrossRef]
16. Li, A.; Pan, Y. Structural information and dynamical complexity of networks. *IEEE Trans. Inf. Theory* **2016**, *62*, 3290–3339. [CrossRef]
17. You, Y.; Chen, T.; Wang, Z.; Shen, Y. Bringing your own view: Graph contrastive learning without prefabricated data augmentations. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, Lyon, France, 25–29 April 2022; pp. 1300–1309.
18. Guo, Q.; Liao, Y.; Li, Z.; Liang, S. Multi-Modal Representation via Contrastive Learning with Attention Bottleneck Fusion and Attentive Statistics Features. *Entropy* **2023**, *25*, 1421. [CrossRef]
19. Hu, W.; Liu, B.; Gomes, J.; Zitnik, M.; Liang, P.; Pande, V.; Leskovec, J. Strategies For Pre-training Graph Neural Networks. In Proceedings of the International Conference on Learning Representations (ICLR), Addis Ababa, Ethiopia, 30 April 2020.
20. Sun, F.Y.; Hoffman, J.; Verma, V.; Tang, J. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In Proceedings of the International Conference on Learning Representations, Addis Ababa, Ethiopia, 30 April 2020.
21. Li, W.; Zhu, E.; Wang, S.; Guo, X. Graph Clustering with High-Order Contrastive Learning. *Entropy* **2023**, *25*, 1432. [CrossRef] [PubMed]
22. Wei, C.; Liang, J.; Liu, D.; Wang, F. Contrastive Graph Structure Learning via Information Bottleneck for Recommendation. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 20407–20420.
23. Cai, X.; Huang, C.; Xia, L.; Ren, X. LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
24. Mowshowitz, A.; Dehmer, M. Entropy and the complexity of graphs revisited. *Entropy* **2012**, *14*, 559–570. [CrossRef]
25. Raychaudhury, C.; Ray, S.; Ghosh, J.; Roy, A.; Basak, S. Discrimination of isomeric structures using information theoretic topological indices. *J. Comput. Chem.* **1984**, *5*, 581–588. [CrossRef]
26. Bianconi, G. Entropy of network ensembles. *Phys. Rev. E* **2009**, *79*, 036114. [CrossRef] [PubMed]
27. Dehmer, M. Information processing in complex networks: Graph entropy and information functionals. *Appl. Math. Comput.* **2008**, *201*, 82–94. [CrossRef]
28. Braunstein, S.L.; Ghosh, S.; Severini, S. The Laplacian of a graph as a density matrix: A basic combinatorial approach to separability of mixed states. *Ann. Comb.* **2006**, *10*, 291–317. [CrossRef]
29. Li, A.L.; Yin, X.; Xu, B.; Wang, D.; Han, J.; Wei, Y.; Deng, Y.; Xiong, Y.; Zhang, Z. Decoding topologically associating domains with ultra-low resolution Hi-C data by graph structural entropy. *Nat. Commun.* **2018**, *9*, 3265. [CrossRef]
30. Gilmer, J.; Schoenholz, S.S.; Riley, P.F.; Vinyals, O.; Dahl, G.E. Neural message passing for quantum chemistry. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 1263–1272.
31. Ma, Y.; Tang, J. *Deep Learning on Graphs*; Cambridge University Press: Cambridge, UK, 2021.
32. Shannon, C. The lattice theory of information. *Trans. IRE Prof. Group Inf. Theory* **1953**, *1*, 105–107. [CrossRef]

33. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
34. Morris, C.; Kriege, N.M.; Bause, F.; Kersting, K.; Mutzel, P.; Neumann, M. TUDataset: A collection of benchmark datasets for learning with graphs. *arXiv* **2020**, arXiv:2007.08663.
35. Shervashidze, N.; Vishwanathan, S.; Petri, T.; Mehlhorn, K.; Borgwardt, K. Efficient graphlet kernels for large graph comparison. In Proceedings of the Artificial Intelligence and Statistics, PMLR, Clearwater Beach, FL, USA, 16–18 April 2009; pp. 488–495.
36. Shervashidze, N.; Schweitzer, P.; Van Leeuwen, E.J.; Mehlhorn, K.; Borgwardt, K.M. Weisfeiler-lehman graph kernels. *J. Mach. Learn. Res.* **2011**, *12*, 2539–2561.
37. Yanardag, P.; Vishwanathan, S. Deep graph kernels. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, 10–13 August 2015; pp. 1365–1374.
38. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
39. Adhikari, B.; Zhang, Y.; Ramakrishnan, N.; Prakash, B.A. Sub2vec: Feature learning for subgraphs. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Melbourne, VIC, Australia, 3–6 June 2018; pp. 170–182.
40. Annamalai, N.; Mahinthan, C.; Rajasekar, V.; Lihui, C.; Yang, L.; Jaiswal, S. graph2vec: Learning Distributed Representations of Graphs. In Proceedings of the 13th International Workshop on Mining and Learning with Graphs (MLG), Halifax, NS, Canada, 14 August 2017.
41. Kipf, T.N.; Welling, M. Variational Graph Auto-Encoders. *arXiv* **2016**, arXiv:1611.07308v1.
42. Yin, Y.; Wang, Q.; Huang, S.; Xiong, H.; Zhang, X. AutoGCL: Automated graph contrastive learning via learnable view generators. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 8892–8900. [CrossRef]
43. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
44. Fan, R.E.; Chang, K.W.; Hsieh, C.J.; Wang, X.R.; Lin, C.J. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.* **2008**, *9*, 1871–1874.
45. Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How Powerful are Graph Neural Networks? In Proceedings of the ICLR, New Orleans, LA, USA, 6–9 May 2019.
46. Erdős, P.; Rényi, A. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* **1960**, *5*, 17–60.
47. Baek, J.; Kang, M.; Hwang, S.J. Accurate Learning of Graph Representations with Graph Multiset Pooling. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 4 May 2021.

entropy

Review

Deep Learning for 3D Reconstruction, Augmentation, and Registration: A Review Paper

Prasoon Kumar Vinodkumar [1], Dogus Karabulut [1], Egils Avots [1,*], Cagri Ozcinar [1] and Gholamreza Anbarjafari [1,2,3,4,*]

1 iCV Lab, Institute of Technology, University of Tartu, 50090 Tartu, Estonia; prasoon.vinodkumar@ut.ee (P.K.V.); dogus.karabulut@ut.ee (D.K.); chagri.ozchinar@ut.ee (C.O.)
2 PwC Advisory, 00180 Helsinki, Finland
3 iVCV OÜ, 51011 Tartu, Estonia
4 Institute of Higher Education, Yildiz Technical University, Beşiktaş, Istanbul 34349, Turkey
* Correspondence: egils.avots@ut.ee (E.A.); shb@ut.ee (G.A.); Tel.: +372-737-4855 (G.A.)

Abstract: The research groups in computer vision, graphics, and machine learning have dedicated a substantial amount of attention to the areas of 3D object reconstruction, augmentation, and registration. Deep learning is the predominant method used in artificial intelligence for addressing computer vision challenges. However, deep learning on three-dimensional data presents distinct obstacles and is now in its nascent phase. There have been significant advancements in deep learning specifically for three-dimensional data, offering a range of ways to address these issues. This study offers a comprehensive examination of the latest advancements in deep learning methodologies. We examine many benchmark models for the tasks of 3D object registration, augmentation, and reconstruction. We thoroughly analyse their architectures, advantages, and constraints. In summary, this report provides a comprehensive overview of recent advancements in three-dimensional deep learning and highlights unresolved research areas that will need to be addressed in the future.

Keywords: deep learning; 3D reconstruction; 3D augmentation; 3D registration; point cloud; voxel; neural networks; convolutional neural networks; graph neural networks; generative adversarial networks; review

Citation: Vinodkumar, P.K.; Karabulut, D.; Avots, E.; Ozcinar, C.; Anbarjafari, G. Deep Learning for 3D Reconstruction, Augmentation, and Registration: A Review Paper. *Entropy* **2024**, *26*, 235. https://doi.org/10.3390/e26030235

Academic Editor: Amelia Carolina Sparavigna

Received: 13 November 2023
Revised: 1 March 2024
Accepted: 5 March 2024
Published: 7 March 2024

1. Introduction

Autonomous navigation, domestic robots, the reconstruction of architectural models of buildings, facial recognition, the preservation of endangered historical monuments, the creation of virtual environments for the film and video game industries, and augmented/virtual reality are just a few examples of real-world applications that depend heavily on the identification of 3D objects based on point clouds. A rising number of these applications require three-dimensional (3D) data. Processing 3D data reliably and effectively is critical for these applications. A powerful method for overcoming these obstacles is deep learning. In this review paper, we concentrate on deep learning methods for *reconstruction*, *augmentation*, and *registration* in three dimensions.

The processing of 3D data employs a wide range of strategies to deal with unique problems. *Registration*, which entails matching several point clouds to a single coordinate system, is one key issue. While conventional approaches rely on geometric changes and parameter optimisation, deep learning provides an all-encompassing approach with promising outcomes. *Augmentation* is another technique for deep learning employed in 3D data processing, and it entails transforming current data while maintaining the integrity of the underlying information to produce new data. Since augmentation may provide new data points that enhance the accuracy and quality of the data, it is a useful technique for resolving problems with data quality and completeness. The final technique in this analysis is called *reconstruction*, which entails building a 3D model from a collection of 2D photos or a 3D point cloud. This is a difficult task since 3D geometry is complicated and 3D data lack spatial order. In order to increase the accuracy and effectiveness of reconstruction, deep learning algorithms have made substantial advancements in this field by proposing novel architectures and loss functions. Overall, these methods have shown promise in resolving the difficulties involved in interpreting 3D data and enhancing the accuracy and value of 3D data.

1.1. Our Previous Work

We have previously conducted [1] an in-depth review of recent advancements in deep learning approaches for 3D object identification, including 3D object segmentation, detection, and classification methods. The models covered in our earlier article were selected based on a number of factors, including the datasets on which they were trained and/or assessed, the category of methods to which they belong, and the tasks they carry out, such as segmentation and classification. The majority of the models that we surveyed in our earlier study were validated, and their results were compared with state-of-the-art technologies using benchmark datasets such as SemanticKITTI [2] and Stanford 3D Large-Scale Indoor Spaces (S3DIS) [3]. We discussed in detail some of the most advanced and/or benchmarking deep learning methods for 3D object recognition in our earlier work. These methods covered a range of 3D data formats, such as RGB-D (IMVoteNet) [4], voxels (VoxelNet) [5], point clouds (PointRCNN) [6], mesh (MeshCNN) [7], and 3D video (Meta-RangeSeg) [1,8].

1.2. Research Methodology

In this paper, we provide a comprehensive overview of recent advances in deep-learning-based 3D object reconstruction, registration, and augmentation as a follow-up to our earlier research [1]. It concentrates on examining frequently employed building components, convolution kernels, and full architectures, highlighting the benefits and drawbacks of each model. Over 37 representative papers that include 32 benchmark and state-of-the-art models and five benchmark datasets that have been used by many models over the last five years are included in this study. Additionally, we review six benchmark models related to point cloud completion over the last five years. We selected these papers based on the number of citations and implementations by other researchers in this field of study. Despite the fact that certain notable 3D object recognition and reconstruction surveys, such as those on RGB-D semantic segmentation and 3D object reconstruction, have been published, these studies do not exhaustively cover all 3D data types and common application domains. Most importantly, these surveys only provide a general overview of 3D object recognition techniques, including some of their advantages and limitations. The current developments in these machine learning models and their potential to enhance the accuracy, speed, and effectiveness of 3D registration, augmentation, and reconstruction are the main reasons for our selection of these particular models. In real-world situations, the use of many of these models in a pipeline has the potential to improve performance even more significantly and achieve even better outcomes.

2. 3D Data Representations

2.1. Point Clouds

Raw 3D data representations, like point clouds, can be obtained using many scanning technologies, such as Microsoft Kinect, structured light scanning, and many more. Point clouds have their origins in photogrammetry and, more recently, in LiDAR. A collection of randomly arranged points in three dimensions, known as a point cloud, resembles the geometry of three-dimensional objects. The implementation of these points results in a non-Euclidean geometric data format. A further way to describe point clouds is to describe a collection of small Euclidean subsets with a common coordinate system, global parametrisation, and consistency in translation and rotation. As a result, determining the structure of point clouds depends on whether the object's global or local structure is taken into account. A point cloud can be used for a range of computer vision applications, including classification and segmentation, object identification, reconstruction, etc. It is conceptualised as a collection of unstructured 3D points that describe the geometry of a 3D object.

Such 3D point clouds can be easily acquired, but processing them can be challenging. Applying deep learning to 3D point cloud data is riddled with difficulties. These issues include point alignment issues, noise/outliers (unintended points), and occlusion (due to congregated scenery or blindsides). Table 1 provides the list of 3D reconstruction models using point cloud representation reviewed in this study. The following, however, are the most significant challenges in applying deep learning to point clouds:

Irregular: Depending on how evenly the points are sampled over the various regions of an object or scene, point cloud data may include dense or sparse points in different parts of an item or scene. Techniques for subsampling can minimise irregularity, but they cannot get rid of it entirely.

Unordered: The collection of points acquired around the objects in a scene is called a point cloud, and it is frequently preserved as a list in a file. These points are earned by interacting with the objects in the scenario. The set itself is referred to as being permutation-invariant since the scene being shown remains constant regardless of the order in which the points are arranged.

Unstructured: A point cloud's data are not arranged on a conventional grid. The distance between each point and its neighbours is individually scanned; therefore, it is not always constant. The space between two adjacent pixels in a picture, on the other hand, remains constant and can only be represented by a two-dimensional grid.

Table 1. 3D reconstruction models using point cloud data representation.

Model	**Dataset**	**Data Representation**
PointOutNet [9]	ShapeNet [10], 3D-R2N2 [11]	Point Cloud
Pseudo-renderer [12]	ShapeNet [10]	Point Cloud
RealPoint3D [13]	ShapeNet [10], ObjectNet3D [14]	Point Cloud
Cycle-consistency-based approach [15]	ShapeNet [10], Pix3D [16]	Point Cloud
3D34D [17]	ShapeNet [10]	Point Cloud
Unsupervised learning of 3D structure [18]	ShapeNet [10], MNIST3D [19]	Point Cloud

2.2. Voxels

Using three-dimensional volumes is an alternative way of representing three-dimensional surfaces using a grid of constant size and dimensions. Three-dimensional data can be represented as a regular grid in three-dimensional space. Voxels are a three-dimensional data description method that defines how an object in three-dimensional space is spread across all three dimensions of a scene. Voxels are used to model 3D data by defining the distribution of the 3D object across the scene's three dimensions. By identifying the occupied voxels as visible, occluded, or self-occluded, viewpoint information about the 3D shape may also be conveyed. Encoding the view information for a 3D shape enables the occupied voxels to be classified as either visible blocks or self-occluded voxels. These grids are maintained either as a binary occupancy grid, where the cell values represent the voxel occupancy, or as a signed distance field, where the voxels represent the distances to the zero-level set that represents the surface boundary. The binary occupancy grid is the more prevalent storage format of the two. Table 2 provides the list of 3D reconstruction models using voxel representation reviewed in this study.

Despite the simplicity of the voxel-based representation and its capacity to encode information about the 3D shape and its viewpoint, it is constrained by one main constraint:

Inefficient: The inefficiency of voxel-based representation stems from the fact that it represents both occupied and unoccupied portions of a scene, which creates an excessive need for memory storage. This is why voxel-based representations are unsuitable for high-resolution data representation.

Table 2. 3D reconstruction models using voxel data representation.

Models	**Dataset**	**Data Representation**
GenRe [20]	ShapeNet [10], Pix3D [16]	Voxels
MarrNet [21]	ShapeNet [10], PASCAL3D+ [22]	Voxels
Perspective Transformer Nets [23]	ShapeNet [10]	Voxels
Rethinking reprojection [24]	ShapeNet [10], PASCAL3D+ [22], SUN [25], MS COCO [26]	Voxels
3D-GAN [27]	ModelNet [28], IKEA [29]	Voxels
Pix2Vox++ [30]	ShapeNet [10], Pix3D [16], Things3D [30]	Voxels
3D-R2N2 [11]	ShapeNet [10], PASCAL3D+ [22], MVS CAD 3D [11]	Voxels
Weak recon [31]	ShapeNet [10], ObjectNet3D [14]	Voxels
Relative viewpoint estimation [32]	ShapeNet [10], Pix3D [16], Things3D [30]	Voxels

2.3. Meshes

3D meshes are one of the most commonly used ways to represent 3D shapes. A 3D mesh structure is composed of a set of polygons called faces, which are represented in terms of a set of vertices that describe the mesh's coordinates in 3D space. The connection list associated with these vertices describes how they are connected to one another. Following the grid-structured data, the local geometry of the meshes can be described as a subset of Euclidean space. Table 3 provides the list of 3D reconstruction models using mesh representation reviewed in this study.

Meshes are non-Euclidean data where the known properties of the Euclidean space, such as shift-invariance, operations of the vector space, and the global parametrisation system, are not well defined. Learning from 3D meshes is difficult for two key reasons:

Irregular: Deep learning approaches have not been effectively extended to such irregular representations, and 3D meshes are highly complex.

Low quality: In addition, such data typically contain noise, missing data, and resolution issues. Figure 1 shows 3D data representations of the Stanford Bunny [33] dataset with point cloud, voxel, and mesh data representations [34].

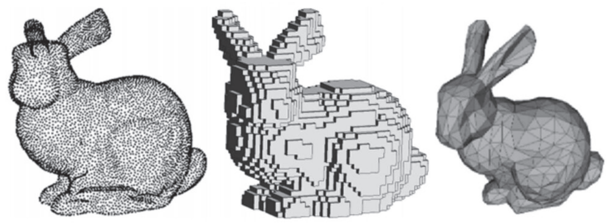


Figure 1. The 3D data representations of the Stanford Bunny [33] model: point cloud (**left**), voxels (**middle**), and 3D mesh (**right**) [34].

Table 3. 3D reconstruction models using mesh data representation.

Model	**Dataset**	**Data Representation**
Neural renderer [35]	ShapeNet [10]	Meshes
Residual MeshNet [36]	ShapeNet [10]	Meshes
Pixel2Mesh [37]	ShapeNet [10]	Meshes
CoReNet [38]	ShapeNet [10]	Meshes

3. 3D Benchmark Datasets

The datasets used in deep learning for 3D registration, augmentation, and reconstruction significantly influence the model's accuracy and effectiveness. In order to train and assess deep learning models for 3D registration, augmentation, and reconstruction, it is imperative to have access to a wider variety of representative datasets. Future studies should concentrate on creating larger and more realistic datasets that include a variety of real-world objects and environments. For 3D registration, augmentation, and reconstruction, this would make it possible to develop even deeper learning models that are more reliable and accurate. This article will only list the most common datasets that have been used by the 3D object registration, augmentation, and reconstruction models discussed in this survey paper in Sections 3 (3D reconstruction), 4 (3D registration), and 5 (3D augmentation). This includes the ModelNet [28], PASCAL3D+ [22], ShapeNet [10], ObjectNet3D [14] and ScanNet [39] datasets. Datasets that are specific only to some 3D recognition models will not be included in this survey. Table 4 provides the properties of data provided by different datasets.

Table 4. Benchmarking datasets included in this survey.

Datasets	**Number of Frames**	**Number of Labels**	**Object Type**	**5 Common Classes**
ModelNet [28]	151,128	660	3D CAD Scans	Bed, Chair, Desk, Sofa, Table
PASCAL3D+ [22]	30,899	12	3D CAD Scans	Boat, Bus, Car, Chair, Sofa

Table 4. *Cont.*

Datasets	Number of Frames	Number of Labels	Object Type	5 Common Classes
ShapeNet [10]	220,000	3135	Scans of Artefact, Plant, Person	Table, Car, Chair, Sofa, Rifle
ObjectNet3D [14]	90,127	100	Scans of Artifact, Vehicles	Bed, Car, Door, Fan, Key
ScanNet [39]	2,492,518	1513	Scans of Bedrooms, Kitchens, Offices	Bed, Chair, Door, Desk, Floor

3.1. ModelNet

By combining 3D CAD models from 3D Warehouse, 261 CAD model websites indexed with the Yobi3D search engine, common item categories searched from the SUN database [25], models from the Princeton Shape Benchmark [40], and models from the SUN database that contain at least 20 object instances per category, ModelNet [28] is a large-scale object collection of 3D computer graphics CAD models. Both the total number of categories and the total number of occurrences per category were constrained in a number of earlier CAD datasets. The writers thoroughly examined each 3D model and removed extraneous elements from each CAD model, such as the floor and thumbnail images, such that each mesh model had just one item from the designated category. ModelNet is almost 22 times larger than the Princeton Shape Benchmark [40] which contains 151,128 3D CAD models representing 660 distinct item categories. ModelNet10 and ModelNet40 are mostly used for classifying and recognising objects.

3.2. PASCAL3D+

Each of the 12 categories of 3D stiff objects that can be found in PASCAL3D+ [22] contains more than 3000 individual items. Pose estimation and the detection of 3D objects are also possible applications for the dataset. In addition to that, it might function as a baseline for the community. Images from PAS-CAL show a lot more diversity and more closely resemble actual situations. As a result, this dataset is less skewed than those that are gathered in controlled environments. Viewpoint annotations are continuous and dense in this dataset. The perspective is usually discretised into numerous bins in the current 3D datasets. Consequently, detectors that have been trained on this dataset may be more broadly capable. The objects in this collection are truncated and occluded; such objects are typically disregarded in the 3D datasets available today. Three-dimensional annotations are added to 12 rigid categories in the PASCAL VOC 2012 [41] dataset using PASCAL3D+. A selection of CAD models that cover intra-class variability are downloaded for each category. The closest CAD model in terms of 3D geometry is then linked to each occurrence of an object inside the category. Additionally, a number of 3D landmarks inside these CAD models have been discovered, and annotators have labelled the landmarks' 2D positions. Eventually, an accurate continuous 3D posture for each item in the collection is generated utilising the 3D–2D correspondences of the landmarks. Consequently, the CAD model that corresponds with each item, along with 2D landmarks and the 3D continuous position, makes up its annotation.

3.3. ShapeNet

More than 50,000 CAD models are available in ShapeNet [10], a significant collection of shapes organised into 55 categories. Additionally, there are annotations for semantic features and categories. This large dataset consists of semantic category labels for models, rigid alignments, parts, bilateral symmetry planes, physical sizes, and keywords, in addition to further recommended annotations. ShapeNet had over 3 million models indexed when the dataset was released, and 220,000 models had been categorised into 3140 categories. ShapeNetCore is a subset of ShapeNet, which has over 51,300 unique 3D models. There are annotations for 55 common item categories. ShapeNetSem is a subset of ShapeNet, which includes 12,000 models. It is more condensed yet has 270 more thorough categories. By making ShapeNet the first large-scale 3D shape dataset of its sort, it has advanced computer graphics research in the direction of data-driven research, building on recent advancements in vision and NLP. It has also supported a wide class of newly revived machine learning and neural network approaches for applications dealing with geometric data by offering a large-scale, extensively annotated dataset.

3.4. ObjectNet3D

Despite having 30,899 photos, PASCAL3D+ [22] is still unable to fully capture the variances of common item categories and their geometric variety due to its limitation in the number of object classes (12 total) and 3D forms (79 total). A large-scale 3D object collection with more item categories, more 3D forms per class, and precise image-shape correspondences is provided by ObjectNet3D [14]. This dataset is comprised of a total of 90,127 photos in 100 distinct categories. Annotations pertaining to the 3D posture as well as the shape of each 2D object found in the photographs are provided. It is also useful for problems involving the development of proposals, the detection of objects in two dimensions, and the estimation of poses in three dimensions. For the automotive category, for instance, 3D forms of sedans, SUVs, vans, trucks, etc., are provided. The sizes of these three-dimensional forms have been normalised to fit [1] within a unit sphere, and they have been oriented in accordance with the category's primary axis (e.g., front view of a bench). Additionally, each 3D form has a set of personally chosen keypoints that may be used to identify significant points in photos or 3D shapes. In total, 783 3D shapes from all 100 categories have been gathered in this manner.

3.5. ScanNet

ScanNet [39] is a collection of RGB-D scans of real-world locations with extensive annotations. It contains 2.5 million RGB-D pictures from 1513 scans taken in 707 different settings. Due to its annotation with approximated calibration parameters, camera postures, 3D surface reconstructions, textured meshes, dense object-level semantic segmentations, and aligned CAD models, the scope of this research is substantial. A capture pipeline is created to make it simpler for novices to obtain semantically labelled 3D models of situations in order to establish a framework that enables many individuals to gather and annotate enormous amounts of data. Data are collected, and off-line processing is performed on RGB-D video. The scene is completely 3D reconstructed and semantically labelled. With the use of ScanNet data, 3D deep networks can be trained, and their performance on a variety of scene comprehension tasks, such as 3D object categorisation, semantic voxel labelling, and CAD model retrieval, can be assessed. ScanNet has several different kinds of places, including offices, homes, and bathrooms. A versatile framework for RGB-D acquisition and semantic annotations is offered by ScanNet. Cutting-edge performance on a number of 3D scene interpretation tasks is made possible with the support of ScanNet's fully annotated scan data. Finally, crowdsourcing employing semantic annotation tasks is used to collect instance-level item category annotations and 3D CAD model alignments for reconstruction. The RBG-D reconstruction and semantic annotation framework is shown in Figure 2.

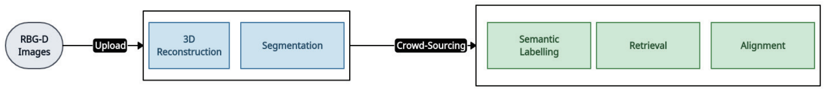


Figure 2. RBG-D reconstruction and semantic annotation framework of ScanNet [39] dataset.

Similar to our previous work [1], to determine which model performs better with each of these datasets, we attempted to compare the performance of the models that use them. While some of the models analysed in this study concentrate on computation time (measured in milliseconds), others focus on performance metrics like accuracy and precision. The majority of these models have assessed their efficacy using visual shape identification of the objects rather than numerical values. As a result, we were unable to compare the performance of these models using the datasets provided.

4. Object Reconstruction

Two types of traditional 3D reconstruction techniques exist: model-driven and data-driven techniques. The goal of the model-driven approaches is to align the item types in a library with the geometry of the objects created using digital surface models (DSMs), such as point clouds [42]. By using this method, the topological correctness of the rebuilt model can be guaranteed; nevertheless, issues might arise if the object shape has no candidates in the library. Additionally, the production accuracy is decreased by model-driven procedures since they only use a small fraction of the pre-defined shapes that are provided in the model libraries. Furthermore, modelling complicated object structures might not be possible. A DSM (often in the form of a point cloud) is used as the main data source in data-driven approaches, and the models are created from these data overall, without focusing on any one parameter. The primary issue with the data-driven technique is the possibility of unsuccessful segment extraction, which could result in topological or geometrical errors throughout the intersection process. Typically, data-driven techniques lack robustness and are extremely susceptible to data noise. Because data-driven methods are sensitive to noise, pre-processing data is a crucial step in preventing inaccurate outcomes [43].

4.1. Procedural-Based Approaches

The extensive and demanding field of automated reconstruction of 3D models from point clouds has attracted significant attention in the fields of photogrammetry, computer vision, and computer graphics due to its potential applications in various domains, including construction management, emergency response, and location-based services [44]. However, the intrinsic noise and incompleteness of the data provide a hurdle to the automated construction of the 3D models and necessitate additional research. These methods extract 3D geometries of structures, such as buildings, solely through a data-driven process that is highly dependent on the quality of the data [45,46].

Procedural-based techniques use shape grammars to reconstruct interior spaces while taking advantage of architectural design principles and structural organisation [47,48]. Because these methods take advantage of the regularity and recurrence of structural parts and architectural design principles in the reconstruction, they are more resilient to data incompleteness and uncertainty. Shape grammars are widely and successfully utilised in the field of urban reconstruction for 3D synthesising architecture (e.g., building façades) [49]. This procedural-based strategy is less sensitive to inaccurate and partial data than the data-driven alternatives. Several academics have successfully proposed shape grammars based on integration with a data-driven method to procedurally recreate building façade models from observation data (i.e., photos and point clouds) in order to reconstruct models of real settings [50,51].

However, because indoor and outdoor contexts differ from one another, the façade grammars cannot be used directly there. The translation of architectural design knowledge and principles into a grammar form, which guarantees the topological accuracy of the rebuilt elements and the plausibility of the entire model, is generally where shape-grammar-based systems have their advantages [44]. A set of grammar rules is necessary for procedural-based approaches, and in the grammar-based indoor modelling techniques currently in use, the parameters and rule application sequence are manually specified. However, these techniques are frequently restricted to straightforward architectural designs, such as the Manhattan design [48,52].

4.2. Deep-Learning-Based Approaches

Artificial intelligence (AI) is profoundly altering the way the geographical domain functions [53]. There is hope that the constraints of traditional 3D modelling and reconstruction techniques can be solved by the recently established deep learning (DL) technologies. In recent years, there has been a lot of study on 3D reconstruction using deep learning, with numerous articles covering the subject. Comparing the DL approaches to the traditional methods, state-of-the-art results were obtained [54–56]. With the recent rapid growth in 3D building models and the availability of a wide variety of 3D shapes, DL-based 3D reconstruction has become increasingly practical. It is possible to train DL models to recognise 3D shapes and all of their attributes [43].

Computational models with several processing layers can learn data representations at different levels of abstraction using deep learning (DL) [57]. The two primary issues with traditional 3D reconstruction techniques are as follows. Initially, they require numerous manual designs, which may result in a build-up of errors, but they are barely capable of automatically picking up on the semantic aspects of 3D shapes. Second, they rely heavily on the calibre and content of the images in addition to a properly calibrated camera. By employing deep networks to automatically learn 3D shape semantics from pictures or point clouds, DL-based 3D reconstruction techniques go beyond these obstacles [43,58].

4.3. Single-View Reconstruction

Over the years, single-image-based 3D reconstruction has progressed from collecting geometry and texture information from limited types of images to learning neural network parameters to estimate 3D shapes. Real progress in computational efficiency, reconstruction performance, and generalisation capability of 3D reconstruction has been demonstrated. The very first deep-learning-based approaches required real 3D shapes of target objects as supervision, which were extremely difficult to obtain at the time. Some researchers have created images from CAD models to extend datasets; nevertheless, such synthesised data lead to a lack of generalisation and authenticity in the reconstruction results. Some studies have used ground truth 2D and 2.5D projections as supervision and reduced reprojection losses throughout the learning process, such as contour, surface normal, and so on. Later, techniques that compared projections of the reconstructed results with the input to minimise the difference required less supervision. Overall, the field of single-image-based 3D reconstruction is rapidly evolving, and the development of new techniques and architectures is paving the way for more accurate and efficient reconstruction methods. Table 5 provides the list of single-view 3D reconstruction models reviewed in this study.

Table 5. Single-view 3D reconstruction models reviewed in this study.

Nr.	Model	Dataset	Data Representation
1	PointOutNet [9]	ShapeNet [10], 3D-R2N2 [11]	Point Cloud
2	Pseudo-renderer [12]	ShapeNet [10]	Point Cloud
3	RealPoint3D [13]	ShapeNet [10], ObjectNet3D [14]	Point Cloud
4	Cycle-consistency-based [15] approach	ShapeNet [10], Pix3D [16]	Point Cloud

Table 5. *Cont.*

Nr.	Model	Dataset	Data Representation
5	GenRe [20]	ShapeNet [10], Pix3D [16]	Voxels
6	MarrNet [21]	ShapeNet [10], PASCAL3D+ [22]	Voxels
7	Perspective Transformer [23] Nets	ShapeNet [10]	Voxels
8	Rethinking reprojection	ShapeNet [10], PASCAL3D+ [22], SUN [25], MS COCO [26]	Voxels
9	3D-GAN [24]	ModelNet [28], IKEA [29]	Voxels
10	Neural renderer [35]	ShapeNet [10]	Meshes
11	Residual MeshNet [36]	ShapeNet [10]	Meshes
12	Pixel2Mesh [37]	ShapeNet [10]	Meshes
13	CoReNet [38]	ShapeNet [10]	Meshes

4.3.1. Point Cloud Representation

PointOutNet [9]: When compared to voxels, a point cloud is a sparse and memory-saving representation. PointOutNet was proposed to reconstruct objects from a single image in early methods that used point clouds as the output of deep learning networks. PointOutNet has a convolution encoder and two parallel predictor branches. The encoder receives an image as well as a random vector that throws off the prediction. One of the branches is a fully connected branch that captures complex structures, while another is a deconvolution branch that generates point coordinates. This network makes good use of geometric continuity and can produce smooth objects. This research introduced the chamfer distance loss, which is invariant to the permutation of points. This loss function has been adopted by many other models as a regulariser [59–61]. The system structure of the PointOutNet model is shown in Figure 3. With the distributional modelling module plugged in, this system may produce several predictions.

Pseudo-renderer [12]: The authors of the pseudo-renderer model use 2D convolutional operations to gain improved efficiency. First, they employ a generator to predict 3D buildings at unique view points from a single image. They then employ a pseudo-renderer to generate depth images of corresponding views, which are later used for joint 2D projection optimisation. They predict denser, more accurate point clouds. However, there is usually a limit to the number of points that cloud-based representations can accommodate [62]. When calculating the colour of a pixel, occlusion is taken into consideration by determining a weighted sum of the points' colours depending on the points' effects. In order to avoid optimising the occluded points, this model chooses the point that is closest to the camera for a particular pixel [63]. This study uses 2D supervision in addition to 3D supervision to obtain multiple projection images from various viewpoints of the generated 3D shape for optimisation by using a combination of binary cross-entropy loss function with L1 loss function [64]. The pseudo-renderer model's pipeline is depicted in Figure 4. The authors suggest using a structure generator based on 2D convolutional processes to predict the 3D structure at N perspectives from an encoded latent representation. The 3D structure at each perspective is transformed to the canonical coordinates in order to merge the point clouds. The pseudo-renderer creates depth pictures from fresh perspectives and then uses them to jointly optimise 2D projection. This is based just on 3D geometry and has no learnable parameters.

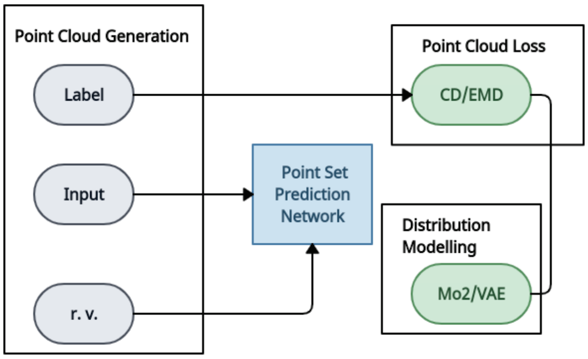


Figure 3. System structure of PointOutNet [9] model.

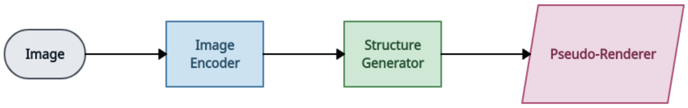


Figure 4. Pipeline of pseudo-renderer [12] model.

RealPoint3D [13]: The authors of the RealPoint3D model built fine-grained point clouds using a nearby 3D shape as an auxiliary input to the reconstruction network. By giving instructions to the closest form from the ShapeNet, RealPoint3D attempts to recreate 3D models from nature photographs with complicated backgrounds [65,66]. To integrate 2D and 3D features adaptively, the model introduces an attention-based 2D–3D fusion module into the network. By projecting the pixel information from a given 2D image into a 3D space, the method creates point cloud data. It then calculates the chamfer distance and produces a projection loss between the generated and actual point cloud data. The network itself is made up of a 2D–3D fusion module, a decoding section, and an encoding section. The input image's 2D features and the input point cloud data's 3D features are extracted throughout the encoding process. The preceding step's image and spatial characteristics are generated by the 2D–3D fusion module. Finally, the object's anticipated 3D point clouds are produced by the decoding phase [67]. Figure 5 shows the network architecture of the RealPoint3D model.

A cycle-consistency-based approach [15]: The authors of this model reconstruct point clouds from images of a certain class, each with appropriate foreground masks. They train the networks in a self-supervised manner using a geometric loss and a pose cycle consistency loss based on an encoder-to-decoder structure, as it is expensive and difficult to collect training data with ground truth 3D annotations. The training impact of multi-view supervision using a single-view dataset is simulated by employing training images with comparable 3D shapes. In addition to two cycle-consistency losses for poses and 3D reconstructions, this model adds a loss-ensuring cross-silhouette consistency [68]. This model uses cycle consistency, which was introduced in CycleGAN [69], to prevent unsupervised learning from annotating 2D and 3D data. It may, however, produce deformed body structures or out-of-view images if unaware of the previous distribution of the 3D features, which would interfere with the training process. Viewed as a basic self-supervised technique, cycle consistency uses the original encoded attribute as the generated image's 3D annotation [70]. In an analysis-by-synthesis approach, this model uses a differentiable renderer to infer a 3D shape without using ground truth 3D annotation [71]. Figure 6 shows an overview of the cycle-consistency-based approach.

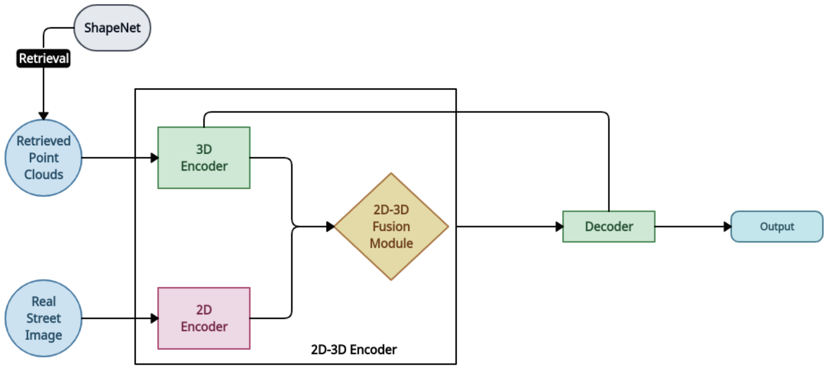


Figure 5. Network architecture of RealPoint3D [13] model.

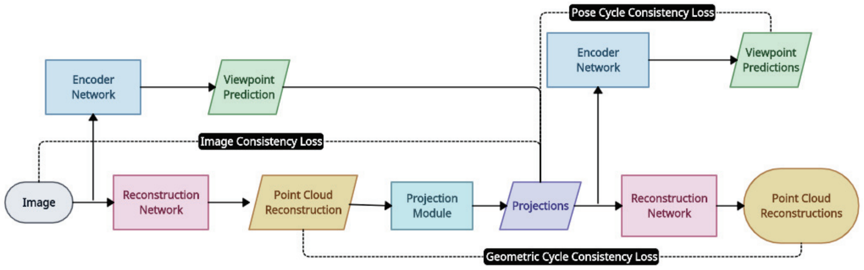


Figure 6. Overview of cycle-consistency-based approach [15].

Point-based techniques use less memory, but since they lack connection information, they need extensive postprocessing [72]. Although point clouds are simple 3D representations, they ignore topological relationships [62]. Since point clouds lack a mesh connection structure, further processing is required in order to extract the geometry from the 3D model using this representation [73].

4.3.2. Voxel Representation

GenRe [20]: A voxel representation is an early 3D representation that lends itself well to convolutional operations. The authors of GenRe train their networks with 3D supervision to predict a depth from a given image in the same view and estimate a single-view spherical map from the depth. They then employ a voxel refinement network to merge two projections and generate a final reconstruction result. This model predicts a 3D voxel grid directly from RGB-D photos using the shape completion approach. This research produces a generalisable and high-quality single-image 3D reconstruction. Others use less supervision in the learning procedure instead of needing 3D ground truth. This model divides the process of converting 2.5D to 3D form into two phases: partial 3D completion and complete 3D completion. This approach differs from the method of directly predicting the 3D shape from 2.5D. To represent the whole surface of the object, the model processes the depth map in turn using an inpainted spherical map and a partial spherical map. Ultimately, the 3D shape is produced by the voxel reconstruction network by combining the back projection of the inpainted spherical image with the depth map. On untrained classes, experimental results demonstrate that the network can also produce outcomes that are more in line with ground truth. These algorithms can rebuild 3D objects with resolutions of up to $128 \times 128 \times 128$ and more detailed reconstruction outcomes. Still, there is a significant difference when it comes to the appearance of actual 3D models [64]. Higher resolutions have been used by this model at the expense of sluggish training or lossy 2D projections, as well as small training batches [74]. Learning-based techniques are usually assessed on new instances from the same category after being trained in a category-specific manner. That said, this approach calls itself category-agnostic [75]. Figure 7 shows the network architecture of the GenRe model.

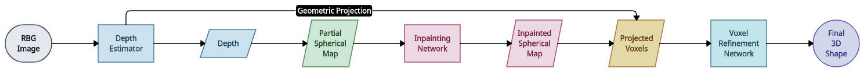


Figure 7. Network architecture of GenRe [20] model.

MarrNet [21]: This model uses depth, normal map, and silhouette as intermediate results to reconstruct 3D voxel shapes and predicts 3D shapes using a reprojection consistency loss. MarrNet contains three key components: (a) 2.5D sketch estimation, (b) 3D shape estimation, and (c) a reprojection consistency loss. From a 2D image, MarrNet initially generates object normal, depth, and silhouette images. The 3D shape is then extrapolated from the generated 2.5D images. It employs an encoding–decoding network in both phases. Finally, a reprojection consistency loss is used to confirm that the estimated 3D shape matches the generated 2.5D sketches. In this work, a multi-view and pose supervised technique is also obtained. This approach avoids modelling item appearance differences within the original image by generating 2.5D drawings from it [76]. Although 3D convolutional neural networks have been used by MarrNet [21] and GenRe [20] to achieve resolutions of up to 128^3, this has only been accomplished with shallow designs and tiny batch sizes, which causes training to go slowly [77]. Due to the global nature of employing image encoders for conditioning, these models exhibit weak generalisation capabilities and are limited by the range of 3D-data-gathering methods employed. Furthermore, in order to guarantee alignment between the predicted form and the input, they need an extra pose estimation phase [78]. This model uses ShapeNet for 3D annotation, which contains objects of basic shapes [79]. Also, it relies on 3D supervision, which is only available for restricted classes or in a synthetic setting [80]. A complete overview is illustrated in Figure 8.

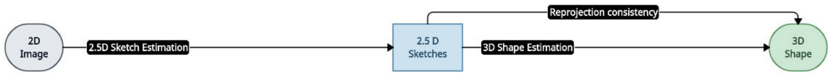


Figure 8. Network architecture of MarrNet [21] model.

Perspective Transformer Nets [23]: This method introduces a novel projection loss for learning 2D observations in the absence of 3D ground truths. To reconstruct 3D voxels, the authors employ a 2D convolutional encoder, a 3D up-convolutional decoder, and a perspective transformer network. They reached cutting-edge performance at the time. When rendering a pixel, all of the voxels along a ray that project to that pixel are considered. The final pixel colour can be selected with this model. When displaying voxels, the gradient problem brought on by primitive shape displacement does not arise since a voxel's location is fixed in three dimensions. Using camera settings, this model projects the voxels from the world space to the screen space and performs more computationally efficient bilinear sampling. Using this strategy, every pixel has an occupancy probability assigned to it. Casting a ray from the pixel, sampling each corresponding voxel, and selecting the one with the highest occupancy probability yields this result [63]. In addition to mainly focusing on inferring depth maps as the scene geometry output, this method has also shown success in learning 3D volumetric representations from 2D observations based on principles of projective geometry [81]. This method requires object masks [82]. Because the underlying 3D scene structure cannot be utilised, this 2D generative model only learns to parameterise the manifold of 2D natural pictures. It struggles to produce images that are consistent across several views [83]. The complete network architecture is illustrated in Figure 9.

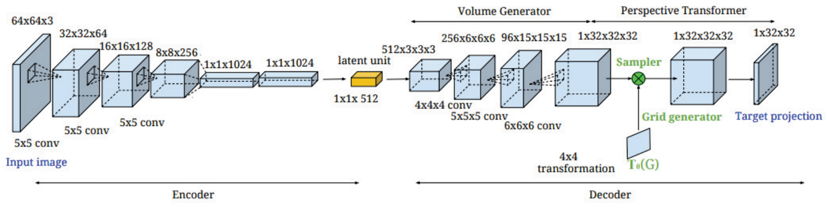


Figure 9. Network architecture of Perspective Transformer Nets [23] model.

Rethinking reprojection [24]: The authors of this model, in contrast to the previous research, reconstruct pose-aware 3D shapes from a single natural image. This model uses a well-known, highly accurate, and resilient approach called reprojection error minimisation for shape reconstruction. It demonstrates how well the genuine projection on the image is recreated by an approximated 3D world point [84]. This approach trains shape regressors by comparing projections of ground truths and predicted shapes [85]. Usually, images containing one or a few conspicuous, distinct items are used to test this strategy [86]. The network reconstructs the 3D shape in a canonical posture from the 2D input. The posture parameters are estimated concurrently by a pose regressor and subsequently applied to the rebuilt canonical shape. Decoupling shape and posture lowers the number of free parameters in the network, increasing efficiency [87]. In the absence of 3D labels, this model uses additional 2D reprojection losses to highlight the border voxels for rigid objects [88]. Most of the time, this approach assumes that the scene or object to be registered is either non-deformable or generally static [89]. This representation is limited in terms of resolution [90]. Figure 10 shows the proposed methods of p-TL and p-3D-VAE-GAN models.

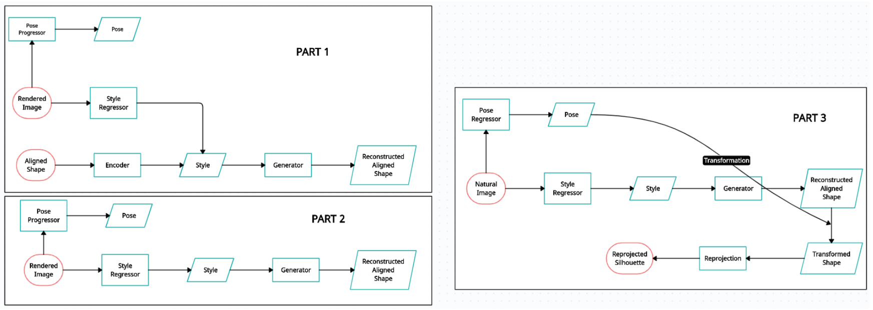


Figure 10. Proposed methods for reconstructing pose-aware 3D voxelised shapes: p-TL (parts 1 and 3) and p-3D-VAE-GAN (parts 2 and 3) [24] models.

3D-GAN [27]: The authors of this model present an unsupervised framework that combines adversarial and volumetric convolutional networks to produce voxels from a probabilistic latent space. They enhance the network's generalisation capacity. Using volumetric convolutions, the developers of this model demonstrated GANs that could create three-dimensional (3D) data samples. They created new items such as vehicles, tables, and chairs. They also demonstrated how to convert two-dimensional (2D) images into three-dimensional (3D) representations of the objects shown in those images [91]. Using this model, visual object networks [92] and PrGANs [93] generate a voxelised 3D shape first, which is then projected into 2D to learn how to synthesise 2D pictures [94]. This approach's generative component aims to map a latent space to a distribution of intricate 3D shapes. The authors train a voxel-based neural network (GAN) to produce objects. The drawback is that GAN training is notoriously unreliable [95]. Figure 11 shows the generator in the 3D-GAN model mirrored by the discriminator.

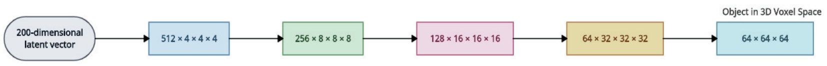


Figure 11. The generator in 3D-GAN [27] model.

Methods to generate voxels frequently do not provide texture or geometric features, and the generating process at high resolution is hampered by the 3D convolution's large memory footprint and computational complexity [96]. Nevertheless, point cloud and voxel-based models are frequently predictable and only provide a single 3D output [97]. Although point clouds and voxels are more compatible with deep learning architectures, they are not amenable to differentiable rendering or suffer from memory inefficiency problems [98].

4.3.3. Mesh Representation

Neural renderer [35]: Building differentiable rendering pipelines is the goal of a new discipline called neural rendering, which is making quick strides towards producing controlled, aesthetically realistic rendering [99]. The authors of this model use an integrated mesh rendering network to reconstruct meshes from low-resolution images. They minimise the difference between reconstructed objects and their respective ground truths on 2D silhouettes. The authors suggest a renderer called neural 3D mesh renderer (NMR) and bring up two problems with a differentiable renderer called OpenDR [100]. The gradient computation's locality is the first problem. Only gradients on border pixels can flow towards vertices due to OpenDR's local differential filtering; gradients at other pixels are not usable. This characteristic might lead to subpar local minima in optimisation. The derivative's failure to make use of the target application's loss gradient—such as picture reconstruction—is the second problem. One technique employed for evaluation involves visualising gradients (without revealing ground truth) and assessing the convergence effectiveness of those gradients throughout the optimisation of the objective function [63]. In the forward pass, NMR carries out conventional rasterisation, and in the backward pass, it computes estimated gradients [101]. For every object instance, the renderings and splits derived from this model offer 24 fixed elevation views with a resolution of 64 × 64 [82]. The objects are trained in canonical pose [72]. This mesh renderer modifies geometry and colour in response to a target image [102]. Figure 12 shows the single-image 3D reconstruction.

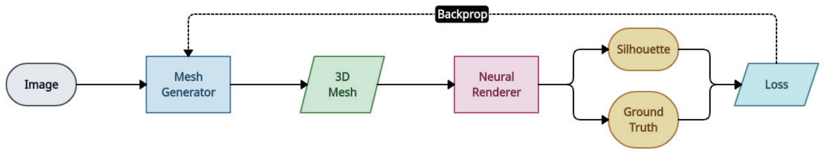


Figure 12. Pipeline for single-image 3D reconstruction [35].

Residual MeshNet [36]: To reconstruct 3D meshes from a single image, the authors present this model, a multilayered framework composed of several multilayer perceptron (MLP) blocks. To maintain geometrical coherence, they use a shortcut connection between two blocks. The authors of this model suggest reconstructing 3D meshes using MLPs in a cascaded hierarchical fashion. Three blocks of stacked MLPs are used for hierarchical mesh deformation in the suggested design, along with a ResNet-18 image encoder for feature extraction. To conduct the fundamental shape deformation, the first block, which has one MLP, is supplied with the coordinates of a 2D mesh primitive and image features. The next blocks include many stacked MLPs that concurrently alter the mesh that was previously deformed [103]. The trained model was built on a chamfer distance (CD)-based goal, which promotes consistency between the generated meshes and the ground truth meshes [67]. This work, however, has challenges in reconstructing smooth results with proper triangulation. The majority of mesh learning techniques aim to achieve a desired shape by deforming a template mesh using the learned shape beforehand, since altering the mesh topology is difficult. This model uses progressive deformation and residual prediction, which adds additional details while reducing learning complexity. Despite having no complicated structure, it results in significant patch overlaps and holes [104]. This model is used to produce meshes automatically during the finite element method (FEM) computation process. Although this does not save time, it increases computing productivity [105]. Figure 13 shows the network structure of Residual MeshNet.

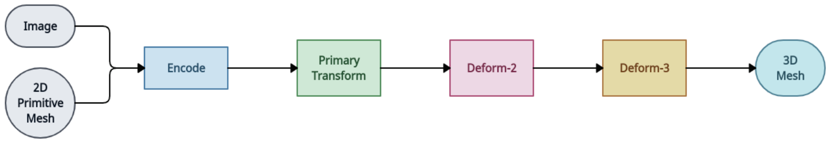


Figure 13. Main network structure of Residual MeshNet [36].

Pixel2Mesh [37]: This model reconstructs 3D meshes of hard objects using a cascaded, graph-based convolutional network to obtain greater realism. The network extracts perceptual features from the input image and gradually deforms an ellipsoid in order to obtain the output geometry. The complete model has three consecutive mesh deformation blocks. Each block enhances mesh resolution and estimates vertex positions, which are later used to extract perceptual image features for the following block. However, several perspectives of the target object or scene must be included in the training data for 3D shape reconstruction, which is seldom the case in real-world scenarios [99]. Figure 14 shows an overview of the Pix2Mesh framework.

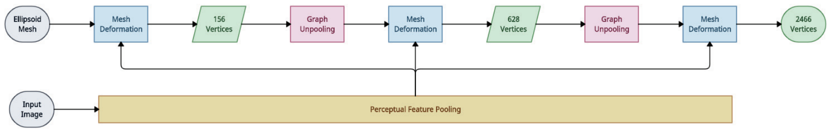


Figure 14. Cascaded mesh deformation network [37].

Other research, in addition to the above, proposes reconstructing inherent deformations in non-rigid objects. Non-rigid reconstruction tasks from a single image typically require additional information about the target objects, which can be predicted during the process or provided as prior knowledge, such as core structures and parameterised models.

CoReNet [38]: This model is a coherent reconstruction network that collaboratively reconstructs numerous objects from a single image for multiple object reconstruction. The authors of this model suggest three enhancements by building on popular encoder–decoder designs for this task: (1) a hybrid 3D volume representation that facilitates the construction of translation equivariant models while encoding fine object details without requiring an excessive memory footprint; (2) ray-traced skip connections that propagate local 2D information to the output 3D volume in a physically correct manner; and (3) a reconstruction loss customised to capture overall object geometry. All objects detected in the input image are represented in a single, consistent 3D coordinate without intersection after passing through a 2D encoder and a 3D decoder. To assure physical accuracy, a ray-traced skip connection is introduced. CoReNet uses a voxel grid with offsets for the reconstruction of scenes with many objects; however, it needs 3D supervision for object placement and identification [82]. Instead of using explicit object recognition, CoReNet used a physical-based ray-traced skip link between the picture and the 3D volume to extract 2D information. Using a single RGB picture, the method reconstructs the shape and semantic class of many objects directly in a 3D volumetric grid [106]. As of late, CoReNet has been able to rebuild many objects on a fixed grid of 128^3 voxels while preserving 3D position data in the global space. Additionally, training on synthetic representations restricts their practicality in real-world situations [107]. Figure 15 shows the pipeline of 3D reconstruction using this model.

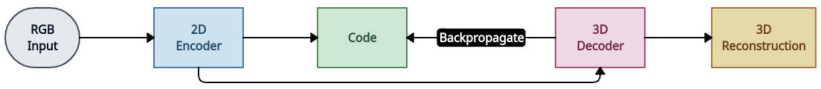


Figure 15. Pipeline of 3D reconstruction using CoReNet [38].

Table 6 provides the advantages and limitations of single-view 3D reconstruction models reviewed in this study. In brief, these approaches show the potential of deep learning for 3D object reconstruction using mesh representation. Nevertheless, most of these methods do not have the ability to dynamically change the template mesh's topology [108]. The majority of these mesh-based techniques do not involve postprocessing, but they frequently call for a deformable template mesh made up of many three-dimensional patches, which results in non-watertight meshes and self-intersections [72].

Table 6. Advantages and limitations of single-view 3D reconstruction models.

Model	Advantages	Limitations
PointOutNet [9]	Introduces the chamfer distance loss, which is invariant to the permutation of points and is adopted by many other models as a regulariser.	Utilises less memory, but since they lack connection information, they need extensive postprocessing.
Pseudo-renderer [12]	Uses 2D supervision in addition to 3D supervision to obtain multiple projection images from various viewpoints of the generated 3D shape for optimisation.	Predicts denser, more accurate point clouds but is limited to the amount of points that point cloud-based representations can accommodate.
RealPoint3D [13]	Attempts to recreate 3D models from nature photographs with complicated backgrounds.	Needs an encoder to extract the input image's 2D features and input point cloud data's 3D features.
Cycle-consistency-based approach [15]	Uses a differentiable renderer to infer a 3D shape without using ground truth 3D annotation.	Cycle consistency produces deformed body structure or out-of-view images if it is unaware of the previous distribution of the 3D features, which interferes with the training process.
GenRe [20]	Can rebuild 3D objects with resolutions of up to 128 × 128 × 128 and more detailed reconstruction outcomes.	Higher resolutions have been used by this model at the expense of sluggish training or lossy 2D projections, as well as small training batches.
MarrNet [21]	Avoids modelling item appearance differences within the original image by generating 2.5D drawings from it.	Relies on 3D supervision which is only available for restricted classes or in a synthetic setting.
Perspective Transformer Nets [23]	Learns 3D volumetric representations from 2D observations based on principles of projective geometry.	Struggles to produce images that are consistent across several views as the underlying 3D scene structure cannot be utilised.
Rethinking reprojection [24]	Decoupling shape and posture lowers the number of free parameters in the network, increasing efficiency.	Assumes that the scene or object to be registered is either non-deformable or generally static.
3D-GAN [27]	Generative component aims to map a latent space to a distribution of intricate 3D shapes.	GAN training is notoriously unreliable.
Neural renderer [35]	Objects are trained in canonical pose.	This mesh renderer modifies geometry and colour in response to a target image.
Residual MeshNet [36]	Reconstructing 3D meshes using MLPs in a cascaded hierarchical fashion.	Produces mesh automatically during the finite element method (FEM) computation process, although it does not save time increasing computing productivity.
Pixel2Mesh [37]	Extracts perceptual features from the input image and gradually deforms an ellipsoid in order to obtain the output geometry.	Several perspectives of the target object or scene are not included in the training data for 3D shape reconstruction, as in real-world scenarios.
CoReNet [38]	Reconstructs the shape and semantic class of many objects directly in a 3D volumetric grid using a single RGB image.	Training on synthetic representations restricts their practicality in real-world situations.

Numerous organised formats, such as voxel grids, point clouds, and meshes that display heterogeneity per element, are used to store 3D data. For instance, the topology and quantity of vertices and faces might vary throughout meshes. Because of this variability, it is challenging to apply batched operations on 3D data in an effective manner with the tensor-centric primitives offered by common deep learning toolkits such as PyTorch [101].

These studies do not address multi-object analysis, but they do provide intriguing solutions to their particular issues with single object pictures [109]. All that is needed for these tasks is single-view self-supervision. Even with this tremendous advancement, these techniques nonetheless have two main drawbacks: (1) ineffective bottom-up reasoning, in which the model is unable to capture minute geometric features like concavities; and (2) incorrect top-down reasoning, in which the model just explains the input perspective and is unable to precisely recreate the entire 3D object shape [110]. The drawback of this single-category technique is that data cannot be pooled across categories, which might be useful for tasks like viewpoint learning and generalisation to previously unknown categories of objects (zero-shot [111] or few-shot [112] learning) [113]. There are restrictions on the kinds of scenes that can be reconstructed using these methods, as they are designed to only use a single input view at test time [82]. Results from single-view 3D reconstruction are typically incomplete and inaccurate, particularly in cases where there are obstructions or obscured regions [114].

4.4. Multiple-View Reconstruction

The apparent uncertainty in the object is decreased and the number of occluded portions is increased when images taken from different angles are fed into the network. Traditionally, there have been two kinds of reconstruction from several perspectives. Rebuilding a static item from a number of images is the first step; reconstructing a moving object's three-dimensional structure from a movie or several frames is the second. In order to match up the incomplete 3D shapes into a full one, both of these algorithms use images to estimate the camera posture and matching shape. As a result, three-dimensional alignment and posture estimation are challenging. First, deep learning techniques are introduced into multi-image reconstruction to address this problem. Next, from the input images, 3D shapes are immediately generated by deep neural networks. Moreover, the rebuilding procedure takes a lot less time when end-to-end structures are used. Table 7 provides the list of multi-view 3D reconstruction models reviewed in this study.

Table 7. Multiple-view 3D reconstruction models reviewed in this study.

Nr.	Model	Dataset	Data Representation
1	3D34D [17]	ShapeNet [10]	Point Cloud
2	Unsupervised learning of 3D structure [18]	ShapeNet [10], MNIST3D [19]	Point Cloud
3	Pix2Vox++ [30]	ShapeNet [10], Pix3D [16], Things3D [30]	Voxels
4	3D-R2N2 [11]	ShapeNet [10], PASCAL3D+ [22], MVS CAD 3D [11]	Voxels
5	Weak recon [31]	ShapeNet [10], ObjectNet3D [14]	Voxels
6	Relative viewpoint estimation [32]	ShapeNet [10], Pix3D [16], Things3D [30]	Voxels

4.4.1. Point Cloud Representation

3D34D [17]: The authors of this model employ a UNet encoder, producing feature maps to produce geometry-aware point representations of object categories unseen during training. For 3D object reconstruction, this study employs multi-view images with ground truth camera postures and pixel-aligned feature representations. A stand-alone 3D reconstruction module that was trained using ground truth camera postures is used by this model [115]. This work has made generalisation a clear goal. The goal of this study is to obtain a more expressive intermediate shape representation by locally assigning features and 3D points [116]. This is an object-centred approach. This work was the first to examine the generalisation characteristics of shape reconstruction using previously unknown shape categories. This approach emphasises reconstruction from many perspectives, uses continuous occupancies, and evaluates generalisation to previously undiscovered categories [117]. The study focused on reconstruction from several perspectives and examined feature description bias for generalisation [118]. While this 3D reconstruction technique performs admirably on synthetic objects rendered with a clear background, it may not translate well to actual photos, novel categories, or more intricate object geometries [75]. According to this research, contemporary learning-based computer vision techniques are unable to generalise to data that is not distributed evenly [119].

Unsupervised learning of 3D structure from images [18]: The authors of this model train deep generative models of 3D objects in an end-to-end fashion and directly from 2D images without the use of 3D ground truth, and then reconstruct objects from 2D images via probabilistic inference. This purely unsupervised method is built on sequential generative models and can generate high-quality samples that represent the multi-modality of the data. With a primary focus on inferring depth maps as the scene geometry output, this study has demonstrated success in learning 3D volumetric representations from 2D observations using the concepts of projective geometry [81]. In [120], synthesised data are used. Ref. [121] explores the use of 3D representations as inductive bias in generative models. Using adversarial loss, the technique presented in [122] usually optimises 3D representations to provide realistic 2D images from all randomly sampled views. An effort based on policy gradient algorithms performs single-view 3D object reconstruction using the non-differentiable OpenGL renderer with this model. Nevertheless, only basic and coarse forms may be recreated in the collection [63]. Figure 16 shows the overall framework for this model.

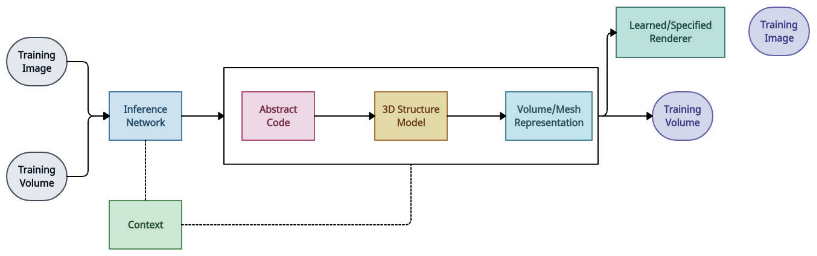


Figure 16. Proposed framework of unsupervised learning of 3D structure from images [18].

Overall, these techniques offer significant progress in the area of multi-view reconstruction, enabling the generation of 3D models from 2D data in a more accurate and efficient manner. There is still room for improvement, especially when it comes to better alignment accuracy and estimating camera poses. Further research and development in this area could lead to even more sophisticated techniques for generating 3D models from multiple images.

4.4.2. Voxel Representation

Pix2Vox++ [30]: The authors of this model listed three limitations for RNN-based methods. First, permutation variance prevents RNNs from reliably estimating the 3D geometry of an item when they are presented with the same collection of pictures in various sequences. Second, the input pictures cannot be properly used to improve reconstruction outcomes due to RNNs' long-term memory loss. Finally, as input pictures are analysed sequentially without parallelisation, RNN-based algorithms take a long time. To overcome these limitations, the authors proposed an encoder–decoder structure framework called Pix2Vox [123] based on RNNs. The authors introduced Pix2Vox++ [30] by making some improvements to the previously created Pix2Vox [123] model. In the Pix2Vox++ [30] network, the authors replaced the backbone of Pix2Vox [123], VGG, with ResNet. The authors of this model proposed Pix2Vox++ to generate a coarse volume for each input image. They fuse all of the coarse volumes using a multi-scale context-aware fusion module, followed by a refiner module to correct the fused volume. Primarily using synthetic data, such as from ShapeNet, this model learns to rebuild the volumetric representation of basic objects [124]. Pix2Vox++'s reconstruction findings are able to precisely recreate the general shape but are unable to provide fine-grained geometries [125]. Because of memory limitations, the model's cubic complexity in space results in coarse discretisations [126]. The visual information is transferred from the image encoder to the 3D decoder using only the feature channels (such as element-wise add, feature concatenation, and attention mechanism). The 3D decoder only receives implicit geometric information with limited semantic attributes, which serves as guidance for shape reconstruction. The decoder can quickly detect and recover such geometric information. On the contrary, the particular, detailed shape of these attributes will be determined by the detailed semantic attributes. However, throughout the reconstruction process, the decoder will seldom discover these semantic properties since they are intricately intertwined with one another in image features. The resolution for voxel data is often constrained due to the cubic growth of the input voxel data, and further raising the resolution would result in unacceptably high computing costs [127]. The accuracy of the method will become saturated when the number of input views exceeds a specific scale (e.g., 4), indicating the challenge of acquiring complementary information from a large number of independent CNN feature extraction units [128]. Figure 17 shows the proposed framework for this model.

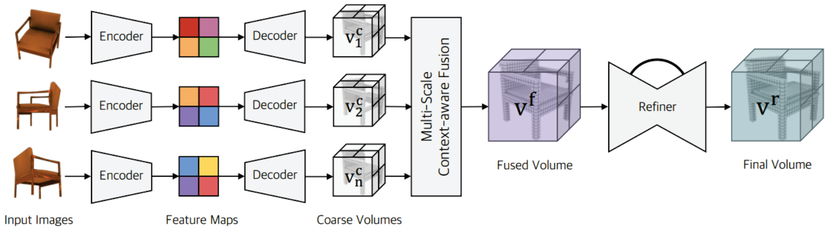


Figure 17. Proposed framework of Pix2Vox++ network [30].

3D-R2N2 [11]: Deeply influenced by the conventional LSTM framework, 3D-R2N2 generates 3D objects in occupancy grids with only bounding box supervision. In an encoder–LSTM–decoder structure, it merges single- and multi-view reconstruction. The 3D convolutional LSTM selectively updates hidden representations via input and forget gates. It successfully manages self-occlusion and refines the reconstruction result progressively as additional observations are collected. An overview of the network is presented in Figure 18. Despite the ability to preserve earlier observations, methods based on such structures may fail when presented with similar inputs and are restricted in their ability to retain features in early inputs. Using encoder–decoder architectures, this technique converts RGB image partial inputs into a latent vector, which is then used to predict the complete volumetric shape using previously learned priors. Fine shape features are lost in voxel-based methods, and since their normals are not smooth when produced, voxels look very different from high-fidelity shapes [95]. This CNN-based method only works with coarse $64 \times 64 \times 64$ grids [129]. This approach has significant memory use and computational overhead [61]. Since voxels are logical extensions of image pixels, cutting-edge methods for shape processing may be transferred from image processing. Nevertheless, low-resolution outcomes are typically produced because voxel representations are limited by GPU memory capacity [130].

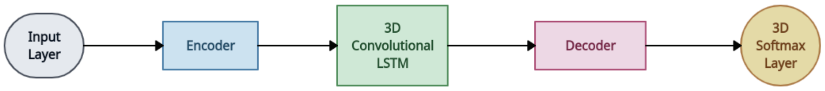


Figure 18. An overview of the 3D-R2N2 network [11].

Weak recon [31]: This method explores an alternative to costly 3D CAD annotation and proposes using lower-cost 2D supervision. Through a ray-trace pooling layer that permits perspective projection and backpropagation, the proposed method leverages foreground masks as weak supervision. By constraining the reconstruction to remain in the space of unlabelled real 3D shapes, this technique makes use of foreground masks for 3D reconstruction. Using ray-tracing pooling, this model learns shapes from multi-view silhouettes and applies a GAN to further limit the ill-posed issue [131]. This method is limited to low-resolution voxel grids [132]. The authors decided to employ GANs to represent 2D projections rather than 3D shapes when investigating adversarial nets for single-image 3D reconstruction. However, their reconstructions are hampered by this weakly supervised environment [133].

Relative viewpoint estimation [32]: The authors of this model propose teaching two networks to address alignment without 3D supervision: one to estimate the 3D shape of an object from two images of different viewpoints with corresponding pose vectors and predict the object's appearance from a third view; and the other to evaluate the misalignment of the two views. They predict a transformation that optimally matches the bottleneck features of two input images during testing. Their networks are also focused on generalising previously unseen objects. When estimating relative 3D poses among a group of little or non-overlapping RGB(-D) images, perspective variation is significantly more dramatic in regions where few co-visible regions are identified, making matching-based algorithms inappropriate. The authors of this model suggest using the hallucination-then-match paradigm to overcome this difficulty [134]. The authors point out that supplying an implicit canonical frame by using a reference image and formulating posture estimation as predicting the relative perspective from this view are the basic requirements to make zero-shot pose estimation a well-posed issue. Unfortunately, this technique does not extend to the category level; it can only predict posture for instances of a single item [135]. Figure 19 shows an overview of the shape-learning approach of this model.

Table 8 provides the advantages and limitations of multi-view 3D reconstruction models reviewed in this study. Point clouds, voxel grids, and mesh scene representations, on the other hand, are discrete, restricting the amount of spatial resolution that can be achieved, meaning they only sample the smooth surfaces underneath a scene sparingly, and they frequently require explicit 3D supervision [83].

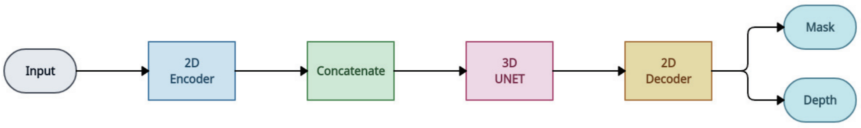


Figure 19. An overview of the shape-learning approach [32].

Table 8. Advantages and limitations of multi-view 3D reconstruction models.

Model	Advantages	Limitations
3D34D [17]	Obtains a more expressive intermediate shape representation by locally assigning features and 3D points.	Performs admirably on synthetic objects rendered with a clear background, but not on actual photos, novel categories, or more intricate object geometries.
Unsupervised learning of 3D structures [18]	Optimises 3D representations to provide realistic 2D images from all randomly sampled views.	Only basic and coarse shapes can be reconstructed.
Pix2Vox++ [30]	Generates a coarse volume for each input image.	Because of memory limitations, the model's cubic complexity in space results in coarse discretisations.
3D-R2N2 [11]	Converts RGB image partial inputs into a latent vector, which is then used to predict the complete volumetric shape using previously learned priors.	Only works with coarse 64 × 64 × 64 grids.
Weak recon [31]	Alternative to costly 3D CAD annotation, and proposes using lower-cost 2D supervision.	Reconstructions are hampered by this weakly supervised environment.
Relative viewpoint estimation [32]	Predicts a transformation that optimally matches the bottleneck features of two input images during testing.	It can only predict posture for instances of a single item and does not extend to the category level.

5. Registration

Determining the correlation between point cloud data of the same image acquired from several methods might be useful in some scenarios. By calculating the transformation for the optimal rotation and translation across the point cloud sets, 3D point cloud registration algorithms reliably align different overlapping 3D point cloud data views into a full model (in a rigid sense). The distance in a suitable metric space between the overlapping regions of two distinct point cloud sets is small in an ideal solution. This is difficult since noise, outliers, and non-rigid spatial transformations all interfere with the process. Finding the optimal solution becomes significantly more difficult when there is no information about the starting posture of various point cloud sets in space or the places where the sets overlap. Table 9 provides the list of 3D registration models reviewed in this study.

Table 9. 3D registration models reviewed in this study.

Nr.	Model	Dataset	Data Representation
1	CPD [136]	Stanford Bunny [33]	Meshes
2	PSR-SDP [137]	TUM RGB-D [138]	Point Cloud
3	RPM-Net [139]	ModelNet [28]	Meshes
4	DeepICP [140]	KITTI [141], SouthBay [142]	Point Cloud, Voxels
5	3D-SmoothNet [143]	3DMatch [144]	Point Cloud, Voxels
6	3D multi-view registration [145]	3DMatch [144], Redwood [146], ScanNet [39]	Point Cloud

5.1. Traditional Methods

Traditional 3D registration methods can be classified based on whether the underlying optimisation method used is global or local. The most well-known works in the global category are based on global stochastic optimisation using genetic algorithms or evolutionary algorithms. However, their main drawback is the computation time. On the other hand, the majority of studies performed in 3D registration nevertheless have local optimisation methods.

CPD [136]: The Coherent Point Drift (CPD) algorithm considers the alignment as a probability density estimation problem where one point cloud set represents the Gaussian mixture model centroids and the other represents the data points. The transformation is estimated by maximising the probability of fitting the centroids to the second set of points. The movement is forced to move coherently as a group to preserve the topological structure. The authors introduced this approach, which uses the methodology for maximum likelihood parameter estimation and establishes a probabilistic framework based on Gaussian mixture models (GMMs) [147]. Registration was reformulated by the authors as a probability density estimation issue. The first set of points served as the centroids of the GMMs that were fitted using likelihood maximisation to the data or points from the second set. To ensure that the centroids moved coherently, extra effort was taken [148]. While GMM-based methods might increase resilience against outliers and bad initialisations, local search remains the foundation of optimisation [149].

PSR-SDP [137]: The authors of this model studied the registration of point cloud sets in a global coordinate system. In other words, with the original set of n points, we want to find the correspondences between (subsets of) the original set and m local coordinate systems, respectively. The authors consider the problem as a semi-definite program (SDP) within the application of Lagrangian duality, and this allows for verifying the global optimality of a local minimiser in a significantly faster manner. The registration of numerous point sets is solved by this approach using semi-definite relaxation. By using a convex SDP relaxation, the non-convex constraint is relaxed [150]. Lagrangian duality and SDP relaxations were used to tackle the multiple point cloud registration problem. This problem was investigated further in this model, where it was demonstrated that the SDP relaxation is always tight under low-noise regimes [151]. A study of global optimality requirements for point set registration (PSR) with incomplete data was presented using this approach. This approach used Lagrangian duality to provide a primal problem candidate solution, allowing it to retrieve the associated dual variable in closed form. This approach provides poor estimates even in the presence of a single outlier because it assumes that all measurements are inliers (i.e., have little noise), a situation that rarely occurs in practice [152].

RPM-Net [139]: RPM-Net inherits the idea of the RPM algorithm, introduces deep learning to desensitise the initialisation, and improves network convergence with learned fusion features. In this method, the initialisation assignments are based on the fusion of hybrid features from a network instead of spatial distances between points. The optimal annealing parameters are predicted by a secondary network, and a modified chamfer distance is introduced to evaluate the quality of registration. This method outperforms previous methods and handles missing keypoints and point cloud sets with partial visibility. RPM-Net presents a deep-learning-based method for rigid point cloud registration that is more resilient and less susceptible to initialisation. The network created by this approach is able to solve the partial visibility of the point cloud and obtain a soft assignment of point correspondences [150]. This model's feature extraction is geared particularly towards artificial, object-centric point clouds [153]. By leveraging soft correspondences that are calculated from the local feature similarity scores to estimate alignment, this approach avoids the non-differentiable nearest-neighbour matching and RANSAC processes. RPM-Net also makes use of surface normal data [154]. Because of matches that are heavily tainted by outliers, this model's resilience and applicability in complicated scenarios does not always live up to expectations [155]. This approach looks for deep features to find correspondences; however, the features that are taken out of point clouds have a low capacity to discriminate, which results in a high percentage of false correspondences and severely reduces the accuracy of registration. In order to establish soft correspondences from local characteristics, which might boost resilience but reduce registration accuracy, RPM-Net suggests a network that predicts the ideal annealing parameters [156]. Figure 20 shows the network architecture of this model.

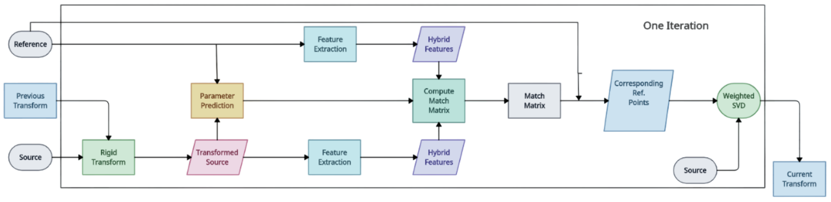


Figure 20. An overview of the RPM-Net network [139].

5.2. Learning-Based Methods

DeepICP [140]: This is an early end-to-end framework achieving comparable registration accuracy to the state-of-the-art traditional methods for point cloud registration. The algorithm utilises PointNet++ [157] to extract local features, followed by a point-weighting layer that helps select a set of keypoints. Once a set of candidate keypoints is selected from the target point cloud set, they pass through a deep-feature-embedding operation together with the keypoints of the source set. Finally, a corresponding point generation layer takes the embeddings and generates the final result. Two losses are incurred: (1) the Euclidean distance between the estimated corresponding points and the ground truth under the ground truth transformation, and (2) the distance between the target under the estimated transformation and the ground truth. These losses are combined to consider both global geometric information and local similarity. By creating a connection using the point cloud's learned attributes, this study improved the conventional ICP algorithm using the neural network technique. This method takes a large amount of training time on the dataset, despite its good performance. If the test data change significantly from the training data, the algorithm's output will not be optimal. Consequently, there are stringent data limits with the neural-network-based enhanced ICP technique [158]. A solution to the point cloud registration problem has been offered [159]. Rather than utilising ICP techniques, this approach might directly match the local and target point clouds in addition to extracting descriptors via neural networks [160]. It still takes a lot of computing effort to combine deep learning with ICP directly [150]. The architecture of the proposed end-to-end learning network for 3D point cloud registration is demonstrated in Figure 21.

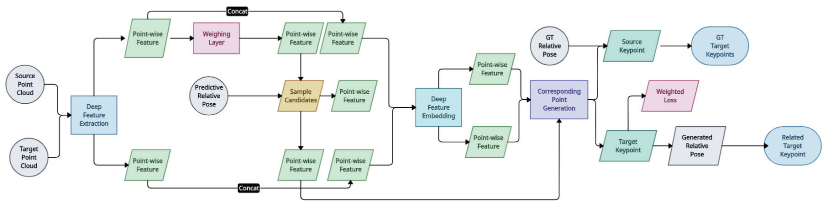


Figure 21. The architecture of DeepICP [140].

3DSmoothNet [143]: 3DSmoothNet matches two point cloud sets with a compactly learned 3D point cloud descriptor. At first, the model computes the local reference frame of the area near the randomly sampled keypoints. This is followed by the near areas being transformed into voxelised smoothed density value representations [161]. Then, the local feature of each keypoint is generated by 3DSmoothNet. The features extracted by this cloud descriptor will be utilised by a RANSAC approach for producing registration results. The proposed 3D point cloud descriptor outperforms traditional binary-occupancy grids, and it is the first learned, universal matching method that allows transferring trained models between modalities. For feature learning, this approach suggests a rotation-invariant handcrafted feature that is fed into a deep neural network. Deep learning is used as a feature extraction technique in all these strategies. Their goal is to estimate robust correspondences by learning distinguishing characteristics through the development of complex network topologies or loss functions. This experiment demonstrates that while applying deep learning directly will not ensure correctness, applying mathematical theories of registration directly will require enormous amounts of computing effort [150]. This approach is designed to mitigate voxelisation and noise artefacts. The receptive field is limited to a predetermined size, and the computational cost is significantly increased by this early work's outstanding performance, which is still based on individual local patches [153]. Fully convolutional geometric features (FCGFs) is the fastest feature extraction method and is 290 times faster than 3DSmoothNet [162].

3D multi-view registration [145]: Following 3DSmoothNet, the authors proposed a method that formulates conventional two-stage approaches (typically an initial pairwise alignment followed by a global refinement) in an end-to-end learnable convention by directly learning and registering all views in a globally consistent fashion. Their work improves a point cloud descriptor studied in [162], using a soft correspondence layer that pairs different sets to compute primary matches. These matches are then fed to a pairwise registration block to obtain transformation parameters and corresponding weights. Finally, these weights and parameters are globally refined by a novel iterative transformation synchronisation layer. This work is the first end-to-end algorithm for joint learning of both stages of the registration problem. This model outperforms previous two-stage algorithms with higher accuracy and less computational complexity. This method utilises FCGF [162] to solve the multi-way registration problem [163]. The primary use for this technique is indoor point clouds [164]. Figure 22 shows the proposed pipeline for this method.

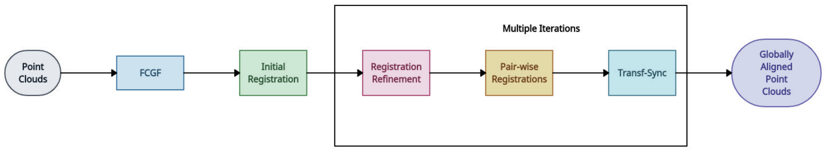


Figure 22. Proposed pipeline for 3D multi-view registration [145].

Table 10 provides the advantages and limitations of 3D registration models reviewed in this study. This category offers the following two benefits: (1) A point feature based on deep learning may offer reliable and precise correspondence searches. (2) By applying a straightforward RANSAC approach, the correct correspondences might result in accurate registration outcomes. Nevertheless, there are limitations to these kinds of methods: (1) A lot of training data are required. (2) If there is a significant distribution discrepancy between the unknown scenes and the training data, the registration performance in such scenes drastically decreases. (3) To learn a stand-alone feature extraction network, they employ a different training procedure. In addition to registration, the learned feature network is used to determine point-to-point matching [150].

Table 10. Advantages and limitations of 3D registration models.

Model	Advantages	Limitations
CPD [136]	Considers the alignment as a probability density estimation problem, where one point cloud set represents the Gaussian mixture model centroids, and the other represents the data points.	While GMM-based methods might increase resilience against outliers and bad initialisations, local search remains the foundation of the optimisation.
PSR-SDP [137]	Allows for verifying the global optimality of a local minimiser in a significantly faster manner.	Provides poor estimates even in the presence of a single outlier because it assumes that all measurements are inliers.
RPM-Net [139]	Able to solve the partial visibility of the point cloud and obtain a soft assignment of point correspondences.	Computational efficacy increases as the number of points in the point clouds increases.

Table 10. *Cont.*

Model	Advantages	Limitations
DeepICP [140]	By creating a connection using the point cloud's learned attributes, this study improved the conventional ICP algorithm using the neural network technique.	Takes a lot of computing effort to combine deep learning with ICP directly.
3DSmoothNet [143]	First learned, universal matching method that allows transferring trained models between modalities.	290 times slower than FCGF [162] model.
3D multi-view registration [145]	First end-to-end algorithm for joint learning of both stages of the registration problem.	A lot of training data are required.

6. Augmentation

The proliferation of 3D data collection equipment and the rising availability of 3D point cloud data are the result of recent advancements in 3D sensing technology. Despite the fact that 3D point clouds offer extensive information on the entire geometry of 3D objects, they are frequently severely flawed by outliers, noise, and missing points. Many strategies, including outlier removal, point cloud completion, and noise reduction, have been proposed to solve these problems; however, the implementation and application differ. While point cloud completion techniques try to fill in the missing portions of the point cloud to provide a comprehensive representation of the object, outlier removal strategies try to detect and eliminate points that do not adhere to the overall shape of the object. On the other hand, noise suppression approaches work to lessen the impact of random noise in the data in order to enhance the point cloud's quality and accuracy. Table 11 provides the list of 3D augmentation models reviewed in this study.

Table 11. 3D augmentation models reviewed in this study.

Nr.	Model	Dataset	Data Representation
1	MaskNet [165]	S3DIS [3], 3DMatch [144], ModelNet [28]	Point Cloud
2	GPDNet [166]	ShapeNet [10]	Point Cloud
3	DMR [167]	ModelNet [28]	Point Cloud
4	PU-Net [168]	ModelNet [28], ShapeNet [10]	Point Cloud
5	MPU [169]	ModelNet [28], MNIST-CP [19]	Point Cloud
6	CP-Net [170]	ModelNet [28]	Point Cloud
7	SampleNet [171]	ModelNet [28], ShapeNet [10]	Point Cloud

6.1. Denoising

While better data gathering methods may result in higher-quality data, noise in point clouds is unavoidable in some circumstances, such as outdoor scenes. A number of denoising methods have been put forward to stop noise from affecting point cloud encoding. Local surface fitting (e.g., jets or MLS surfaces), local or non-local averaging, and statistical presumptions on the underlying noise model are examples of early conventional approaches. Since then, learning-based techniques have been put forward that, in the majority of situations, perform better than traditional solutions.

MaskNet [165]: The authors of this model presented MaskNet for determining outlier points in point clouds by computing a mask. The method can be used to reject noise in even partial clouds in a rather computationally inexpensive manner. This approach, which uses learning-based techniques to estimate global descriptors of each point in the point cloud in addition to a global feature of the point cloud, was presented to address the sparse overlap of point clouds. After that, a predicted inlier mask is used to compute the transformation using these features. This model's ability to effectively tackle the partial-to-partial registration problem is one of its key advantages. However, this model's primary drawback is that it requires the input of both a partial and complete point cloud [172]. It requires a point cloud without outliers as a template. Voxelisation or projection are required to convert the initial point clouds into structured data because of the chaos of point clouds. Due to the inevitable rise in computing load and loss in geographical information in certain categories, this process results in issues with significant time consumption and inaccuracy [173]. The feature interaction module of MaskNet is meant to take two point clouds as input and output the posterior probability [174]. To anticipate whether points in template point clouds coincide with those in source point clouds, it makes use of a PointNet-like network. But only in the template point cloud can it identify the overlapping points [175]. One typical issue with raw-point-based algorithms is that they assume a considerable overlap or good starting connections between the provided pair of point sets [176]. MaskNet is not easily transferred to other tasks or real-world situations due to its high sensitivity to noise [177]. According to this method, the extracted overlapping points are assumed to be entirely correct, and they are thought to have equivalent points. However, the accuracy of the overlapping spots that the network estimates cannot be guaranteed [178]. Figure 23 shows the architecture of this model.

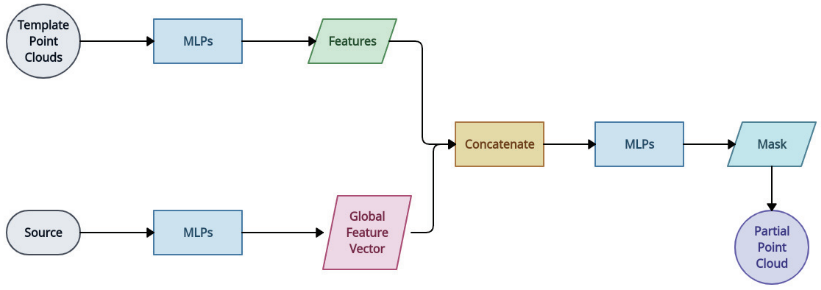


Figure 23. Architecture of MaskNet [165].

However, all of the aforesaid deep learning approaches are fully supervised and require pairs of clean and noisy point clouds.

GPDNet [166]: The authors of this model proposed a new graph convolutional neural network targeted at point cloud denoising. The algorithm deals with the permutation-invariance problem and builds hierarchies of local or non-local features to effectively address the denoising problem. This method is robust to high levels of noise and also has structured noise distributions. In order to regularise the underlying noise in the input point cloud, GPDNet suggests creating hierarchies of local and non-local features [179]. Edge-conditioned convolution (ECC) [180] was further expanded to 3D denoising problems using this approach [181]. The two primary artefacts that affect this class of algorithms are shrinkage and outliers, which result from either an overestimation or an underestimation of the displacement [182]. The point clouds' geometric characteristics are often oversmoothed using GPDNet [183].

DMR [167]: The authors of this model presented a novel method to use differentiably subsampled points for learning the underlying manifold of a noisy point cloud. The proposed algorithm is different from the aforementioned methods as it resembles more of a human-like cleaning of a noisy point cloud using multi-scale geometric feature information as well as supervision from ground truths. This network can also be trained in an unsupervised manner. A simple implementation of the graph convolutional network (GCN) is unstable as the denoising process mostly deals with local representations of point neighbourhoods. In order to learn the underlying manifold of the noisy input from differentiably subsampled points and their local features with minimal disruption, DMR relies on dynamic graph CNN (DGCNN) [184] to handle this problem [179]. In this model, the patch manifold reconstruction (PMR) upsampling technique is straightforward and efficient [185]. This method's downsampling step invariably results in detail loss, especially at low noise levels, and it could also oversmooth by removing some useful information [182]. The goal of these techniques is to automatically and directly learn latent representations for denoising from the noisy point cloud. Its overall performance on noise in the actual world is still restricted though [186]. Figure 24 shows the architecture of this model.

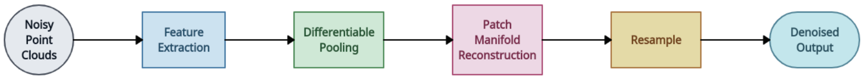


Figure 24. Illustration of the proposed DMR network [167].

6.2. Upsampling

In 3D point cloud processing, upsampling is a typical challenge when the objective is to produce a denser set of points that faithfully depicts the underlying geometry. Though the uneven structure and lack of spatial order of point clouds present extra obstacles, the problem is analogous to the image super-resolution problem. Points had to be adjusted in the early, traditional point cloud upsampling techniques, which were optimisation-based. Although these approaches frequently yielded satisfactory results, their application was limited since they assumed smooth underlying geometry. Recently, data-driven approaches have emerged for point cloud upsampling, which have demonstrated significant improvements over traditional methods.

PU-Net [168]: PU-Net is one such approach that uses a multi-branch convolutional unit to expand the set of points in a point cloud by learning multi-level features for each point. During the end-to-end training of PU-Net, both reconstruction loss and repulsion loss are jointly utilised to improve the quality of the output. PU-Net learns the representation from the raw point dataset using unsupervised methods. This model learns sparse and irregular point clouds. Each point's multi-level features are learned, and the enlarged feature is obtained by applying multi-branch convolution. This feature is then divided to rebuild the point cloud. PU-Net consists of four components: patch extraction, which gathers point clouds of different sizes; point feature embedding, which extracts the point clouds' local and global geometric information; feature expansion, which increases the number of features; and coordinate reconstruction, which implements the expanded features' 3D coordinates [187]. It is a LiDAR-based technique that uses raw LiDAR scans to learn high level point-wise information. From each high-dimensional feature vector, many upsampled point clouds are then reconstructed [188]. Recovering the 3D shape of objects that have only partially been observed can be achieved to a limited extent by upsampling point clouds. Moreover, there would be a noticeable increase in latency if the whole point cloud was upsampled. A low-density point cloud can be converted to a high-density one via point cloud upsampling. Nevertheless, during training, they require high-density point cloud ground truths [189]. This model only learns spatial relationships at a single level of multi-step point cloud decoding via self-attention [190]. With exceptionally sparse and non-uniform low-quality inputs, this network might not generate findings that are believable. PU-Net replicates the point features and processes each copy independently using different MLPs in order to upsample a point collection. But the enhanced features would be too close to the inputs, degrading the quality of the upsampling [191]. The detailed architecture of PU-Net is presented in Figure 25.

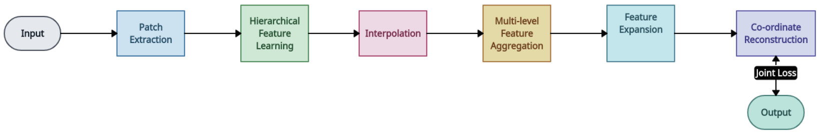

Input
Patch Extraction
Hierarchical Feature Learning
Interpolation
Multi-level Feature Aggregation
Feature Expansion
Co-ordinate Reconstruction
Joint Loss
Output

Figure 25. Architecture of PU-Net [168].

MPU [169]: The authors of this model proposed an adaptive patch-based point cloud upsampling network that was inspired by recent neural image super-resolution methods. This network is trained end-to-end on high-resolution point clouds and emphasises a certain level of detail by altering the spatial span of the receptive field in various steps. It is a LiDAR-based technique that uses raw LiDAR scans to learn high-level point-wise information. From each high-dimensional feature vector, it then reconstructs many upsampled point clouds [188]. Recovering the 3D shape of objects that have only partially been observed can be achieved to a limited extent by upsampling point clouds. Moreover, there would be a noticeable increase in latency if the whole point cloud was upsampled. A low-density point cloud can be converted to a high-density one via point cloud upsampling. Nevertheless, during training, they require high-density point cloud ground truths [189]. This model only learns spatial relationships at a single level of multi-step point cloud decoding via self-attention [190]. With exceptionally sparse and non-uniform low-quality inputs, this network might not generate findings that are believable. A multi-step progressive upsampling (MPU) network was provided by the authors in order to reduce noise and preserve information. This approach divides a 16× upsampling network into four consecutive 2× upsampling subnets to upsample a point set incrementally in numerous phases. The training procedure is intricate and needs more subgroups for a greater upsampling rate, even if details are better maintained in the upsampled output [191]. The amount of computing memory used during training is higher. More significantly, this approach cannot be used for completion tasks and is restricted to upsampling sparse locations [192]. Figure 26 shows an overview of the MPU network with three levels of detail.

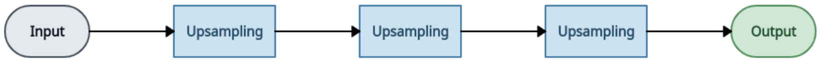


Figure 26. Overview of MPU with 3 levels of detail [169].

Even with the recent advancements, point cloud upsampling still faces difficulties, particularly when managing intricate structures with a range of densities and imperfections. Another problem is that the quality of the input has a significant impact on the quality of the point clouds that are created. More investigation is required to create point cloud upsampling algorithms that are more effective and efficient in order to overcome these obstacles.

6.3. Downsampling

In practical settings, the point cloud often contains a large number of points due to the use of high-density data acquisition sensors. While some applications benefit from this density, increased computation and low efficiency are common issues. One conventional approach is downsampling the point cloud using a neural network.

CP-Net [170]: The authors of this model propose a critical points layer (CPL) that downsamples the points adaptively based on learned features. The following network layer receives the points with the most active features from this critical points layer [193]. CPL globally filters out unimportant points while preserving the important ones. The proposed CPL layer can be used with a graph-based point cloud convolution layer to form CP-Net. When using this approach, the final representations typically retain crucial points that take up a significant amount of channels [194]. This graph-based downsampling approach uses K-nearest neighbours (K-NNs) to locate neighbouring points, in contrast to the majority of graph-based point cloud downsampling techniques. In addition, the global downsampling technique known as the critical points layer (CPL) has very high computing efficiency. A graph-based layer and the suggested layer can be used to create a convolutional neural network [195]. It is not possible to describe the underlying geometric structures of points or to properly capture the non-local dispersed contextual correlations in geographical locations and semantic information using this point-based approach that has recently been presented, which needs complex network designs to aggregate local features [196,197], as shown in Figure 27.

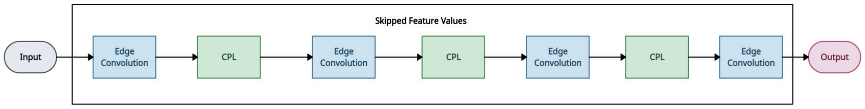


Figure 27. General overview of CP-Net [170].

SampleNet [171]: SampleNet is a differentiable sampling network used for reconstruction and classification tasks in point clouds [198]. It introduces a differentiable relaxation for point cloud sampling by approximating sampled points as a mixture of points in the original point cloud. This network can be used as a front to networks for multiple tasks, unlike conventional approaches that do not consider the downstream task. With this model, the sampling procedure for the representative point cloud classification problem becomes differentiable, allowing for end-to-end optimisation [194]. For the downstream tasks, SampleNet suggests a learned sampling strategy [199]. In this work, the creation of additional data points is how sampling is accomplished [200]. This neural network is intended to choose the keypoints more accurately [201]. By choosing already-existing points from the point cloud, this method restricts itself [202]. The model fails to attain a satisfactory equilibrium between maintaining geometric features and uniform density. Following the sampling process, the original point clouds' moving least squares (MLS) surfaces are modified [203]. There are two major drawbacks to this method: it requires supervised annotations in the form of labels. The first is the restricted scalability due to the high cost of building a systematic annotation strategy and obtaining human annotations. Second, when several objects are present in labelled data obtained in the field (e.g., from a vehicle-mounted LiDAR sensor), it becomes very difficult and time-consuming to determine whether points in a point cloud of 10,000 points belong to a car, the street, or another automobile [204]. Figure 28 shows the training of the proposed method.

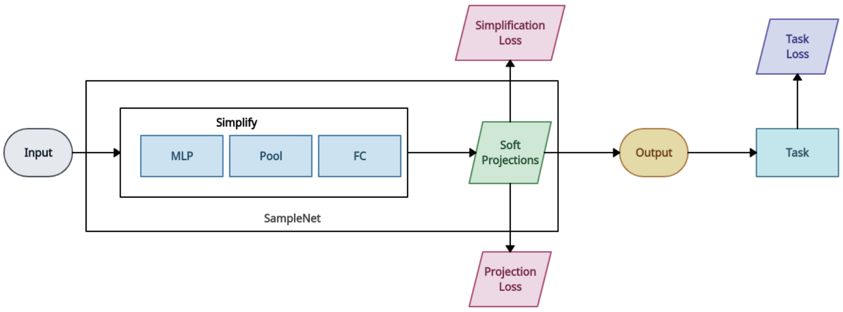


Figure 28. Training of the proposed sampling method [171].

Table 12 provides the advantages and limitations of the 3D augmentation models reviewed in this study. All things considered, traditional techniques for downsampling point clouds frequently result in more computation or the removal of significant points. SamPleNet [171] and CP-Net [170] provide answers to these problems. While SampleNet [171] presents a differentiable relaxation for point cloud sampling, which may be used as a front to networks for numerous tasks, CP-Net [170] globally filters away unnecessary points while maintaining the significant ones. These recent developments provide the groundwork for future improvements in downstream tasks and are critical to 3D point cloud processing.

Table 12. Advantages and limitations of 3D augmentation models.

Model	Advantages	Limitations
MaskNet [165]	Rejects noise in even partial clouds in a rather computationally inexpensive manner.	Requires the input of both a partial and complete point cloud.
GDPNet [166]	Deals with the permutation-invariance problem and builds hierarchies of local or non-local features to effectively address the denoising problem.	The point clouds' geometric characteristics are often oversmoothed.
DMR [167]	Patch manifold reconstruction (PMR) upsampling technique is straightforward and efficient.	Downsampling step invariably results in detail loss, especially at low noise levels, and could also oversmooth by removing some useful information.
PU-Net [168]	Both reconstruction loss and repulsion loss are jointly utilised to improve the quality of the output.	Only learns spatial relationships at a single level of multi-step point cloud decoding via self-attention.
MPU [169]	Trained end-to-end on high-resolution point clouds and emphasises a certain level of detail by altering the spatial span of the receptive field in various steps.	Cannot be used for completion tasks and is restricted to upsampling sparse locations.
CP-Net [170]	Final representations typically retain crucial points that take up a significant number of channels.	Potential loss of information due to the down-sampling process.
SampleNet [171]	Sampling procedure for the representative point cloud classification problem becomes differentiable, allowing for end-to-end optimisation.	Fails to attain a satisfactory equilibrium between maintaining geometric features and uniform density.

7. Point Cloud Completion

Point clouds are the most widely used depiction of 3D data, and they are frequently used in practical applications. However, acquired point clouds are typically highly incomplete and sparse due to self-occlusion and poor sensor resolution, which hinders further applications. Thus, recovering complete point clouds is an essential task, the main goals of which are to densify sparse surfaces, infer missing sections, and preserve the details of incomplete observations. Because point clouds are inherently chaotic and unstructured (especially when taken from real-world settings), their completion is typically non-trivial. Table 13 provides the advantages and limitations of point cloud completion models reviewed in this study.

PCN [205]: A coarse-to-fine point set generator and a permutation-invariant, non-convolutional feature extractor were combined by the model's designers to create a single, end-to-end trained network. PCN is an encoder–decoder network, where the encoder produces a k-dimensional feature vector from the input point cloud. Using this feature vector as input, the decoder generates a coarse and detailed output point cloud. The loss function, which is used to train the entire network via backpropagation, is calculated between the outputs of the decoder and the ground truth point cloud. The authors did not specifically mandate the network to maintain the input points in its output, in contrast to an autoencoder. On the contrary, the network acquires knowledge of a projection from the space of incomplete observations to the space of fully formed shapes. This network's primary drawback is that the encoder requires training data to be prepared in partial shapes since it expects a test input that is identical to the training data [95]. The lack of utilisation of object completion and shape creation architectures, like PCN [205], by 3D object detectors during the LiDAR point cloud inference could result in improved detection performance [206]. This point cloud completion method's max-pooling process during the encoding phase, where fine-grained information is lost and scarcely recoverable during the decoding phase, is its bottleneck [61]. This model, which focuses on object-level completion, works under the assumption that a single item has been found manually and that the input consists only of the points on this object. Consequently, this model is not appropriate for the goal of object detection [189]. Figure 29 shows the architecture of PCN.

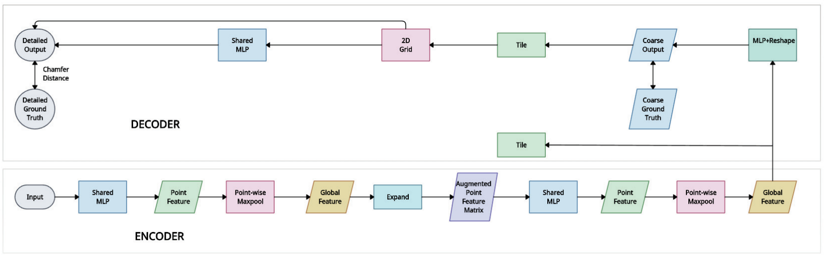


Figure 29. Architecture of PCN [205].

Unpaired scan completion network [207]: The authors provide an unpaired point-based scan completion technique that can be learned without having explicit correspondence between example complete shape models (like synthetic models) and incomplete point sets (like raw scans). Due to the lack of specific instances of real complete scans required by this network, large-scale real 3D scans that are already available (unpaired) can be used directly as training data. This is accomplished by creating a generative adversarial network (GAN), in which the input is transformed into a suitable latent representation by a generator, also known as an adaptation network, so that a discriminator is unable to distinguish between the transformed latent variables and the latent variables derived from training data (i.e., whole-shape models). Working in two distinct latent spaces with independently learned manifolds of scanned and synthesised object data, the generator intuitively performs the crucial operation of transforming raw partial point sets into clean and complete point sets. This model struggles to generate diverse samples, capture fine-grained details, or condition on sparse inputs. However, it can infer believable global structures [97]. Training GANs can be difficult due to common errors such as mode collapse [208]. This model, which focuses on object-level completion, works under the assumption that a single item has been found manually and that the input consists only of the points on this object. Consequently, this model is not appropriate for the goal of object detection [189].

Morphing and sampling-based network [209]: A network that completes the partial point cloud in two steps has been proposed by the authors. Using an autoencoder architecture, a set of 2-manifold-like surface elements that can be 2D parameterised is used in the first stage to put together an entire point cloud. In order to obtain an evenly distributed subset point cloud from the combination of the coarse-grained prediction and the input point cloud, a sampling procedure is used in the second stage. Then, given the point cloud, a point-wise residual is learned, allowing for fine-grained features. This model uses the earth mover's distance (EMD) as a better metric for measuring completion quality because, by solving the linear assignment problem, it forces model outputs to have the same density as the ground truths [210]. This model relieves the structural loss brought on by MLPs and recovers the entire point cloud of an object by estimating a group of parametric surface elements [211]. This approach frequently disregards the spatial correlation between points [190]. This model lacks the conditional generative ability based on partial observation, instead generating complete shapes mostly through learning a deterministic partial-to-complete mapping [212]. Although this approach produces encouraging results when applied to in-domain data, it is difficult to generalise to out-of-domain data, which includes real-world scans or data with various incomplete forms [213]. Figure 30 shows the architecture of the morphing and sampling-based network.

PF-Net [214]: This model accepts a partial point cloud as input and only outputs the portion of the point cloud that is missing, not the entire object, in order to maintain the original part's spatial arrangements. As a result, it helps the network concentrate on identifying the location and structure of missing components by preserving the geometrical features of the original point cloud after restoration. Using a new feature extractor called combined multilayer perception (CMLP), the authors propose a multi-resolution encoder (MRE) to extract multilayer features from the partial point cloud and its low-resolution feature points. The missing point cloud is also intended to be generated hierarchically using a point pyramid decoder (PPD). PPD is a multi-scale generating network that predicts primary, secondary, and detailed points from layers with varying depths. It is based on feature points. The lack of utilisation of object completion and shape creation architectures, like PF-Net [214], by 3D object detectors during the LiDAR point cloud inference could result in improved detection performance [206]. This point cloud completion method's max-pooling process during the encoding phase, where fine-grained information is lost and scarcely recoverable during the decoding phase, is its bottleneck [61]. In the ShapeNet-55 benchmarks, PFNet, which aims to predict objects' missing components directly, fails because of the huge diversity [61]. This approach is still unable to predict a point splitting pattern that is locally structured. The primary issue is the fact that this approach solely concentrates on increasing the number of points and reconstructing the overall shape, neglecting to maintain an organised generation process for points inside specific regions. This makes it challenging to capture localised, intricate 3D shape structures and geometries using this method [190]. This model's intricate design results in a comparatively large number of parameters [215]. Figure 31 shows the architecture of PF-Net.

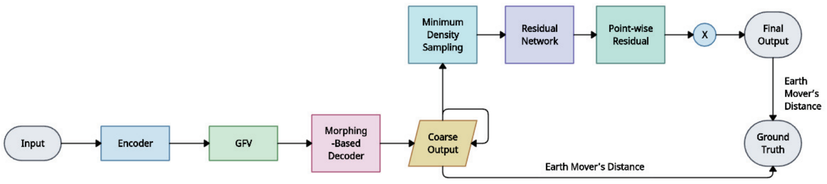


Figure 30. Architecture of MSN [209].

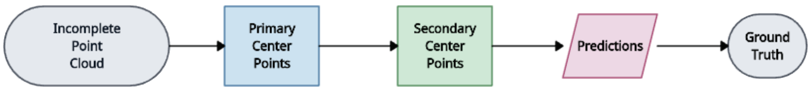


Figure 31. Architecture of PF-Net [214].

GRNet [211]: In order to regularise unordered point clouds and specifically maintain the structure and context of point clouds, the authors introduce 3D grids as intermediary representations. *Gridding*, *gridding reverse*, and *cubic feature sampling* are the three differentiable layers that make up the Gridding Residual Network (GRNet), which is proposed for point cloud completion together with 3D CNN and MLP. In the process of *gridding*, an interpolation function that quantifies the geometric relationships of the point cloud is used to weight the eight vertices of the 3D grid cell that each point in the point cloud resides in. The network then uses a 3D convolutional neural network (3D CNN) with skip connections to learn spatially and contextually aware features, filling in the gaps in the incomplete point cloud. *Gridding reverse* then replaces each 3D grid cell with a new point whose location is the weighted sum of the 3D grid cell's eight vertices, converting the resulting 3D grid into a coarse point cloud. By concatenating the features of the corresponding eight vertices of the 3D grid cell that the point lies in; then, the following *cubic feature sampling* recovers features for every point in the coarse point cloud. To obtain the final finished point cloud, an MLP receives the features and the coarse point cloud. This model, which focuses on object-level completion, works under the assumption that a single item has been found manually and that the input consists only of the points on this object. Consequently, this model is not appropriate for the goal of object detection [189]. It is difficult to maintain a well-organised structure for points in small patches due to the discontinuous character of the point cloud and the unstructured prediction of points in local regions in this method [125]. Rebuilding low-resolution shapes is the only use for GRNet's voxel representation. This model's intricate design results in a comparatively large number of parameters [215]. Figure 32 shows the overview of GRNet.

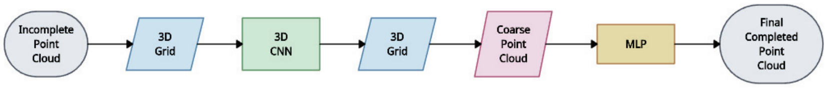


Figure 32. Overview of GRNet [211].

SnowflakeNet [190]: This model focuses specifically on the process of decoding incomplete point clouds. The primary building block of SnowflakeNet is its layers of snowflake point deconvolution (SPD), which simulate the creation of whole point clouds similar to the snowflake growth of points in three dimensions. This model creates points gradually by piling one SPD layer on top of another. Each SPD layer creates child points by dividing their parent point and inheriting the shape properties that the parent point captures. The purpose of disclosing geometrical details is to enable the use of a skip-transformer in SPD to identify point splitting modes that are most appropriate for specific localities. The current SPD layer is split by the skip-transformer, which uses an attention mechanism to summarise the splitting patterns from the previous SPD layer. The network is able to predict extremely detailed geometries because the locally compact, structured point cloud generated by SPD can precisely capture the structural properties of 3D shapes in limited patches. This model's intricate design results in a comparatively large number of parameters [215]. Point clouds are sparse, thus recovering surfaces from them requires non-trivial postprocessing using traditional techniques [216]. There are two inherent limitations to the global feature structure that is extracted from partial inputs by this model. Firstly, fine-grained details are lost easily during pooling operations in the encoding phase and are difficult to recover from a diluted global feature during generation. Secondly, such a global feature is captured from a partial point cloud, which represents only the "incomplete" information of the visible part and goes against the goal of generating the complete shape [217]. Figure 33 shows the overview of SnowflakeNet.

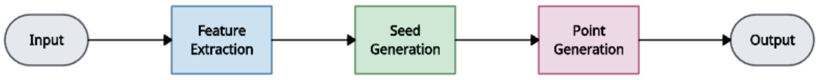


Figure 33. Overview of SnowflakeNet [190].

Table 13. Advantages and limitations of point cloud completion models.

Model	Advantages	Limitations
PCN [205]	Acquires knowledge of a projection from the space of incomplete observations to the space of fully formed shapes.	Requires training data to be prepared in partial shapes since it expects a test input that is identical to the training data.
USCN [207]	Does not require explicit correspondence between example complete shape models and incomplete point sets.	Training GANs can be difficult due to common errors such as mode collapse.
MSN [209]	Uses EMD as a better metric for measuring completion quality.	Frequently disregards the spatial correlation between points.
PF-Net [214]	Accepts a partial point cloud as input and only outputs the portion of the point cloud that is missing.	Model's intricate design results in a comparatively large number of parameters.
GRNet [211]	Uses 3D grids as intermediary representations to maintain unordered point clouds.	Difficult to maintain an organised structure for points in small patches due to the discontinuous character of the point cloud.
SnowflakeNet [190]	Focuses specifically on the process of decoding incomplete point clouds.	Fine-grained details are lost easily during pooling operations in the encoding phase.

8. Conclusions

This article presents a thorough examination of deep learning models applied in the areas of 3D reconstruction, registration, and augmentation. This study delivers an comprehensive overview of diverse models employed for these specific tasks. The advantages and disadvantages of the mentioned models are thoroughly analysed, highlighting the appropriateness of each approach for the specific task. In addition, the study analyses multiple datasets encompassing diverse activities and various 3D data formats. Deep learning has shown promising results in the areas of 3D registration, augmentation, and reconstruction. The objective of this survey was to examine the techniques used by deep learning frameworks for analysing and enhancing 3D image representation, augmentation, and reconstruction. The review of the literature thoroughly examined the advantages and disadvantages of different computer vision algorithms, network architectures, 3D structured data representation, and comparative data methodologies. Several point cloud completion techniques were also examined in relation to the advancement of deep-learning-based image processing technology.

Each phase of the generic methodology for 3D reconstruction, augmentation, and registration can be accomplished utilising distinct algorithms. Distinct methods are required for each constructed object, depending on its size, texture, and visual arrangement. In addition to efficient algorithms, the development of sensors has the potential to enhance the precision of 3D reconstruction in the future. Neural network modelling has numerous advantages. These operations are crucial for various sectors, including robots, autonomous autos, and medical imaging. The problem-solving precision and effectiveness of these domains have experienced a substantial improvement due to the efforts of deep learning models. In order to enhance the performance of these models, a significant amount of additional effort needs to be invested; the field is still in its early stages of development.

While there may be some who argue that traditional, non-deep-learning methods are more advantageous in certain situations, the review has demonstrated that deep learning models have consistently achieved state-of-the-art results in most cases. Given this evaluation, future research endeavours should prioritise the development of more accurate and efficient models capable of handling increasingly larger and more complex data. Moreover, the introduction of new datasets that more accurately represent real-life scenarios can help improve the effectiveness of these models. Furthermore, one attractive avenue for future research involves exploring the concatenation of diverse models to obtain enhanced outcomes.

Author Contributions: P.K.V.: Lead investigator, significantly shaped and wrote the survey. D.K.: Played a key role in writing and enriching the survey's content. E.A.: Originated the survey's structure and reviewed its content for coherence and accuracy. C.O.: Acted as the technical editor and internal reviewer, ensuring academic clarity. G.A.: Principal investigator, guided the survey's strategic direction and scholarly integrity. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the SilentBorder project under the grant agreement ID 101021812 of the European Union's Horizon 2020 research and innovation program.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: Author Gholamreza Anbarjafari was employed by the company PwC Advisory and is the owner of iVCV OÜ. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

3D	Three-dimensional
2D	Two-dimensional
LiDAR	Light detection and ranging
RGB-D	Red, green, blue plus depth
CAD	Computer-aided design
MLP	Multilayer perceptron
CNN	Convolutional neural network
FCGFs	Fully convolutional geometric features
GPU	Graphics processing unit
RAM	Random access memory

References

1. Vinodkumar, P.K.; Karabulut, D.; Avots, E.; Ozcinar, C.; Anbarjafari, G. A Survey on Deep Learning Based Segmentation, Detection and Classification for 3D Point Clouds. *Entropy* **2023**, *25*, 635. [CrossRef]
2. Behley, J.; Garbade, M.; Milioto, A.; Quenzel, J.; Behnke, S.; Stachniss, C.; Gall, J. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9297–9307.
3. Armeni, I.; Sener, O.; Zamir, A.R.; Jiang, H.; Brilakis, I.; Fischer, M.; Savarese, S. 3d semantic parsing of large-scale indoor spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1534–1543.
4. Qi, C.R.; Chen, X.; Litany, O.; Guibas, L.J. ImVoteNet: Boosting 3D Object Detection in Point Clouds with Image Votes. *arXiv* **2020**, arXiv:2001.10692. [CrossRef]
5. Zhou, Y.; Tuzel, O. Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4490–4499.
6. Shi, S.; Wang, X.; Li, H. PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. *arXiv* **2018**, arXiv:1812.04244. [CrossRef]
7. Hanocka, R.; Hertz, A.; Fish, N.; Giryes, R.; Fleishman, S.; Cohen-Or, D. Meshcnn: A network with an edge. *ACM Trans. Graph. (TOG)* **2019**, *38*, 1–12. [CrossRef]
8. Wang, S.; Zhu, J.; Zhang, R. Meta-RangeSeg: LiDAR Sequence Semantic Segmentation Using Multiple Feature Aggregation. *arXiv* **2022**, arXiv:2202.13377.
9. Fan, H.; Su, H.; Guibas, L.J. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. *arXiv* **2016**, arXiv:1612.00603. [CrossRef]
10. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. ShapeNet: An Information-Rich 3D Model Repository. *arXiv* **2015**, arXiv:1512.03012.
11. Choy, C.B.; Xu, D.; Gwak, J.; Chen, K.; Savarese, S. 3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer: Cham, Switzerland, 2016; pp. 628–644.
12. Lin, C.H.; Kong, C.; Lucey, S. Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction. *arXiv* **2017**, arXiv:1706.07036. [CrossRef]
13. Zhang, Y.; Liu, Z.; Liu, T.; Peng, B.; Li, X. RealPoint3D: An Efficient Generation Network for 3D Object Reconstruction From a Single Image. *IEEE Access* **2019**, *7*, 57539–57549. [CrossRef]
14. Xiang, Y.; Kim, W.; Chen, W.; Ji, J.; Choy, C.; Su, H.; Mottaghi, R.; Guibas, L.; Savarese, S. Objectnet3d: A large scale database for 3d object recognition. In Proceedings of the Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part VIII 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 160–176.
15. Navaneet, K.L.; Mathew, A.; Kashyap, S.; Hung, W.C.; Jampani, V.; Babu, R.V. From Image Collections to Point Clouds with Self-supervised Shape and Pose Networks. *arXiv* **2020**, arXiv:2005.01939. [CrossRef]
16. Sun, X.; Wu, J.; Zhang, X.; Zhang, Z.; Zhang, C.; Xue, T.; Tenenbaum, J.B.; Freeman, W.T. Pix3d: Dataset and methods for single-image 3d shape modeling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2974–2983.
17. Bautista, M.A.; Talbott, W.; Zhai, S.; Srivastava, N.; Susskind, J.M. On the generalization of learning-based 3D reconstruction. *arXiv* **2020**, arXiv:2006.15427. [CrossRef]
18. Rezende, D.J.; Eslami, S.M.A.; Mohamed, S.; Battaglia, P.; Jaderberg, M.; Heess, N. Unsupervised Learning of 3D Structure from Images. *arXiv* **2016**, arXiv:1607.00662. [CrossRef]
19. LeCun, Y. The MNIST Database of Handwritten Digits. 1998. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 12 November 2023).

20. Zhang, X.; Zhang, Z.; Zhang, C.; Tenenbaum, J.B.; Freeman, W.T.; Wu, J. Learning to Reconstruct Shapes from Unseen Classes. *arXiv* **2018**, arXiv:1812.11166.
21. Wu, J.; Wang, Y.; Xue, T.; Sun, X.; Freeman, W.T.; Tenenbaum, J.B. MarrNet: 3D Shape Reconstruction via 2.5D Sketches. *arXiv* **2017**, arXiv:1711.03129. [CrossRef]
22. Xiang, Y.; Mottaghi, R.; Savarese, S. Beyond PASCAL: A Benchmark for 3D Object Detection in the Wild. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), Steamboat Springs, CO, USA, 24–26 March 2014.
23. Yan, X.; Yang, J.; Yumer, E.; Guo, Y.; Lee, H. Perspective Transformer Nets: Learning Single-View 3D Object Reconstruction without 3D Supervision. *arXiv* **2016**, arXiv:1612.00814. [CrossRef]
24. Zhu, R.; Galoogahi, H.K.; Wang, C.; Lucey, S. Rethinking Reprojection: Closing the Loop for Pose-Aware Shape Reconstruction from a Single Image. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 57–65. [CrossRef]
25. Xiao, J.; Hays, J.; Ehinger, K.A.; Oliva, A.; Torralba, A. SUN database: Large-scale scene recognition from abbey to zoo. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 3485–3492. [CrossRef]
26. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part V 13; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
27. Wu, J.; Zhang, C.; Xue, T.; Freeman, W.T.; Tenenbaum, J.B. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. *arXiv* **2016**, arXiv:1610.07584. [CrossRef]
28. Wu, Z.; Song, S.; Khosla, A.; Tang, X.; Xiao, J. 3D ShapeNets for 2.5D Object Recognition and Next-Best-View Prediction. *arXiv* **2014**, arXiv:1406.5670.
29. Lim, J.J.; Pirsiavash, H.; Torralba, A. Parsing ikea objects: Fine pose estimation. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 2992–2999.
30. Xie, H.; Yao, H.; Zhang, S.; Zhou, S.; Sun, W. Pix2Vox++: Multi-scale Context-aware 3D Object Reconstruction from Single and Multiple Images. *Int. J. Comput. Vis.* **2020**, *128*, 2919–2935. [CrossRef]
31. Gwak, J.; Choy, C.B.; Garg, A.; Chandraker, M.; Savarese, S. Weakly supervised 3D Reconstruction with Adversarial Constraint. *arXiv* **2017**, arXiv:1705.10904. [CrossRef]
32. Banani, M.E.; Corso, J.J.; Fouhey, D.F. Novel Object Viewpoint Estimation through Reconstruction Alignment. *arXiv* **2020**, arXiv:2006.03586. [CrossRef]
33. Turk, G.; Levoy, M. Zippered polygon meshes from range images. In Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, Orlando, FL, USA, 24–29 July 1994; pp. 311–318.
34. Hoang, L.; Lee, S.H.; Kwon, O.H.; Kwon, K.R. A Deep Learning Method for 3D Object Classification Using the Wave Kernel Signature and A Center Point of the 3D-Triangle Mesh. *Electronics* **2019**, *8*, 1196. [CrossRef]
35. Kato, H.; Ushiku, Y.; Harada, T. Neural 3D Mesh Renderer. *arXiv* **2017**, arXiv:1711.07566. [CrossRef]
36. Pan, J.; Li, J.Y.; Han, X.; Jia, K. Residual MeshNet: Learning to Deform Meshes for Single-View 3D Reconstruction. In Proceedings of the 2018 International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; pp. 719–727.
37. Wang, N.; Zhang, Y.; Li, Z.; Fu, Y.; Liu, W.; Jiang, Y.G. Pixel2mesh: Generating 3d mesh models from single rgb images. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 52–67.
38. Popov, S.; Bauszat, P.; Ferrari, V. CoReNet: Coherent 3D scene reconstruction from a single RGB image. *arXiv* **2020**, arXiv:2004.12989. [CrossRef]
39. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5828–5839.
40. Shilane, P.; Min, P.; Kazhdan, M.; Funkhouser, T. The princeton shape benchmark. In Proceedings of the Shape Modeling Applications, Genova, Italy, 7–9 June 2004; IEEE: Piscataway, NJ, USA, 2004; pp. 167–178.
41. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
42. Henn, A.; Gröger, G.; Stroh, V.; Plümer, L. Model driven reconstruction of roofs from sparse LIDAR point clouds. *ISPRS J. Photogramm. Remote Sens.* **2013**, *76*, 17–29. [CrossRef]
43. Buyukdemircioglu, M.; Kocaman, S.; Kada, M. Deep learning for 3D building reconstruction: A review. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2022**, *43*, 359–366. [CrossRef]
44. Tran, H.; Khoshelham, K. Procedural reconstruction of 3D indoor models from lidar data using reversible jump Markov Chain Monte Carlo. *Remote Sens.* **2020**, *12*, 838. [CrossRef]
45. Mura, C.; Mattausch, O.; Pajarola, R. Piecewise-planar reconstruction of multi-room interiors with arbitrary wall arrangements. In *Proceedings of the Computer Graphics Forum*; Wiley Online Library: Hoboken, NJ, USA, 2016; Volume 35, pp. 179–188.
46. Oesau, S.; Lafarge, F.; Alliez, P. Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS J. Photogramm. Remote Sens.* **2014**, *90*, 68–82. [CrossRef]
47. Khoshelham, K.; Díaz-Vilariño, L. 3D modelling of interior spaces: Learning the language of indoor architecture. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2014**, *40*, 321–326. [CrossRef]

48. Tran, H.; Khoshelham, K.; Kealy, A.; Díaz-Vilariño, L. Shape grammar approach to 3D modeling of indoor environments using point clouds. *J. Comput. Civ. Eng.* **2019**, *33*, 04018055. [CrossRef]
49. Wonka, P.; Wimmer, M.; Sillion, F.; Ribarsky, W. Instant architecture. *ACM Trans. Graph. (TOG)* **2003**, *22*, 669–677. [CrossRef]
50. Becker, S. Generation and application of rules for quality dependent façade reconstruction. *ISPRS J. Photogramm. Remote Sens.* **2009**, *64*, 640–653. [CrossRef]
51. Dick, A.R.; Torr, P.H.; Cipolla, R. Modelling and interpretation of architecture from several images. *Int. J. Comput. Vis.* **2004**, *60*, 111–134. [CrossRef]
52. Becker, S.; Peter, M.; Fritsch, D. Grammar-supported 3d indoor reconstruction from point clouds for "as-built" BIM. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2015**, *2*, 17–24. [CrossRef]
53. Döllner, J. Geospatial artificial intelligence: Potentials of machine learning for 3D point clouds and geospatial digital twins. *PFG-Photogramm. Remote Sens. Geoinf. Sci.* **2020**, *88*, 15–24. [CrossRef]
54. Ma, L.; Liu, Y.; Zhang, X.; Ye, Y.; Yin, G.; Johnson, B.A. Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS J. Photogramm. Remote Sens.* **2019**, *152*, 166–177. [CrossRef]
55. Hoeser, T.; Kuenzer, C. Object detection and image segmentation with deep learning on earth observation data: A review-part i: Evolution and recent trends. *Remote Sens.* **2020**, *12*, 1667. [CrossRef]
56. Rottensteiner, F.; Sohn, G.; Jung, J.; Gerke, M.; Baillard, C.; Benitez, S.; Breitkopf, U. The ISPRS benchmark on urban object classification and 3D building reconstruction. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci. I-3* **2012**, *1*, 293–298. [CrossRef]
57. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
58. Liu, C.; Kong, D.; Wang, S.; Wang, Z.; Li, J.; Yin, B. Deep3D reconstruction: Methods, data, and challenges. *Front. Inf. Technol. Electron. Eng.* **2021**, *22*, 652–672. [CrossRef]
59. Bhat, S.F.; Alhashim, I.; Wonka, P. Adabins: Depth estimation using adaptive bins. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4009–4018.
60. Kasieczka, G.; Nachman, B.; Shih, D.; Amram, O.; Andreassen, A.; Benkendorfer, K.; Bortolato, B.; Brooijmans, G.; Canelli, F.; Collins, J.H.; et al. The LHC Olympics 2020 a community challenge for anomaly detection in high energy physics. *Rep. Prog. Phys.* **2021**, *84*, 124201. [CrossRef] [PubMed]
61. Yu, X.; Rao, Y.; Wang, Z.; Liu, Z.; Lu, J.; Zhou, J. Pointr: Diverse point cloud completion with geometry-aware transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 12498–12507.
62. Peng, S.; Niemeyer, M.; Mescheder, L.; Pollefeys, M.; Geiger, A. Convolutional occupancy networks. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part III 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 523–540.
63. Kato, H.; Beker, D.; Morariu, M.; Ando, T.; Matsuoka, T.; Kehl, W.; Gaidon, A. Differentiable rendering: A survey. *arXiv* **2020**, arXiv:2006.12057.
64. Fu, K.; Peng, J.; He, Q.; Zhang, H. Single image 3D object reconstruction based on deep learning: A review. *Multimed. Tools Appl.* **2021**, *80*, 463–498. [CrossRef]
65. Zhang, Y.; Huo, K.; Liu, Z.; Zang, Y.; Liu, Y.; Li, X.; Zhang, Q.; Wang, C. PGNet: A Part-based Generative Network for 3D object reconstruction. *Knowl.-Based Syst.* **2020**, *194*, 105574. [CrossRef]
66. Lu, Q.; Xiao, M.; Lu, Y.; Yuan, X.; Yu, Y. Attention-based dense point cloud reconstruction from a single image. *IEEE Access* **2019**, *7*, 137420–137431. [CrossRef]
67. Yuniarti, A.; Suciati, N. A review of deep learning techniques for 3D reconstruction of 2D images. In Proceedings of the 2019 12th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 18 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 327–331.
68. Monnier, T.; Fisher, M.; Efros, A.A.; Aubry, M. Share with thy neighbors: Single-view reconstruction by cross-instance consistency. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 285–303.
69. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
70. Hu, T.; Wang, L.; Xu, X.; Liu, S.; Jia, J. Self-supervised 3D mesh reconstruction from single images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 6002–6011.
71. Joung, S.; Kim, S.; Kim, M.; Kim, I.J.; Sohn, K. Learning canonical 3d object representation for fine-grained recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 1035–1045.
72. Niemeyer, M.; Mescheder, L.; Oechsle, M.; Geiger, A. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 3504–3515.
73. Biundini, I.Z.; Pinto, M.F.; Melo, A.G.; Marcato, A.L.; Honório, L.M.; Aguiar, M.J. A framework for coverage path planning optimization based on point cloud for structural inspection. *Sensors* **2021**, *21*, 570. [CrossRef]

74. Chibane, J.; Alldieck, T.; Pons-Moll, G. Implicit functions in feature space for 3d shape reconstruction and completion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6970–6981.
75. Collins, J.; Goel, S.; Deng, K.; Luthra, A.; Xu, L.; Gundogdu, E.; Zhang, X.; Vicente, T.F.Y.; Dideriksen, T.; Arora, H.; et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 21126–21136.
76. Sahu, C.K.; Young, C.; Rai, R. Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: A review. *Int. J. Prod. Res.* **2021**, *59*, 4903–4959. [CrossRef]
77. Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4460–4470.
78. Liu, R.; Wu, R.; Van Hoorick, B.; Tokmakov, P.; Zakharov, S.; Vondrick, C. Zero-1-to-3: Zero-shot one image to 3d object. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–3 October 2023; pp. 9298–9309.
79. Xu, D.; Jiang, Y.; Wang, P.; Fan, Z.; Shi, H.; Wang, Z. Sinnerf: Training neural radiance fields on complex scenes from a single image. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 736–753.
80. Kanazawa, A.; Tulsiani, S.; Efros, A.A.; Malik, J. Learning category-specific mesh reconstruction from image collections. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 371–386.
81. Zhou, T.; Brown, M.; Snavely, N.; Lowe, D.G. Unsupervised learning of depth and ego-motion from video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1851–1858.
82. Yu, A.; Ye, V.; Tancik, M.; Kanazawa, A. pixelnerf: Neural radiance fields from one or few images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4578–4587.
83. Sitzmann, V.; Zollhöfer, M.; Wetzstein, G. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Adv. Neural Inf. Process. Syst.* **2019**, *32*. [CrossRef]
84. Enebuse, I.; Foo, M.; Ibrahim, B.S.K.K.; Ahmed, H.; Supmak, F.; Eyobu, O.S. A comparative review of hand-eye calibration techniques for vision guided robots. *IEEE Access* **2021**, *9*, 113143–113155. [CrossRef]
85. Tatarchenko, M.; Richter, S.R.; Ranftl, R.; Li, Z.; Koltun, V.; Brox, T. What do single-view 3d reconstruction networks learn? In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3405–3414.
86. Sünderhauf, N.; Brock, O.; Scheirer, W.; Hadsell, R.; Fox, D.; Leitner, J.; Upcroft, B.; Abbeel, P.; Burgard, W.; Milford, M.; et al. The limits and potentials of deep learning for robotics. *Int. J. Robot. Res.* **2018**, *37*, 405–420. [CrossRef]
87. Han, X.F.; Laga, H.; Bennamoun, M. Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1578–1604. [CrossRef]
88. Varol, G.; Ceylan, D.; Russell, B.; Yang, J.; Yumer, E.; Laptev, I.; Schmid, C. Bodynet: Volumetric inference of 3d human body shapes. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 20–36.
89. Najibi, M.; Ji, J.; Zhou, Y.; Qi, C.R.; Yan, X.; Ettinger, S.; Anguelov, D. Motion inspired unsupervised perception and prediction in autonomous driving. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 424–443.
90. Xu, Q.; Wang, W.; Ceylan, D.; Mech, R.; Neumann, U. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *Adv. Neural Inf. Process. Syst.* **2019**, *32*. [CrossRef]
91. Creswell, A.; White, T.; Dumoulin, V.; Arulkumaran, K.; Sengupta, B.; Bharath, A.A. Generative adversarial networks: An overview. *IEEE Signal Process. Mag.* **2018**, *35*, 53–65. [CrossRef]
92. Zhu, J.Y.; Zhang, Z.; Zhang, C.; Wu, J.; Torralba, A.; Tenenbaum, J.; Freeman, B. Visual object networks: Image generation with disentangled 3D representations. *Adv. Neural Inf. Process. Syst.* **2018**, *31*. [CrossRef]
93. Gadelha, M.; Maji, S.; Wang, R. 3d shape induction from 2d views of multiple objects. In Proceedings of the 2017 International Conference on 3D Vision (3DV), Qingdao, China, 10–12 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 402–411.
94. Chan, E.R.; Monteiro, M.; Kellnhofer, P.; Wu, J.; Wetzstein, G. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 5799–5809.
95. Park, J.J.; Florence, P.; Straub, J.; Newcombe, R.; Lovegrove, S. Deepsdf: Learning continuous signed distance functions for shape representation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–19 June 2019; pp. 165–174.
96. Gao, J.; Shen, T.; Wang, Z.; Chen, W.; Yin, K.; Li, D.; Litany, O.; Gojcic, Z.; Fidler, S. Get3d: A generative model of high quality 3d textured shapes learned from images. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 31841–31854.
97. Mittal, P.; Cheng, Y.C.; Singh, M.; Tulsiani, S. Autosdf: Shape priors for 3d completion, reconstruction and generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 306–315.

98. Li, X.; Liu, S.; Kim, K.; De Mello, S.; Jampani, V.; Yang, M.H.; Kautz, J. Self-supervised single-view 3d reconstruction via semantic consistency. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XIV 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 677–693.
99. de Melo, C.M.; Torralba, A.; Guibas, L.; DiCarlo, J.; Chellappa, R.; Hodgins, J. Next-generation deep learning based on simulators and synthetic data. *Trends Cogn. Sci.* **2022**, *26*. [CrossRef]
100. Loper, M.M.; Black, M.J. OpenDR: An approximate differentiable renderer. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part VII 13; Springer: Berlin/Heidelberg, Germany, 2014; pp. 154–169.
101. Ravi, N.; Reizenstein, J.; Novotny, D.; Gordon, T.; Lo, W.Y.; Johnson, J.; Gkioxari, G. Accelerating 3d deep learning with pytorch3d. *arXiv* **2020**, arXiv:2007.08501.
102. Michel, O.; Bar-On, R.; Liu, R.; Benaim, S.; Hanocka, R. Text2mesh: Text-driven neural stylization for meshes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 13492–13502.
103. Fahim, G.; Amin, K.; Zarif, S. Single-View 3D reconstruction: A Survey of deep learning methods. *Comput. Graph.* **2021**, *94*, 164–190. [CrossRef]
104. Tang, J.; Han, X.; Pan, J.; Jia, K.; Tong, X. A skeleton-bridged deep learning approach for generating meshes of complex topologies from single rgb images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4541–4550.
105. Xu, Q.; Nie, Z.; Xu, H.; Zhou, H.; Attar, H.R.; Li, N.; Xie, F.; Liu, X.J. SuperMeshing: A new deep learning architecture for increasing the mesh density of physical fields in metal forming numerical simulation. *J. Appl. Mech.* **2022**, *89*, 011002. [CrossRef]
106. Dahnert, M.; Hou, J.; Nießner, M.; Dai, A. Panoptic 3d scene reconstruction from a single rgb image. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 8282–8293.
107. Liu, F.; Liu, X. Voxel-based 3d detection and reconstruction of multiple objects from a single image. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 2413–2426.
108. Pan, J.; Han, X.; Chen, W.; Tang, J.; Jia, K. Deep mesh reconstruction from single rgb images via topology modification networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9964–9973.
109. Mustikovela, S.K.; De Mello, S.; Prakash, A.; Iqbal, U.; Liu, S.; Nguyen-Phuoc, T.; Rother, C.; Kautz, J. Self-supervised object detection via generative image synthesis. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 8609–8618.
110. Huang, Z.; Jampani, V.; Thai, A.; Li, Y.; Stojanov, S.; Rehg, J.M. ShapeClipper: Scalable 3D Shape Learning from Single-View Images via Geometric and CLIP-based Consistency. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 12912–12922.
111. Kar, A.; Häne, C.; Malik, J. Learning a multi-view stereo machine. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. [CrossRef]
112. Yang, G.; Cui, Y.; Belongie, S.; Hariharan, B. Learning single-view 3d reconstruction with limited pose supervision. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 86–101.
113. Huang, Z.; Stojanov, S.; Thai, A.; Jampani, V.; Rehg, J.M. Planes vs. chairs: Category-guided 3d shape learning without any 3d cues. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 727–744.
114. Jiao, L.; Huang, Z.; Liu, X.; Yang, Y.; Ma, M.; Zhao, J.; You, C.; Hou, B.; Yang, S.; Liu, F.; et al. Brain-inspired Remote Sensing Interpretation: A Comprehensive Survey. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2023**, Volume *16*, 2992–3033. [CrossRef]
115. Yang, Z.; Ren, Z.; Bautista, M.A.; Zhang, Z.; Shan, Q.; Huang, Q. FvOR: Robust joint shape and pose optimization for few-view object reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 2497–2507.
116. Bechtold, J.; Tatarchenko, M.; Fischer, V.; Brox, T. Fostering generalization in single-view 3d reconstruction by learning a hierarchy of local and global shape priors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 15880–15889.
117. Thai, A.; Stojanov, S.; Upadhya, V.; Rehg, J.M. 3d reconstruction of novel object shapes from single images. In Proceedings of the 2021 International Conference on 3D Vision (3DV), London, UK, 1–3 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 85–95.
118. Yang, X.; Lin, G.; Zhou, L. Single-View 3D Mesh Reconstruction for Seen and Unseen Categories. *IEEE Trans. Image Process.* **2023**, *32*, 3746–3758. [CrossRef] [PubMed]
119. Anciukevicius, T.; Fox-Roberts, P.; Rosten, E.; Henderson, P. Unsupervised Causal Generative Understanding of Images. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 37037–37054.
120. Fan, H.; Su, H.; Guibas, L.J. A point set generation network for 3d object reconstruction from a single image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 605–613.
121. Niemeyer, M.; Geiger, A. Giraffe: Representing scenes as compositional generative neural feature fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 11453–11464.
122. Or-El, R.; Luo, X.; Shan, M.; Shechtman, E.; Park, J.J.; Kemelmacher-Shlizerman, I. Stylesdf: High-resolution 3d-consistent image and geometry generation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 13503–13513.

123. Xie, H.; Yao, H.; Sun, X.; Zhou, S.; Zhang, S. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 2690–2698.
124. Melas-Kyriazi, L.; Laina, I.; Rupprecht, C.; Vedaldi, A. Realfusion: 360deg reconstruction of any object from a single image. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 8446–8455.
125. Xiang, P.; Wen, X.; Liu, Y.S.; Cao, Y.P.; Wan, P.; Zheng, W.; Han, Z. Snowflake point deconvolution for point cloud completion and generation with skip-transformer. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 6320–6338. [CrossRef]
126. Boulch, A.; Marlet, R. Poco: Point convolution for surface reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 6302–6314.
127. Wen, X.; Zhou, J.; Liu, Y.S.; Su, H.; Dong, Z.; Han, Z. 3D shape reconstruction from 2D images with disentangled attribute flow. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 3803–3813.
128. Wang, D.; Cui, X.; Chen, X.; Zou, Z.; Shi, T.; Salcudean, S.; Wang, Z.J.; Ward, R. Multi-view 3d reconstruction with transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 5722–5731.
129. Kirillov, A.; Wu, Y.; He, K.; Girshick, R. Pointrend: Image segmentation as rendering. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9799–9808.
130. Chen, Z.; Zhang, H. Learning implicit fields for generative shape modeling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 5939–5948.
131. Wen, C.; Zhang, Y.; Li, Z.; Fu, Y. Pixel2mesh++: Multi-view 3d mesh generation via deformation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1042–1051.
132. Jiang, Y.; Ji, D.; Han, Z.; Zwicker, M. Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1251–1261.
133. Wu, J.; Zhang, C.; Zhang, X.; Zhang, Z.; Freeman, W.T.; Tenenbaum, J.B. Learning shape priors for single-view 3d completion and reconstruction. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 646–662.
134. Ma, W.C.; Yang, A.J.; Wang, S.; Urtasun, R.; Torralba, A. Virtual correspondence: Humans as a cue for extreme-view geometry. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 15924–15934.
135. Goodwin, W.; Vaze, S.; Havoutis, I.; Posner, I. Zero-shot category-level object pose estimation. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 516–532.
136. Myronenko, A.; Song, X. Point Set Registration: Coherent Point Drift. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 2262–2275. [CrossRef] [PubMed]
137. Iglesias, J.P.; Olsson, C.; Kahl, F. Global Optimality for Point Set Registration Using Semidefinite Programming. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 8284–8292. [CrossRef]
138. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A benchmark for the evaluation of RGB-D SLAM systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 573–580.
139. Yew, Z.J.; Lee, G.H. RPM-Net: Robust Point Matching using Learned Features. *arXiv* **2020**, arXiv:2003.13479. [CrossRef]
140. Lu, W.; Wan, G.; Zhou, Y.; Fu, X.; Yuan, P.; Song, S. DeepICP: An End-to-End Deep Neural Network for 3D Point Cloud Registration. *arXiv* **2019**, arXiv:1905.04153. [CrossRef]
141. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? the kitti vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–24 June 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 3354–3361.
142. Lu, W.; Zhou, Y.; Wan, G.; Hou, S.; Song, S. L3-net: Towards learning based lidar localization for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–19 June 2019; pp. 6389–6398.
143. Gojcic, Z.; Zhou, C.; Wegner, J.D.; Wieser, A. The Perfect Match: 3D Point Cloud Matching with Smoothed Densities. *arXiv* **2018**, arXiv:1811.06879. [CrossRef]
144. Zeng, A.; Song, S.; Nießner, M.; Fisher, M.; Xiao, J.; Funkhouser, T. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1802–1811.
145. Gojcic, Z.; Zhou, C.; Wegner, J.D.; Guibas, L.J.; Birdal, T. Learning multiview 3D point cloud registration. *arXiv* **2020**, arXiv:2001.05119. [CrossRef]
146. Choi, S.; Zhou, Q.Y.; Koltun, V. Robust reconstruction of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5556–5565.

147. Ma, J.; Jiang, X.; Fan, A.; Jiang, J.; Yan, J. Image matching from handcrafted to deep features: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 23–79. [CrossRef]
148. Sotiras, A.; Davatzikos, C.; Paragios, N. Deformable medical image registration: A survey. *IEEE Trans. Med Imaging* **2013**, *32*, 1153–1190. [CrossRef]
149. Yang, J.; Li, H.; Campbell, D.; Jia, Y. Go-ICP: A globally optimal solution to 3D ICP point-set registration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 2241–2254. [CrossRef]
150. Huang, X.; Mei, G.; Zhang, J.; Abbas, R. A comprehensive survey on point cloud registration. *arXiv* **2021**, arXiv:2103.02690.
151. Brynte, L.; Larsson, V.; Iglesias, J.P.; Olsson, C.; Kahl, F. On the tightness of semidefinite relaxations for rotation estimation. *J. Math. Imaging Vis.* **2022**, *64*, 57–67. [CrossRef]
152. Yang, H.; Carlone, L. Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 2816–2834. [CrossRef] [PubMed]
153. Huang, S.; Gojcic, Z.; Usvyatsov, M.; Wieser, A.; Schindler, K. Predator: Registration of 3d point clouds with low overlap. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 4267–4276.
154. Yew, Z.J.; Lee, G.H. Regtr: End-to-end point cloud correspondences with transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 6677–6686.
155. Bai, X.; Luo, Z.; Zhou, L.; Chen, H.; Li, L.; Hu, Z.; Fu, H.; Tai, C.L. Pointdsc: Robust point cloud registration using deep spatial consistency. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 15859–15869.
156. Fu, K.; Liu, S.; Luo, X.; Wang, M. Robust point cloud registration framework based on deep graph matching. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 8893–8902.
157. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv* **2017**, arXiv:1706.02413. [CrossRef]
158. Ren, S.; Chen, X.; Cai, H.; Wang, Y.; Liang, H.; Li, H. Color point cloud registration algorithm based on hue. *Appl. Sci.* **2021**, *11*, 5431. [CrossRef]
159. Yao, W.; Chu, T.; Tang, W.; Wang, J.; Cao, X.; Zhao, F.; Li, K.; Geng, G.; Zhou, M. SPPD: A Novel Reassembly Method for 3D Terracotta Warrior Fragments Based on Fracture Surface Information. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 525. [CrossRef]
160. Liu, J.; Liang, Y.; Xu, D.; Gong, X.; Hyyppä, J. A ubiquitous positioning solution of integrating GNSS with LiDAR odometry and 3D map for autonomous driving in urban environments. *J. Geod.* **2023**, *97*, 39. [CrossRef]
161. Du, G.; Wang, K.; Lian, S.; Zhao, K. Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: A review. *Artif. Intell. Rev.* **2021**, *54*, 1677–1734. [CrossRef]
162. Choy, C.; Park, J.; Koltun, V. Fully convolutional geometric features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 8958–8966.
163. Lee, J.; Kim, S.; Cho, M.; Park, J. Deep hough voting for robust global registration. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 15994–16003.
164. Lu, F.; Chen, G.; Liu, Y.; Zhang, L.; Qu, S.; Liu, S.; Gu, R. Hregnet: A hierarchical network for large-scale outdoor lidar point cloud registration. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 16014–16023.
165. Sarode, V.; Dhagat, A.; Srivatsan, R.A.; Zevallos, N.; Lucey, S.; Choset, H. MaskNet: A Fully-Convolutional Network to Estimate Inlier Points. *arXiv* **2020**, arXiv:2010.09185. [CrossRef]
166. Pistilli, F.; Fracastoro, G.; Valsesia, D.; Magli, E. Learning Graph-Convolutional Representations for Point Cloud Denoising. *arXiv* **2020**, arXiv:2007.02578. [CrossRef]
167. Luo, S.; Hu, W. Differentiable Manifold Reconstruction for Point Cloud Denoising. *arXiv* **2020**, arXiv:2007.13551. [CrossRef]
168. Yu, L.; Li, X.; Fu, C.; Cohen-Or, D.; Heng, P. PU-Net: Point Cloud Upsampling Network. *arXiv* **2018**, arXiv:1801.06761. [CrossRef]
169. Wang, Y.; Wu, S.; Huang, H.; Cohen-Or, D.; Sorkine-Hornung, O. Patch-based Progressive 3D Point Set Upsampling. *arXiv* **2018**, arXiv:1811.11286. [CrossRef]
170. Nezhadarya, E.; Taghavi, E.; Liu, B.; Luo, J. Adaptive Hierarchical Down-Sampling for Point Cloud Classification. *arXiv* **2019**, arXiv:1904.08506. [CrossRef]
171. Lang, I.; Manor, A.; Avidan, S. SampleNet: Differentiable Point Cloud Sampling. *arXiv* **2019**, arXiv:1912.03663. [CrossRef]
172. Zaman, A.; Yangyu, F.; Ayub, M.S.; Irfan, M.; Guoyun, L.; Shiya, L. CMDGAT: Knowledge extraction and retention based continual graph attention network for point cloud registration. *Expert Syst. Appl.* **2023**, *214*, 119098. [CrossRef]
173. Zhang, Z.; Li, T.; Tang, X.; Lei, X.; Peng, Y. Introducing Improved Transformer to Land Cover Classification Using Multispectral LiDAR Point Clouds. *Remote Sens.* **2022**, *14*, 3808. [CrossRef]
174. Huang, X.; Li, S.; Zuo, Y.; Fang, Y.; Zhang, J.; Zhao, X. Unsupervised point cloud registration by learning unified gaussian mixture models. *IEEE Robot. Autom. Lett.* **2022**, *7*, 7028–7035. [CrossRef]
175. Zhao, Y.; Fan, L. Review on Deep Learning Algorithms and Benchmark Datasets for Pairwise Global Point Cloud Registration. *Remote Sens.* **2023**, *15*, 2060. [CrossRef]
176. Shi, C.; Chen, X.; Huang, K.; Xiao, J.; Lu, H.; Stachniss, C. Keypoint matching for point cloud registration using multiplex dynamic graph attention networks. *IEEE Robot. Autom. Lett.* **2021**, *6*, 8221–8228. [CrossRef]

177. Wu, Y.; Zhang, Y.; Fan, X.; Gong, M.; Miao, Q.; Ma, W. Inenet: Inliers estimation network with similarity learning for partial overlapping registration. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *33*, 1413–1426. [CrossRef]
178. Wu, Y.; Zhang, Y.; Ma, W.; Gong, M.; Fan, X.; Zhang, M.; Qin, A.; Miao, Q. RORNet: Partial-to-partial registration network with reliable overlapping representations. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**. [CrossRef] [PubMed]
179. Chen, C.; Wu, Y.; Dai, Q.; Zhou, H.Y.; Xu, M.; Yang, S.; Han, X.; Yu, Y. A survey on graph neural networks and graph transformers in computer vision: A task-oriented perspective. *arXiv* **2022**, arXiv:2209.13232.
180. Simonovsky, M.; Komodakis, N. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 June 2017; pp. 3693–3702.
181. Mou, C.; Zhang, J.; Wu, Z. Dynamic attentive graph learning for image restoration. In Proceedings of the IEEE/CVF International Conference on Computer Vision, New Orleans, LA, USA, 18–24 June 2021; pp. 4328–4337.
182. Luo, S.; Hu, W. Score-based point cloud denoising. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 4583–4592.
183. Chen, H.; Wei, Z.; Li, X.; Xu, Y.; Wei, M.; Wang, J. Repcd-net: Feature-aware recurrent point cloud denoising network. *Int. J. Comput. Vis.* **2022**, *130*, 615–629. [CrossRef]
184. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph. (TOG)* **2019**, *38*, 1–12. [CrossRef]
185. Chen, H.; Luo, S.; Gao, X.; Hu, W. Unsupervised learning of geometric sampling invariant representations for 3d point clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 893–903.
186. Zhou, L.; Sun, G.; Li, Y.; Li, W.; Su, Z. Point cloud denoising review: From classical to deep learning-based approaches. *Graph. Model.* **2022**, *121*, 101140. [CrossRef]
187. Liu, W.; Sun, J.; Li, W.; Hu, T.; Wang, P. Deep learning on point clouds and its application: A survey. *Sensors* **2019**, *19*, 4188. [CrossRef]
188. Yin, T.; Zhou, X.; Krähenbühl, P. Multimodal virtual point 3d detection. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 16494–16507.
189. Xu, Q.; Zhou, Y.; Wang, W.; Qi, C.R.; Anguelov, D. Spg: Unsupervised domain adaptation for 3d object detection via semantic point generation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 15446–15456.
190. Xiang, P.; Wen, X.; Liu, Y.S.; Cao, Y.P.; Wan, P.; Zheng, W.; Han, Z. Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 5499–5509.
191. Li, R.; Li, X.; Fu, C.W.; Cohen-Or, D.; Heng, P.A. Pu-gan: A point cloud upsampling adversarial network. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 7203–7212.
192. Wang, X.; Ang, M.H., Jr.; Lee, G.H. Cascaded refinement network for point cloud completion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 790–799.
193. Lang, I.; Manor, A.; Avidan, S. Samplenet: Differentiable point cloud sampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 7578–7588.
194. Chen, C.; Chen, Z.; Zhang, J.; Tao, D. Sasa: Semantics-augmented set abstraction for point-based 3d object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 22 February–1 March 2022; Volume 36, pp. 221–229.
195. Cui, B.; Tao, W.; Zhao, H. High-precision 3D reconstruction for small-to-medium-sized objects utilizing line-structured light scanning: A review. *Remote Sens.* **2021**, *13*, 4457. [CrossRef]
196. Liu, K.; Gao, Z.; Lin, F.; Chen, B.M. Fg-net: A fast and accurate framework for large-scale lidar point cloud understanding. *IEEE Trans. Cybern.* **2022**, *53*, 553–564. [CrossRef]
197. Liu, K.; Gao, Z.; Lin, F.; Chen, B.M. Fg-net: Fast large-scale lidar point clouds understanding network leveraging correlated feature mining and geometric-aware modelling. *arXiv* **2020**, arXiv:2012.09439.
198. Wang, Y.; Yan, C.; Feng, Y.; Du, S.; Dai, Q.; Gao, Y. Storm: Structure-based overlap matching for partial point cloud registration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 1135–1149. [CrossRef] [PubMed]
199. Yang, L.; Shrestha, R.; Li, W.; Liu, S.; Zhang, G.; Cui, Z.; Tan, P. Scenesqueezer: Learning to compress scene for camera relocalization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 8259–8268.
200. Wang, T.; Yuan, L.; Chen, Y.; Feng, J.; Yan, S. Pnp-detr: Towards efficient visual analysis with transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 4661–4670.
201. Zhu, M.; Ghaffari, M.; Peng, H. Correspondence-free point cloud registration with SO (3)-equivariant implicit shape representations. In Proceedings of the Conference on Robot Learning, Auckland, NZ, USA, 14–18 December 2022; pp. 1412–1422.
202. Wang, H.; Pang, J.; Lodhi, M.A.; Tian, Y.; Tian, D. Festa: Flow estimation via spatial-temporal attention for scene point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 14173–14182.
203. Lv, C.; Lin, W.; Zhao, B. Approximate intrinsic voxel structure for point cloud simplification. *IEEE Trans. Image Process.* **2021**, *30*, 7241–7255. [CrossRef] [PubMed]

204. Yang, P.; Snoek, C.G.; Asano, Y.M. Self-Ordering Point Clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 15813–15822.
205. Yuan, W.; Khot, T.; Held, D.; Mertz, C.; Hebert, M. Pcn: Point completion network. In Proceedings of the 2018 International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 728–737.
206. Zamanakos, G.; Tsochatzidis, L.; Amanatiadis, A.; Pratikakis, I. A comprehensive survey of LIDAR-based 3D object detection methods with deep learning for autonomous driving. *Comput. Graph.* **2021**, *99*, 153–181. [CrossRef]
207. Chen, X.; Chen, B.; Mitra, N.J. Unpaired point cloud completion on real scans using adversarial training. *arXiv* **2019**, arXiv:1904.00069.
208. Achituve, I.; Maron, H.; Chechik, G. Self-supervised learning for domain adaptation on point clouds. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Virtual, 5–9 January 2021; pp. 123–133.
209. Liu, M.; Sheng, L.; Yang, S.; Shao, J.; Hu, S.M. Morphing and sampling network for dense point cloud completion. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11596–11603.
210. Zhou, L.; Du, Y.; Wu, J. 3d shape generation and completion through point-voxel diffusion. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 5826–5835.
211. Xie, H.; Yao, H.; Zhou, S.; Mao, J.; Zhang, S.; Sun, W. Grnet: Gridding residual network for dense point cloud completion. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 365–381.
212. Pan, L.; Chen, X.; Cai, Z.; Zhang, J.; Zhao, H.; Yi, S.; Liu, Z. Variational relational point completion network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 8524–8533.
213. Zhang, J.; Chen, X.; Cai, Z.; Pan, L.; Zhao, H.; Yi, S.; Yeo, C.K.; Dai, B.; Loy, C.C. Unsupervised 3d shape completion through gan inversion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 1768–1777.
214. Huang, Z.; Yu, Y.; Xu, J.; Ni, F.; Le, X. Pf-net: Point fractal network for 3d point cloud completion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 7662–7670.
215. Fei, B.; Yang, W.; Chen, W.M.; Li, Z.; Li, Y.; Ma, T.; Hu, X.; Ma, L. Comprehensive review of deep learning-based 3d point cloud completion processing and analysis. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 22862–22883. [CrossRef]
216. Yan, X.; Lin, L.; Mitra, N.J.; Lischinski, D.; Cohen-Or, D.; Huang, H. Shapeformer: Transformer-based shape completion via sparse representation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 6239–6249.
217. Zhou, H.; Cao, Y.; Chu, W.; Zhu, J.; Lu, T.; Tai, Y.; Wang, C. Seedformer: Patch seeds based point cloud completion with upsample transformer. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 416–432.

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Entropy Editorial Office
E-mail: entropy@mdpi.com
www.mdpi.com/journal/entropy

ISBN 978-3-7258-3066-4