Acadlore Transactions on AI and Machine Learning
https://www.acadlore.com/journals/ATAIML

Enhanced Named Entity Recognition Based on Multi-Feature Fusion Using Dual Graph Neural Networks

Hanzhao Gu[1], Jialin Ma[1*], Yanran Zhao[2], Ashim Khadka[3]

[1] Faculty of Computer and Software Engineering, Huaiyin Institute of Technology, 223003 Huaian, China
[2] Faculty of Internet of Things Engineering, Wuxi Taihu University, 214063 Wuxi, China
[3] Nepal College of Information Technology, Pokhara University, 44700 Lalitpur, Nepal

[*] Correspondence: Jialin Ma (majl@hyit.edu.cn)

Received: 01-15-2024 **Revised:** 03-20-2024 **Accepted:** 03-28-2024

Citation: H. Z. Gu, J. L. Ma, Y. R. Zhao, and A. Khadka, "Enhanced Named Entity Recognition based on multi-feature fusion using dual Graph Neural Networks," *Acadlore Trans. Mach. Learn.*, vol. 3, no. 2, pp. 84–93, 2024. https://doi.org/10.56578/ataiml030202.

© 2024 by the author(s). Published by Acadlore Publishing Services Limited, Hong Kong. This article is available for free download and can be reused and cited, provided that the original published version is credited, under the CC BY 4.0 license.

Abstract: Named Entity Recognition (NER), a pivotal task in information extraction, is aimed at identifying named entities of various types within text. Traditional NER methods, however, often fall short in providing sufficient semantic representation of text and preserving word order information. Addressing these challenges, a novel approach is proposed, leveraging dual Graph Neural Networks (GNNs) based on multi-feature fusion. This approach constructs a co-occurrence graph and a dependency syntax graph from text sequences, capturing textual features from a dual-graph perspective to overcome the oversight of word interdependencies. Furthermore, Bidirectional Long Short-Term Memory Networks (BiLSTMs) are utilized to encode text, addressing the issues of neglecting word order features and the difficulty in capturing contextual semantic information. Additionally, to enable the model to learn features across different subspaces and the varying degrees of information significance, a multi-head self-attention mechanism is introduced for calculating internal dependency weights within feature vectors. The proposed model achieves F1-scores of 84.85% and 96.34% on the CCKS-2019 and Resume datasets, respectively, marking improvements of 1.13 and 0.67 percentage points over baseline models. The results affirm the effectiveness of the presented method in enhancing performance on the NER task.

Keywords: Named Entity Recognition; Graph Neural Networks; Dependency syntax graph; Co-occurrence graph; Multi-feature fusion

1 Introduction

NER, one of the significant research directions in the field of Natural Language Processing (NLP), is recognized for its capability to identify specific entities such as names of people, places, organizations, and other named entities within textual data. In various downstream tasks of NLP, including information retrieval, knowledge graphs, sentiment analysis, and question-answering systems, NER plays a crucial role. Consequently, the effective and accurate identification of specific entity information from text holds significant importance for computer processing of textual data.

In recent years, GNNs [1] have demonstrated exceptional performance in capturing topological information within data, making them particularly suitable for handling irregular graph-structured data. They have achieved remarkable results in both computer vision and NLP domains. Yao et al. [2] constructed a heterogeneous graph for the entire corpus based on word co-occurrence and text relationships, followed by the extraction of graph features using Graph Convolutional Networks (GCNs) [1]. To obtain more information within the GCN, Hu et al. [3] introduced thematic nodes during graph construction, while Xin et al. [4] incorporated label information. Zhang et al. [5] constructed co-occurrence graphs for each text, utilizing Gated Graph Neural Networks (GGNNs) for information propagation. The models mentioned above relied solely on co-occurrence information to build text graphs, neglecting other types of information such as semantics. Sui et al. [6] proposed a Trigger-GNN for nested NER, which obtains complementary annotation embeddings through entity trigger encoding and semantic matching, utilizing an efficient graph message passing mechanism. Peng et al. [7] introduced a method using semantic and syntactic graphs for aspect-level sentiment analysis. Wu et al. [8] modeled documents as graphs to capture non-contiguous and long-distance semantics, extracting

features from graphs across different domains through joint training of hierarchical GNNs. Some of these GNN models require constructing graphs for the entire corpus, leading to high space complexity. Others build text networks solely from the perspective of neighborhood word co-occurrence, overlooking the inherent syntactic dependency relationships within textual language, i.e., neglecting the representation of textual information at the level of syntactic dependency relationships in the language itself.

Addressing the aforementioned challenges, this study explores various graph construction methods to comprehensively capture textual features from multiple dimensions, thereby enhancing NER performance. A novel NER model based on multi-feature fusion using dual GNN (M-DGNN) is proposed. This approach involves constructing a co-occurrence graph and a dependency syntax graph to learn textual features from a dual-graph perspective; incorporating BiLSTMs to enhance the model's capability in capturing contextual information. Two separate GCNs are employed to learn the global semantic information of both graphs, i.e., the original distance dependency relationships. These are then fused with contextual information through a multi-head self-attention mechanism, enabling the model to learn features across different subspaces and the varying degrees of information significance. This, in turn, allows for a more effective learning of graphical representations. The main contributions of this study are as follows:

(i) The introduction of a joint updating module for GCNs based on co-occurrence graphs and dependency syntax graphs, extracting textual features from a dual-graph perspective. This module learns structured information representations dependent on context and mitigates the impact of incorrect dependency labels.

(ii) Comparative experiments were conducted on two public datasets with several mainstream methods. The results demonstrate that the M-DGNN model outperforms other methods.

2 Related Work

Currently, NER methods based on deep learning capitalize on the strong representational learning capabilities of deep neural networks, which autonomously learn features related to named entities from textual data. Models based on Convolutional Neural Networks (CNNs) are particularly powerful in generating local features for sentences [9], and they are very fast in their extended forms due to their parallelism [10]. Models based on Recurrent Neural Networks (RNN) excel in modeling sequential information [11]. Huang et al. [12] proposed to capture contextual information utilizing BiLSTMs and to model label relationships through Conditional Random Field (CRF) models, thereby enhancing the accuracy of NER. Li et al. [13] introduced a BiLSTM sub-classification model into the base model, effectively addressing the issue of nested medical entities. Ma and Hovy [14] presented the BiLSTM-CNN-CRF model, which first extracts character-level features using CNN, followed by further extraction of contextual features and final output using the BiLSTM-CRF model. Gui et al. [15] processed the task of Chinese NER in parallel with a CNN model that integrates dictionary information and improved model performance through a novel feedback mechanism to resolve ambiguity between words. Dang et al. [16] optimized word vectors with linguistic features in the D3NER model, which incorporates a BiLSTM-CRF framework. Strubell et al. [10] utilized dilated convolutions to extend the sequence coverage distance, thereby acquiring more contextual information. This method effectively balances computational speed and feature extraction for long sequences in NER tasks.

In recent years, GNNs have garnered increasing attention in the field of NLP, as traditional CNNs and RNNs have shown limitations in effectively utilizing structural information between documents, hierarchical classifications, and dependency trees among other graph data. GNN models, through the use of recurrent neural structures, propagate information from surrounding nodes, iteratively reaching a stable fixed point to obtain the vector representation of target nodes. Driven by GNNs, scholars, drawing on the concepts of CNNs and RNNs, have defined and designed GCNs for processing graph-structured data, applying them to classification tasks [2]. The convolution operation in GCNs combines the feature vectors of nodes with the graph structure between them. With each graph convolution operation, a node's feature vector is updated through the graph structure using information from neighboring nodes, thereby ensuring that similar nodes possess similar feature vectors. Existing evidence demonstrates the powerful feature extraction capabilities of GCNs, capable of extracting the data features of graph structures and applying them in areas such as relation classification and label classification. This study utilizes the existing capabilities of GCNs in processing graph-structured data to extract features from long distances.

3 The NER Model Based on M-DGNN

The NER model based on M-DGNN, as proposed in this study, is depicted in Figure 1. The encoding layer of the model comprises two sub-modules: a BiLSTM network module for extracting contextual features and a GCN module for extracting global features. Initially, textual sequences are input into the BiLSTM model to learn context feature vectors. Concurrently, by utilizing textual sequences in conjunction with co-occurrence graphs and dependency syntax graphs, dual-graph GCNs are constructed to obtain global feature vectors. Subsequently, these two types of feature vectors are input into a multi-head self-attention layer for feature fusion. Finally, a CRF model is used to decode the optimal encoding sequence from the fused feature vectors.

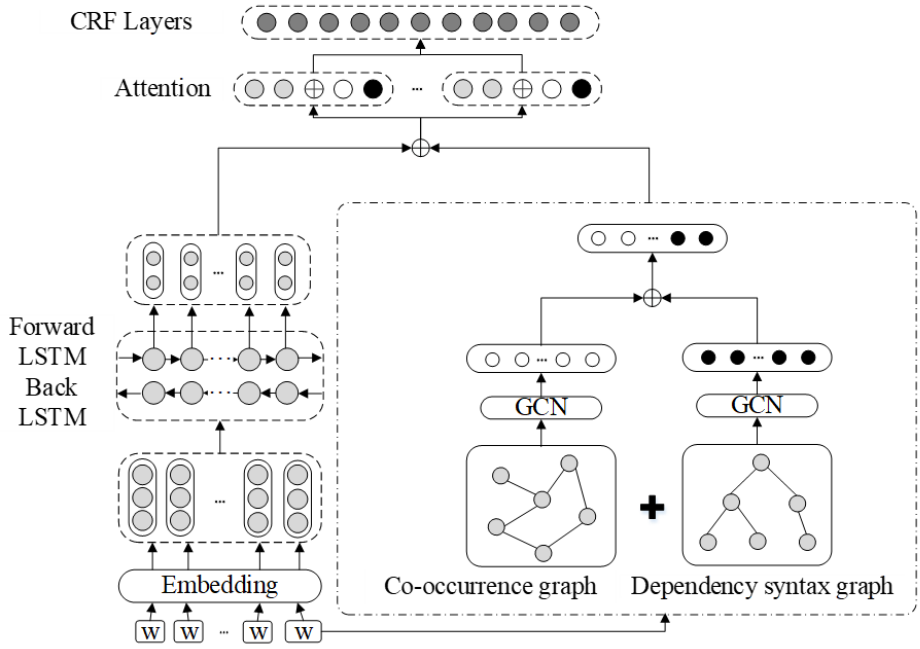

Figure 1. The NER model based on M-DGNN

3.1 Contextual Semantic Learning

Deep learning-based NER models typically transform text sequences into vector sequences, where traditional one-hot encoding representations fail to effectively capture the contextual relationships and issues of data sparsity within sentences. In recent years, low-dimensional dense vector representations, such as word2vec and Global Vectors for Word Representation (GloVe), have gradually replaced traditional methods, offering advantages in extracting semantic information, conserving computational resources, and exhibiting strong generalization capabilities. In this study, the GloVe model is utilized for unsupervised training of text sequences to generate word vector representations rich in semantic information. Let the preprocessed text sequence be denoted as $W = \{w_1, w_2, \ldots, w_3, \ldots, w_n\}$. The pre-trained word vector model is employed to represent the text w_i, resulting in $w_i = \left\{w_i^1, w_i^2, \ldots, w_i^j, \ldots, w_i^n\right\}$, where, $w_i^j \in R^{d_w}$, n is the sentence length, and d_w is the word vector dimension.

Long Short-Term Memory Networks (LSTMs), a variant of RNNs, possess the advantage of capturing long-distance dependencies and contextual features within sentences, while effectively mitigating issues of gradient vanishing and explosion. A BiLSTM, composed of forward and backward LSTM, is more conducive to learning contextual information in sentences. This study employs BiLSTM for feature extraction from sentences w_i, with both forward and backward LSTM extracting features that are then concatenated to obtain hidden layer features h_t. The specific formulas are as follows:

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{1}$$

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{2}$$

$$\check{C}_t = tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * \check{C}_t \tag{4}$$

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{5}$$

$$h_t = o_t * tanh\left(C_t\right) \tag{6}$$

where, f_t, i_t, and o_t respectively represent the forget gate, input gate, and output gate at time t; $\check{C}_t$ denotes the candidate memory cell vector at time t; C_t represents the memory cell vector at time t; h_t denotes the output vector of the hidden layer; * indicates the dot product; W and b represent the weight matrix and bias vector, respectively; $\sigma(\cdot)$ and $tanh(\cdot)$ denote the sigmoid activation function and hyperbolic tangent activation function, respectively.

3.2 Global Semantic Learning Based on Dual-Graph GCNs

GCNs are adept at capturing relationships between nodes and have been widely applied in NLP. To effectively capture the dependencies between words, GCNs are utilized to extract features from both co-occurrence graphs and syntactic dependency graphs. Through the dual-graph structure, a complementary acquisition of textual sequential information and global information is achieved.

3.2.1 Construction of co-occurrence graphs

Co-occurrence graphs utilize graph structures to express the co-occurrence relationships between words, aiming to showcase semantic connections between vocabularies by capturing words that frequently appear in the same context within texts. Words in text sequences are represented as nodes, and the edges between nodes denote the co-occurrence relationships between words. A predefined sliding window is moved from left to right along the text sequence. When words appear within the same sliding window, they are considered to have a co-occurrence relationship, thereby constructing a co-occurrence graph, as illustrated in Figure 2.

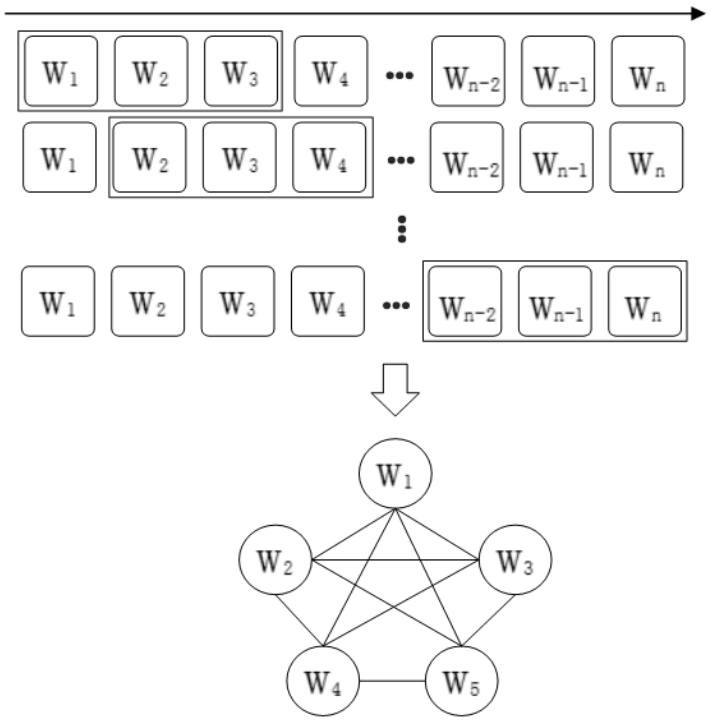

Figure 2. Example of co-occurrence graph construction

3.2.2 Construction of syntactic dependency graphs

Syntactic parsing, a fundamental component of NLP, plays a pivotal role in presenting the grammatical dependency relationships between words within sentences. By analyzing the dependency relationships among words in sentences, the syntactic structure of sentences is determined, revealing their hierarchical structure and grammatical dependencies. Text sequences are analyzed using the StanfordNLP syntactic parser to effectively transform text sequences into graph network data structures. The result of the syntactic analysis for a given text sequence is shown in Figure 3.

From the example in Figure 3, it can be observed that the syntactic analysis results obtained from the syntactic parser not only provide the grammatical roles of each word but also mark the dependency relationships between words. For instance, the relationship between “suffers” and “pain” is labeled as “ccomp,” indicating a clausal complement.

Although the dependency relationships obtained from syntactic analysis are represented as directed edges, for the purposes of constructing syntactic dependency graphs and feature extraction, this study opts to consider directed edges as undirected. By analyzing the dependency relationships among words in sentences with a syntactic parser, syntactic dependency graphs are constructed, enabling a more effective expression of the dependency relationships between words and the overall text structure.

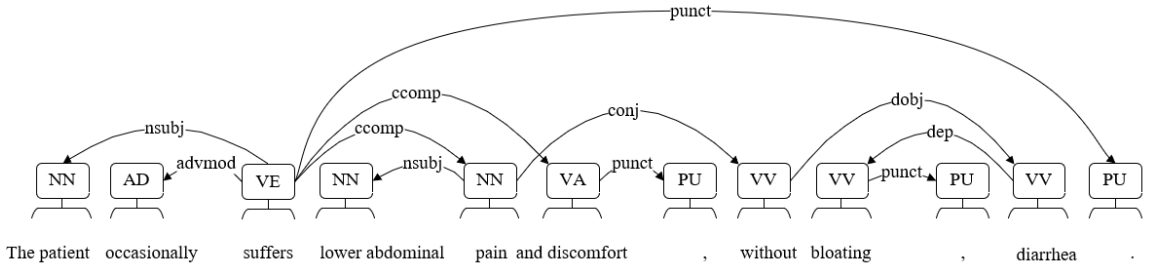

Figure 3. Example of syntactic dependency analysis

Through the construction of graphs, two textual graphs are obtained: a co-occurrence graph $G_1 = (V, E_1)$ and a syntactic dependency graph $G_2 = (V, E_2)$. X is used to represent the initial embedding representations of the textual word nodes. In addition, V denotes the set of nodes in the graph, which corresponds to the collection of words in the text sequence; E_1 represents the set of edges in the co-occurrence graph, indicating the existence of a relationship between two words; E_2 is the set of edges in the syntactic dependency graph.

Two GCNs are employed to learn the topological structures of the two text networks separately. Through neighborhood aggregation operations based on the word connection relationships in the two text networks, the embedding representations of textual words are obtained. The computational formulas are as follows:

$$H_M = RELU\left(A_m H_M W^l\right) \tag{7}$$

$$H_S = RELU\left(A_s H_S W^l\right) \tag{8}$$

where, H_M represents the embedding matrix of the co-occurrence graph, and H_S denotes the embedding matrix of the syntactic dependency graph. $H_M^0 = X\left(X \in R^{N\times L}\right)$ is the input to the first layer of the GCN, N represents the number of word nodes, and L denotes the embedding dimension of word nodes. $A \in R^{N\times N}$ signifies the adjacency matrix, with elements being decimals ranging from 0 to 1, indicating the edge weight between two word nodes. W^l represents the model training weight parameter matrix, and $RELU(\cdot)$ denotes the activation function.

In the dual-graph GCN module, features H_M obtained from the co-occurrence graph G_1 and features H_S obtained from the syntactic dependency graph G_2 are concatenated. The concatenated H_C represents the global features obtained by word embeddings through the dual graphs:

$$H_C = H_M \oplus H_S \tag{9}$$

3.3 Feature Fusion

The multi-head self-attention mechanism aids in capturing syntactic and semantic features between words in sentences, particularly focusing on long-distance dependencies between words, enabling a more effective understanding of internal dependencies within feature vectors. To fuse feature vectors extracted by the GCN and BiLSTM modules, this study introduces a multi-head self-attention mechanism, concentrating on the structural features of the fusion vectors. The fusion vector representing global semantic information and contextual information is denoted as $R = [r_1, r_2, \ldots, r_n]$, where r_i is the vector of the i-th character in the sentence. After undergoing linear transformation, three different vector matrices Q(query), K(Key) and V(Value) are obtained, and Q=K=V=R. By performing dot product operations on each key matrix K using Q, the results are scaled and normalized using a softmax function to obtain attention weights. The specific computational formula is as follows:

$$SelfAttention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{10}$$

where, $\sqrt{d_k}$ represents the dimensionality of key, and $softmax(\cdot)$ indicates probability mapping of the input elements.

Through the multi-head self-attention mechanism, high-dimensional feature vectors are subjected to multiple parallel attention computations, with the number of heads being e. Each attention head possesses distinct feature parameters. After computing the attention information for one head, the information from e heads is concatenated and then linearly transformed to obtain comprehensive feature information fused with attention, represented as:

$$M_i = SelfAttention\left(QW_Q, KW_K, VW_V\right) \tag{11}$$

$$MulAtt(Q, K, V) = Concat\left(M_1, M_2, \ldots, M_e\right) W \tag{12}$$

where, W_Q, W_K, W_V denotes the weight matrix learned during model training, $Concat(\cdot)$ represents the concatenation of feature vectors extracted by e attention heads, and W is the weight matrix generated in the process.

3.4 CRF Decoding

While the multi-head self-attention mechanism effectively fuses features and independently identifies the most probable label for each word, it does not address the issue of the rationality between adjacent labels. To resolve this, a CRF model is employed to introduce effective constraints between labels, thereby ensuring that reasonable label sequences attain higher probability values, resulting in the optimal predicted sequence. For a given sequence $X = \{x_1, x_2, \ldots, x_n\}$ and its predicted sequence $y = \{y_1, y_2, \ldots, y_n\}$, the computed score can be represented as:

$$S(X, y) = \sum_{i=1}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i} \tag{13}$$

where, $A_{y_i, y_{i+1}}$ denotes the probability of transitioning y_i to y_{i+1}, P_{i, y_i} represents the probability of the i-th word being labeled y_i, and $S(x, y)$ signifies the probability of the input sentence sequence x being tagged with the label sequence y.

Thus, the conditional probability of label sequence y in the text representation X is:

$$p(y \mid X) = \frac{e^{S(X,y)}}{\sum_{\tilde{y} \in Y_X} S(X, \tilde{y})} \tag{14}$$

where, Y_X represents the set of all possible labels under the text sequence X. Subsequently, the likelihood function is solved, and the optimal label sequence is decoded to obtain the prediction result, as follows:

$$\log(p(y \mid X)) = S(X, y) - \log\left(\sum_{\tilde{y} \in Y_X} e^{S(X, \tilde{y})}\right) \tag{15}$$

$$y^* = argmax S(X, \tilde{y}) \tag{16}$$

4 Results and Analysis

4.1 Datasets

Experiments were conducted on the Chinese Electronic Medical Records Entity Dataset released by the China Conference on Knowledge Graph and Semantic Computing (CCKS-2019) and the Chinese Resume Dataset. The distribution of the two datasets is shown in Table 1. The datasets employ the BIO tagging scheme: "B" represents Beginning, indicating the start of an entity; "I" represents Inside, indicating the middle part of an entity; "O" represents Outside, indicating information unrelated to any entity.

4.2 Evaluation Metrics

To accurately measure the effectiveness of entity recognition, this study employs three evaluation metrics: precision (P), recall (R), and F1-score (F1). The specific formulas are as follows:

$$P = TP/(TP + FP)$$
$$R = TP/(TP + FN)$$
$$F_1 = 2PR/(P + R)$$

where, TP represents the number of correctly identified entities, FP denotes the number of entities with either boundary determination errors or category classification errors, and FN denotes the number of unrecognized entities.

Table 1. Introduction of the datasets

Dataset	Dataset Division	Number of Sentences/Documents	Entity Types
CCKS-2019	Training set	1000	Surgical procedures, body parts, medications, imaging examinations, diseases and diagnoses, laboratory tests.
	Development set	79	
	Test set	300	
Resume	Training set	3821	Nationalities, educational institutions, addresses, personal names, organization names.
	Development set	463	
	Test set	477	

4.3 Parameter Settings

The word vectors were initialized using GloVe, with an embedding dimension set to 300. The embedding dimension for BiLSTM was set to 100. The graph convolutional module utilized two layers of convolution, with a word embedding dimension of 300. The number of heads in the self-attention mechanism was set to 6. The Adam optimizer was used for training with epochs=200, an initial learning rate of 0.01, and a Dropout rate of 0.5.

4.4 Experimental Results and Analysis

To validate the effectiveness of the model proposed in this study, comparative experiments were conducted on the CCKS-2019 and Resume datasets. The performance comparison of various models on the CCKS-2019 dataset is presented in Table 2 and Table 3.

Chiu and Nichols [17] utilized a CNN model to capture character features, which were then combined with word vectors and other additional features for input into a BiLSTM network. Li et al. [18] employed both sentence and bi-word vectors in their model. Dang et al. [16] proposed the D3NER model, which introduces linguistic features into the BiLSTM-CRF structure to optimize word vectors. Peters et al. [19] introduced the Embeddings from Language Models (ELMo), utilizing BiLSTM as a pre-training structure to obtain dynamic word vectors. Through experimental comparison, the model presented in this study demonstrated superior performance on the CCKS-2019 dataset, with an F1-score exceeding the next best model by 1.13 percentage points.

The performance comparison of models on the Resume dataset is shown in Table 3. Zhang and Yang [20] first proposed using Lattice-LSTM to achieve character-word fusion, enhancing the performance of Chinese NER. Building on this, Liu et al. [21] further optimized the model with four strategies for acceleration, enabling the model to support larger batch sizes. Gui et al. [22] introduced graph methods to the field of NER, utilizing co-occurrence relationships for sequence tagging. Li et al. [23] improved the self-attention mechanism and introduced position encoding to improve the F1-score for NER. Through comparative experiments, the model presented in this study achieved the best results on the Resume dataset, with an F1-score exceeding the next best model by 0.67 percentage points.

Table 2. Comparison of evaluation metrics for different models on the CCKS-2019 dataset

Model	P (%)	R (%)	F1 (%)
BCEL [17]	83.65	82.56	83.24
VT-BiLSTM [18]	83.45	83.68	83.54
D3NER [16]	82.35	83.41	83.62
ELMo [19]	83.65	83.52	83.72
M-DGNN	84.72	84.49	84.85

Table 3. Comparison of evaluation metrics for different models on the resume dataset

Model	P (%)	R (%)	F1 (%)
Lattice-LSTM [20]	94.62	94.23	94.65
WC-LSTM [21]	95.33	94.60	94.74
LB-GNN [22]	95.58	95.52	95.40
FL-Transformer [22]	-	-	95.67
M-DGNN	96.12	95.86	96.34

The experimental results indicate that the introduction of GCNs into NER tasks, combined with the co-occurrence and syntactic dependency information within text sequences, is effective in enhancing the accuracy of NER. Constructing text sequences into graph structures facilitates the exploration of hidden structural information within

the text. GCN models can effectively model the global co-occurrence relationships of words, with embedding representations considering both topological information and word co-occurrence information. Simultaneously, combining BiLSTMs to capture the bidirectional contextual semantic information of texts addresses the insufficiency of GCNs in neglecting textual word order features.

4.5 Parameter Analysis

Figure 4 presents the impact of different sliding window sizes on model performance across two datasets. It can be observed that, on both the CCKS-2019 and Resume datasets, the model's F1-score reaches its peak when the sliding window size is set to 3, and shows a declining trend when the size is either decreased or further increased. Therefore, setting the sliding window too large or too small is not the optimal strategy. A too large sliding window might lead to unnecessary node connections. Conversely, a too small setting reduces the number of words within the window, decreasing the number of node connections in the constructed text graph. This could result in the loss of some important connections between nodes, thereby losing key information and reducing the performance of entity recognition.

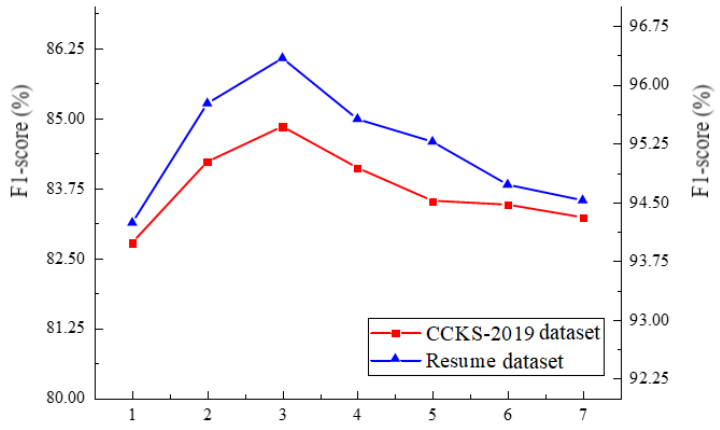

Figure 4. Impact of different sliding window sizes on model performance

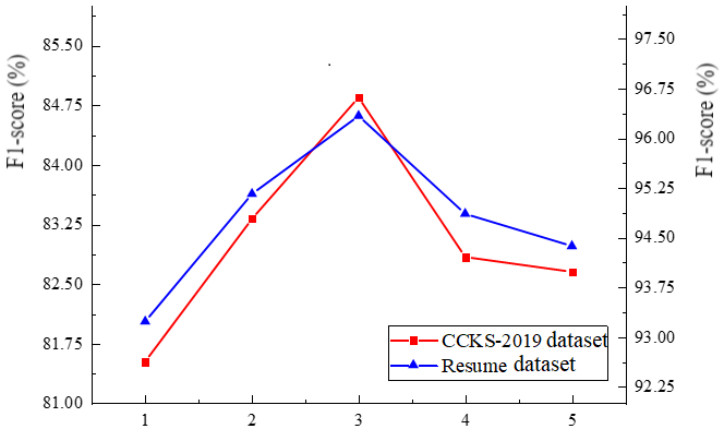

Figure 5. Impact of different numbers of graph convolutional layers on model performance

Figure 5 illustrates the performance of the model under different numbers of graph convolutional layers. It is observed that the model exhibits the highest accuracy when the number of graph convolutional layers is set to 3. Compared to having one or two layers of graph convolution, both accuracy and F1-score show improvement. However, an increase to four layers results in a decrease in both accuracy and F1-score. This decline can be attributed to the issue of over-smoothing that may occur with too many stacked layers of graph convolution, leading to the features of adjacent nodes becoming increasingly similar and thus negatively impacting the performance of entity recognition.

5 Conclusions

This study introduced a NER method based on dual GNNs with multi-feature fusion. The model leverages GCNs, combining co-occurrence and syntactic dependency graphs, to learn the feature information of text word nodes. In addition, a multi-head self-attention mechanism is introduced to calculate the internal dependency significance of feature vectors. Experimental results demonstrate that fully considering the grammatical characteristics of text language and introducing syntactic dependency information significantly enhances the effectiveness of NER. A limitation of this study is the construction of static text graphs. Future work could involve constructing dynamic text graphs and exploring various methods for constructing text graphs, including those based on semantic associations and entity relationships. Furthermore, different variants of GNNs based on text graph features could be experimented with for further improvements.

Data Availability

The data used to support the research findings are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint*, 2016. https://doi.org/10.48550/arXiv.1609.02907

[2] L. Yao, C. S. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*, Hawaii, USA, 2019, pp. 7370–7377. https://doi.org/10.48550/arXiv.1809.05679

[3] L. M. Hu, T. C. Yang, C. Shi, H. Y. Ji, and X. L. Li, "Heterogeneous graph attention networks for semi-supervised short text classification," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019, pp. 4821–4830. https://doi.org/10.18653/v1/D19-1488

[4] Y. Xin, L. L. Xu, J. L. Guo, J. Q. Li, X. Sheng, and Y. Y. Zhou, "Label incorporated graph neural networks for text classification," in *Proceedings of the 25th International Conference on Pattern Recognition*, Milan, Italy, 2021, pp. 8892–8898. https://doi.org/10.1109/ICPR48806.2021.9413086

[5] Y. F. Zhang, X. L. Yu, Z. Y. Cui, S. Wu, Z. Z. Wen, and L. Wang, "Every document owns its structure: Inductive text classification via graph neural networks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 334–339. https://doi.org/10.18653/v1/2020.acl-main.31

[6] Y. Sui, F. Y. Bu, Y. T. Hu, L. Zhang, and W. Yan, "TriggerGNN: A trigger-based graph neural network for nested named entity recognition," in *2022 International Joint Conference on Neural Networks (IJ-CNN)*, Padua, Italy, 2022, pp. 1–8. https://doi.org/10.1109/IJCNN55064.2022.9892555

[7] H. Peng, J. X. Li, Y. He, Y. P. Liu, M. J. Bao, L. H. Wang, Y. Q. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *Proceedings of the 2018 World Wide Web Conference*, Lyon, France, 2018, pp. 1063–1072. https://doi.org/10.1145/3178876.3186005

[8] M. Wu, S. R. Pan, X. Q. Zhu, C. Zhou, and L. Pan, "Domain-adversarial graph neural networks for text classification," in *2019 IEEE International Conference on Data Mining (ICDM)*, Beijing, China, 2019, pp. 648–657. https://doi.org/10.1109/ICDM.2019.00075

[9] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.

[10] E. Strubell, P. Verga, D. Belanger, and A. McCallum, "Fast and accurate entity recognition with iterated dilated convolutions," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, 2017, pp. 2670–2680. https://doi.org/10.18653/v1/D17-1283

[11] R. Kadari, Y. Zhang, W. N. Zhang, and T. Liu, "CCG supertagging via bidirectional LSTM-CRF neural architecture," *Neurocomputing*, vol. 283, pp. 31–37, 2018. https://doi.org/10.1016/j.neucom.2017.12.050

[12] Z. H. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint*, 2015. https://doi.org/10.48550/arXiv.1508.01991

[13] F. Li, M. S. Zhang, B. Tian, B. Chen, G. H. Fu, and D. H. Ji, "Recognizing irregular entities in biomedical text via deep neural networks," *Pattern Recognit. Lett.*, vol. 105, pp. 105–113, 2018. https://doi.org/10.1016/j.patrec.2017.06.009

[14] X. Z. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional LSTM-CNNS-CRF," *arXiv preprint*, 2016. https://doi.org/10.48550/arXiv.1603.01354

[15] T. Gui, R. T. Ma, Q. Zhang, L. J. Zhao, Y. G. Jiang, and X. J. Huang, "CNN-based Chinese NER with lexicon rethinking," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, 2019, pp. 4982–4988. https://doi.org/10.24963/ijcai.2019/692

[16] T. H. Dang, H. Q. Le, T. M. Nguyen, and S. T. Vu, "D3NER: Biomedical named entity recognition using CRF-biLSTM improved with fine-tuned embeddings of various linguistic information," *Bioinformatics*, vol. 34, no. 20, pp. 3539–3546, 2018. https://doi.org/10.1093/bioinformatics/bty356

[17] J. P. C. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNNs," *Trans. Assoc. Comput. Linguist.*, vol. 4, pp. 357–370, 2016. https://doi.org/10.1162/tacl_a_00104

[18] L. S. Li, L. K. Jin, Y. X. Jiang, and D. Huang, "Recognizing biomedical named entities based on the sentence vector/twin word embeddings conditioned bidirectional LSTM," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, 2016, pp. 165–176. https://doi.org/10.1007/978-3-319-47674-2_15

[19] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, New Orleans, Louisiana, USA, 2018, pp. 2227–2237. https://doi.org/10.18653/v1/N18-1202

[20] Y. Zhang and J. Yang, "Chinese NER using lattice LSTM," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, Melbourne, Australia, 2018, pp. 1554–1564. https://doi.org/10.18653/v1/P18-1144

[21] W. Liu, T. G. Xu, Q. H. Xu, J. Y. Song, and Y. R. Zu, "An encoding strategy based word-character LSTM for Chinese NER," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, Minnesota, USA, 2019, pp. 2379–2389. https://doi.org/10.18653/v1/N19-1247

[22] T. Gui, Y. C. Zou, Q. Zhang, M. L. Peng, J. L. Fu, Z. Y. Wei, and X. J. Huang, "A lexicon based graph neural network for Chinese NER," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019, pp. 1040–1050. https://doi.org/10.18653/v1/D19-1096

[23] X. N. Li, H. Yan, X. P. Qiu, and X. J. Huang, "FLAT: Chinese NER using flat lattice transformer," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 6836–6842. https://doi.org/10.18653/v1/2020.acl-main.611