



OPEN

# Relationship extraction between entities with long distance dependencies and noise based on semantic and syntactic features

Lei Wang<sup>1</sup>, Fei Wu<sup>1✉</sup>, Xiaoqing Liu<sup>1,2</sup>, Jilong Cao<sup>1</sup>, Mingwei Ma<sup>1</sup> & Zhaoyang Qu<sup>1</sup>

Relation extraction plays a crucial role in tasks such as text processing and knowledge graph construction. However, existing extraction algorithms struggle to maintain accuracy when dealing with long-distance dependencies between entities and noise interference. To address these challenges, this paper proposes a novel relation extraction method that integrates semantic and syntactic features for handling noisy long-distance dependencies. Specifically, we leverage contextual semantic features generated by the pre-trained BERT model alongside syntactic features derived from dependency syntax graphs, effectively utilizing the complementary strengths of both sources of information to enhance the model's performance in long-distance dependency scenarios. To further improve robustness, we introduce a Self-Attention-based Graph Convolutional Network (SA-GCN) to rank neighboring nodes within the syntactic graph, filtering out irrelevant nodes and capturing long-distance dependencies more precisely in noisy environments. Additionally, a residual shrinking network is incorporated to dynamically remove noise from the syntactic graph, further strengthening the model's noise resistance. Moreover, we propose a loss computation method based on predictive interpolation, which dynamically balances the contributions of semantic and syntactic features through weighted interpolation, thereby enhancing relation extraction accuracy. Experiments conducted on two public relation extraction datasets demonstrate that the proposed method achieves significant improvements in accuracy, particularly in handling long-distance dependencies and noise suppression.

**Keywords** Multi-Relational extraction, Dependency syntax graph, Attention-Based ranking, Residual shrinking, Predictive interpolation, Pre-Trained model

Relation extraction aims to classify relationships between entities in a given text<sup>1</sup>, enabling effective information mining and supporting downstream tasks such as knowledge graph construction, question answering systems, and search engines. Existing relation extraction methods can be broadly categorized into two approaches: sequence-based classification and dependency-based classification<sup>2</sup>. Sequence-based extraction methods primarily focus on predicting relationships using vector transformations of word-level information within sentences. Representative models include Convolutional Neural Networks (CNN)<sup>3</sup>, Long Short-Term Memory Networks (LSTM)<sup>4</sup>, and pre-trained models such as BERT and GPT<sup>5,6</sup>. Although these character-level methods can capture relationships between entities and sentences to some extent, they struggle with long-distance dependencies. As sentence length increases, relying solely on entity vector representations makes it difficult to accurately capture dependencies between distant entities. To address this issue, additional features such as radicals or syntactic information are often incorporated to enrich the representation of inter-entity relationships<sup>7</sup>. Dependency trees serve as a structured visualization of sentence semantics, providing additional features beyond word embeddings. They enable direct interactions between distant entities in a sentence, thereby improving relation extraction performance<sup>8,9</sup>.

However, directly incorporating dependency syntax graphs as supplementary features introduces challenges. Since these graphs are generated using external parsers, they often contain noisy information that may hinder effective entity interactions<sup>10</sup>. The sensitivity to noise increases as entity distance grows, negatively impacting classification accuracy. To mitigate this issue, fixed pruning strategies or subgraph sampling techniques are

<sup>1</sup>School of Computer Science, Northeast Electric Power University, Jilin 132012, China. <sup>2</sup>Jilin Institute of Chemical Technology, Jilin 132022, China. ✉email: 2202201032@neepu.edu.cn

commonly used, such as extracting the shortest path between given entities<sup>11</sup> or pruning dependency trees<sup>12</sup>. However, such static pruning and subgraph sampling approaches fail to dynamically adapt to sentence structures, leading to residual noise that affects long-distance entity interactions. To further improve relation extraction, this paper proposes a novel method that integrates semantic and syntactic features to handle noisy long-distance dependencies dynamically. Specifically, we introduce an adaptive pruning strategy for dependency syntax graphs and apply soft-threshold filtering to entity vector representations. This approach enhances long-distance dependency modeling while reducing noise from syntactic information, ultimately achieving superior relation extraction performance compared to baseline models.

The main contributions of this paper are as follows:

*Long-distance dependency capture with SA-GCN:* We propose a Self-Attention-based Graph Convolutional Network (SA-GCN) to dynamically update node representations within the dependency syntax graph. By leveraging a self-attention mechanism, our method effectively removes irrelevant nodes, enabling more precise long-distance dependency modeling in noisy environments.

*Noise suppression with residual-dependent syntax graphs:* We introduce a noise suppression mechanism based on a residual shrinking network, which retains crucial information within the dependency syntax graph while filtering out irrelevant features. This dynamic noise removal process enhances the extraction of meaningful information and improves model robustness against noisy input.

*Relation extraction optimization via predictive interpolation:* We propose a predictive interpolation-based optimization strategy for integrating semantic and syntactic features. By combining contextual semantic representations from BERT with syntactic features from the dependency syntax graph, our approach achieves a deep synergy between the two information sources. Additionally, during training, a predictive interpolation loss function facilitates effective interaction between BERT and SA-GCN, significantly enhancing the model's ability to capture long-distance dependencies.

## Related work

Relation extraction methods can generally be categorized into two main approaches: sequence-based relation extraction and dependency-based extraction methods. Sequence-based relation extraction methods classify relationships between entities by leveraging word vector representations within sentences. Zeng et al.<sup>13</sup> were the first to employ Convolutional Neural Networks (CNN) combined with entity position markers to extract word and token embeddings from sentences, followed by a relation classifier to determine entity relationships. Li et al.<sup>14</sup> proposed an entity-aware attention mechanism integrated with a Long Short-Term Memory network (LSTM) to capture word representations, constructing latent entity-type vectors to encode contextual information and classify relations. Nathani et al.<sup>15</sup> introduced an attention-based feature embedding approach using CNNs, capturing entity and relation features from neighboring entities and performing link prediction between missing entities. Although these methods achieve reasonable performance in relation classification, CNN- and LSTM-based models struggle to extract deep semantic features from sentences effectively. With the emergence of pre-trained models such as BERT<sup>16,17</sup>, these models have been widely adopted in various natural language processing (NLP) tasks. Wu et al.<sup>18</sup> introduced the R-BERT model, which utilizes pre-trained embeddings to encode sentences while marking entity positions with special tokens, followed by classification for relation prediction. Hou et al.<sup>19</sup> combined BERT with CNN for relation extraction in specialized domains, where BERT encodes word embeddings, and CNN extracts multi-scale features via pooling. However, simple pooling operations such as summation or averaging can result in information loss. Xu et al.<sup>20</sup> observed that most existing models rely on neural network architectures but overlook the impact of key phrases on relation extraction. To address this, they proposed a BERT-based gated multi-window attention network (BERT-GMAN), which extracts sentence-level semantic features from BERT, builds a key phrase semantic network for multi-granularity phrase information, and applies feature fusion before classification. Shi et al.<sup>21</sup> further enhanced BERT-based extraction by integrating lexical and syntactic features, such as part-of-speech tags and dependency trees, achieving superior extraction performance.

However, sequence-based methods heavily rely on word vector representations while neglecting the influence of syntactic dependencies within a sentence. Dependency-based extraction methods incorporate sentence- and syntax-level features to enrich entity representations. Since sentence dependencies can be naturally represented as graphs, these methods are often combined with Graph Neural Networks (GNNs). Hao et al.<sup>22</sup> proposed a generative relation extraction approach that constructs dependency graphs between entities and utilizes GNNs to propagate information, enabling not only relation extraction but also multi-hop link prediction. However, the constructed dependency graphs may contain irrelevant information, which can negatively impact information propagation and reduce classification accuracy. To address this issue and improve relation extraction accuracy, Tian et al.<sup>23</sup> introduced a dependency-driven method that employs an attention mechanism within GNNs to distinguish the importance of different word dependencies. Guo et al.<sup>24</sup> proposed an attention-guided graph convolutional network (AGGCN) that applies soft pruning to automatically focus on relevant structures in relation extraction tasks, effectively filtering out irrelevant information. Xue et al.<sup>25</sup> took a different approach by eliminating reliance on external parsers. They used a Gaussian generator to construct multi-view graphs directly from raw text, refining the graph structure through interaction between graph convolutions and DTWPool before final relation classification.

Although these methods refine dependency trees by assigning attention weights to different nodes, a major limitation is that they may still distort the information of entity nodes. In this work, we fully integrate sentence-level semantic information with dependency syntax trees by leveraging a Self-Attention-based Graph Convolutional Network (SA-GCN) to process entire dependency graphs. This approach enhances long-distance dependency modeling while utilizing a residual shrinking mechanism to improve noise robustness. By effectively fusing semantic and syntactic information, our method achieves superior relation extraction performance.

## Methods

This paper proposes a noisy long-distance dependency relation extraction method integrating semantic and syntactic features to address the challenges posed by long-distance dependencies and noise interference in entity relations. The proposed method consists of three core components: (1) contextual semantic feature extraction based on BERT, (2) long-distance dependency modeling incorporating syntactic features, and (3) relation extraction by integrating semantic and syntactic representations. The overall framework is illustrated in Fig. 1.

First, the input text is encoded using a pre-trained model (BERT), transforming it into corresponding token embeddings. Simultaneously, entity vectors are extracted based on entity index mappings. Then, a dependency syntax graph is constructed from both the textual input and an external parser, where nodes in the graph are derived from the BERT-encoded token embeddings. A self-attention-based graph convolutional network (SA-GCN) is employed to capture long-distance dependencies between entities effectively. Additionally, a residual shrinking network is introduced to dynamically suppress noise in the syntactic graph, further enhancing the model's robustness against noisy dependencies. Finally, the integrated semantic and syntactic feature vectors are passed through a dual multi-layer perceptron (MLP) to construct a relation prediction matrix. The final entity relation output is obtained using an activation function combined with a masking mechanism, improving the accuracy and robustness of relation extraction.

### Contextual semantic feature extraction based on BERT

This paper uses large-scale pre-trained models to extract the contextual semantic representations of entities in the text. The input data requires not only the text  $X = \{x_1, x_2, x_3, \dots, x_n\}$ , but also the entity position indexes  $index = \{(i_1, j_1) \dots (i_k, j_k)\}$ , where  $i_k, j_k \in \{1, 2, \dots, n\}, i_k \leq j_k$  represent the head and tail indices of the  $k$ -th entity in the text. The vectorization process involves encoding batch text data using a pre-trained BERT model to obtain output  $N \in \mathbb{R}^{b \times n \times 768}$ . Then, leveraging the entity position index information  $\{i, j\} \in (0, n)$ , an averaging operation is performed to obtain the vectorized representation of the entity  $E \in \mathbb{R}^{b \times k \times 768}$ . Equation (1) represents the expression of an entity vector  $e_k$  in  $E$ .

$$e_k = \frac{\sum_{(i_k, j_k) \in index} \text{extract}(N; < i_k, j_k >)}{j_k - i_k} \quad (1)$$

Where  $b$  represents the batch size.  $e_k \in \mathbb{R}^{1 \times 768}$  represents the entity vector indexed by  $k$ ;  $i$  and  $j$  represent the head and tail positions of the entity, respectively; and  $\text{extract}$  denotes the matrix slice taken at the specified positions.

### Long distance dependency relationship capture considering syntactic features

Compared to traditional vector extraction methods, the dependency syntax graph can significantly reflect the internal structure and syntactic information of a sentence. However, some irrelevant nodes (such as commas and periods) can introduce irrelevant information between entities during the message-passing process of the graph neural network, making it difficult for entity nodes to learn key features. To reduce the error propagation caused by reliance on the parser and enhance long-distance dependencies between entities, this paper proposes the use of a self-attention-based graph convolutional network (SA-SAG) after constructing the syntax graph. By combining the attention mechanism with convolutional operations, this approach sorts node scores and removes the influence of irrelevant nodes, effectively capturing long-distance dependencies between entities.

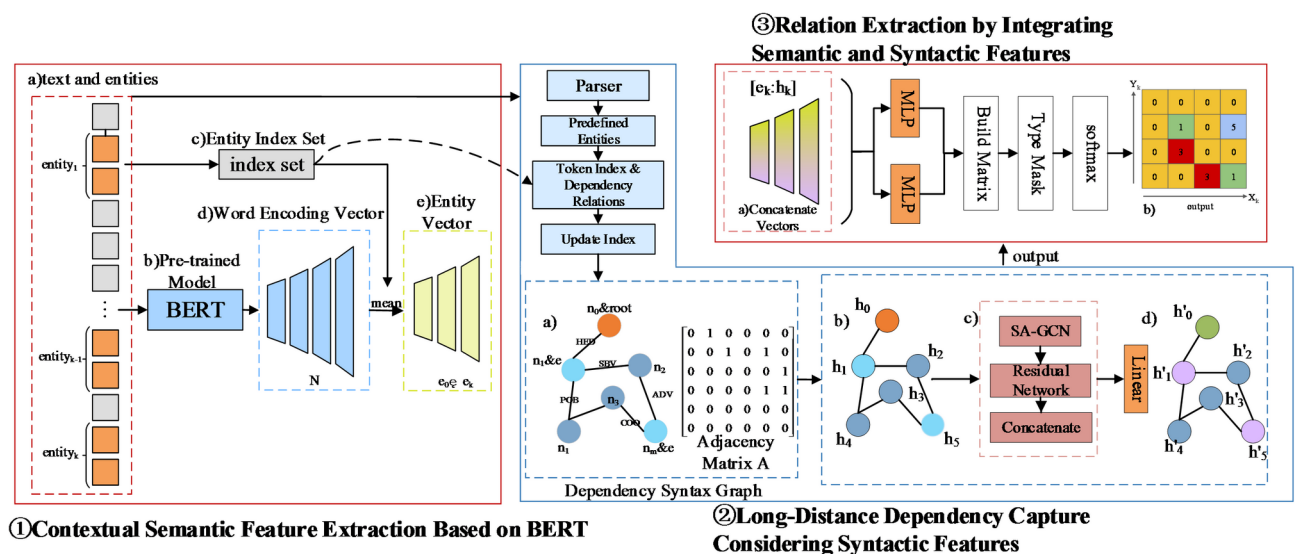


Fig. 1. Method Flowchart.

### Construction of text dependency syntax graph based on external parser

The dependency syntax graph<sup>27</sup> represents the dependency relationships between words in a natural language sentence, with each token being represented as a node and connected by directed edges (such as subject-predicate relations, verb-object relations, etc.). As a graphical structure, the dependency graph contains rich semantic and syntactic information. In this paper, the dependency syntax graph of the text is constructed using an external parser, with the aim of providing more syntactic features for the relationships between entities within the sentence, and effectively leveraging structural information to offer more entity-level semantic relations.

The parser automatically infers the dependency relationships between words based on syntactic rules and contextual information, thereby generating the corresponding dependency syntax graph<sup>28,29</sup>. The process of constructing the dependency syntax graph using a parser is illustrated in Fig. 2. First, entity information is input into an external parser, and tokenization rules are modified to prevent inconsistencies between the default tokenization results of the external parser and the entity information. Then, the text is tokenized by the external parser, obtaining the tokenization index *cws* and dependency relationships *dep*. The tokenization index and semantic vector *N* are then used to generate the vector representations of the dependency syntax graph nodes. Meanwhile, the dependency relationships *dep* are utilized to construct the corresponding adjacency matrix.

Specifically, for text  $X = \{x_1, x_2, x_3 \dots, x_n\}$ , first, the parser's tokenization rules are adjusted based on the entities in the text. Then, the tokenization and dependency parsing results are obtained accordingly, as shown in the following equation:

$$\begin{cases} \text{parser} = \text{special\_case}(\text{parser}, \text{entity}) \\ \text{cws}, \text{dep} = \text{parser}(X) \end{cases} \quad (2)$$

Where, *entity* represents the entities in the text; *special\_case* denotes the modified tokenization function; *cws* refers to the tokenization results; *dep* refers to the dependency relations; *parser* represents the external parser. To construct the dependency syntax graph, the tokenization result *cws* is combined with the word vector *N* to obtain the vector representation of the syntax graph nodes. The vector representation of the *i*-th node in the graph is given by Eq. (3).

$$H^i = \begin{cases} \text{mean}([N; < \text{cws}[i]_a, \text{cws}[i]_b >]), & \text{if } i \neq 0 \\ N_{[\text{CLS}]} , & \text{if } i = 0 \end{cases} \quad (3)$$

Where, *N* represents the vector generated by BERT; *cws*[*i*]<sub>*a*</sub> and *cws*[*i*]<sub>*b*</sub> denote the start and end indices of the *i*-th token in the text; *H<sup>i</sup>* denotes the embedding vector of the *i*-th node in the syntax graph; *N*<sub>[CLS]</sub> represents the CLS vector output by BERT, it is mapped to the “root” node of the external parser's output; *mean*([*N*; < *u*, *v* >]) represents the summation of the *u*-th vector to the *v*-th vector in *N*, followed by averaging.

The adjacency matrix in the syntactic graph is transformed through the dependencies of the dependency relations, and can be expressed as Eq. (4):

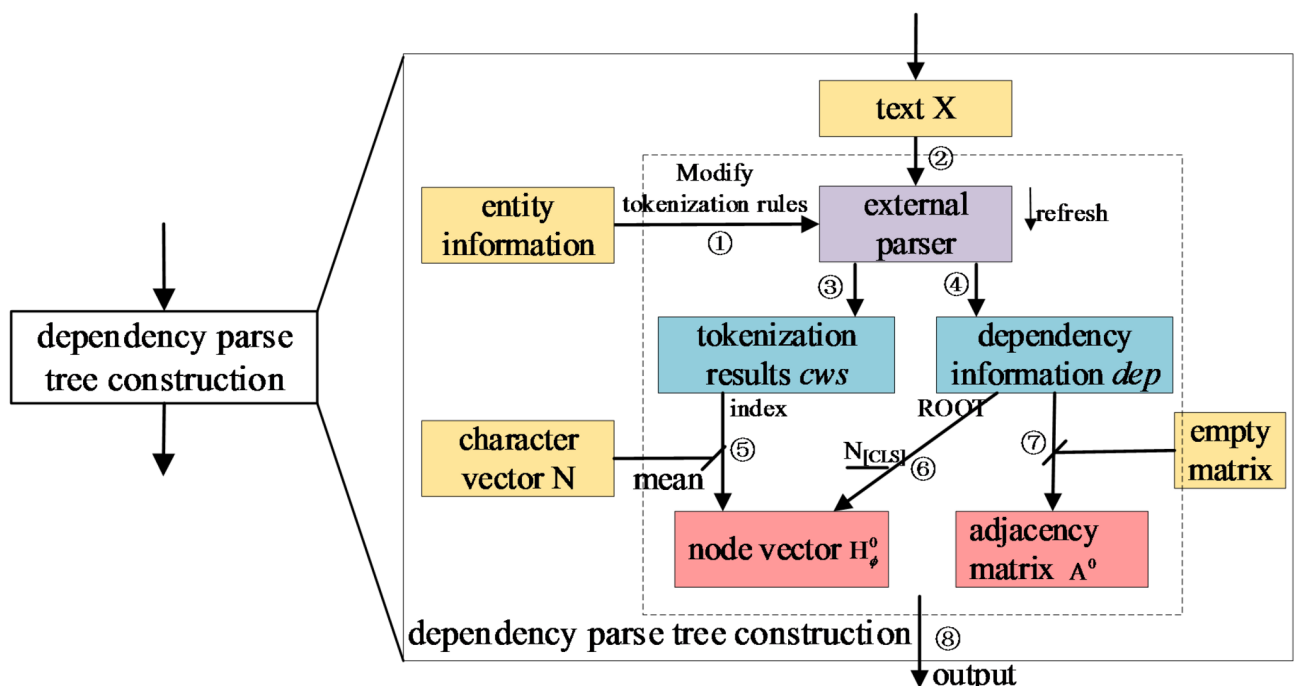


Fig. 2. Construction of Dependency Syntax Graph.

$$A_{\alpha,\beta} = \begin{cases} 1, & \text{if } \text{dep}[\alpha] = \beta \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

Where,  $\text{dep}[\alpha]$  represents the dependent parent node of node  $\alpha$ ;  $A_{\alpha,\beta}$  represents the value of row  $\alpha$  and column  $\beta$  of the adjacency matrix.

### Long distance dependency relationship capture method based on SA-GCN

GCN, as a graph-structured deep learning model<sup>30</sup>, it is capable of preserving the topological structure and node characteristics within a graph, integrating both node and structural information. The core idea of GCN is to update a node's representation using information from its neighboring nodes. The node representation is updated by aggregating the features of its neighboring nodes. The formula for the traditional GCN is shown in Eq. (5) as follows:

$$H_{\phi}^{l+1} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H_{\phi}^l W^{l+1} \quad (5)$$

Where,  $H_{\phi}^l \in \mathbb{R}^{n \times d}$  represents the node vector at layer  $l$ ;  $\tilde{D} \in \mathbb{R}^{n \times n}$  is the degree matrix of  $\tilde{A} \in \{0, 1\}^{n \times n}$ ; and  $W^{l+1}$  represents the weight matrix used to generate the node embeddings for the  $l + 1$  layer.

To guide the model in removing the propagation of irrelevant node information, this paper proposes a self-attention graph convolutional network (SA-GCN) that assigns a weight to each node, capturing the long-distance dependency relationships between entities. As shown in Fig. 3, the process begins by averaging each node vector generated by the BERT pre-trained model to obtain the input vector for the module. Then, the node representations of the dependency graph and the adjacency matrix are fed into the graph convolutional network. They are multiplied by a learnable attention parameter matrix  $M_1$  to obtain a node importance matrix that integrates both node and structural information. This is represented by the following Eq. (6):

$$\begin{cases} H_{\phi}^1 = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H_{\phi}^0 W_1 \\ \text{score}_1 = \tanh(H_{\phi}^1 \times M_1) \end{cases} \quad (6)$$

Where,  $A$  is the adjacency matrix of the original dependency syntactic graph;  $\tilde{A} = A + I$  is the adjacency matrix with self-loops added;  $W_1$  represents the learnable weight matrix of the first layer;  $I \in \{0, 1\}^{n \times n}$  is the

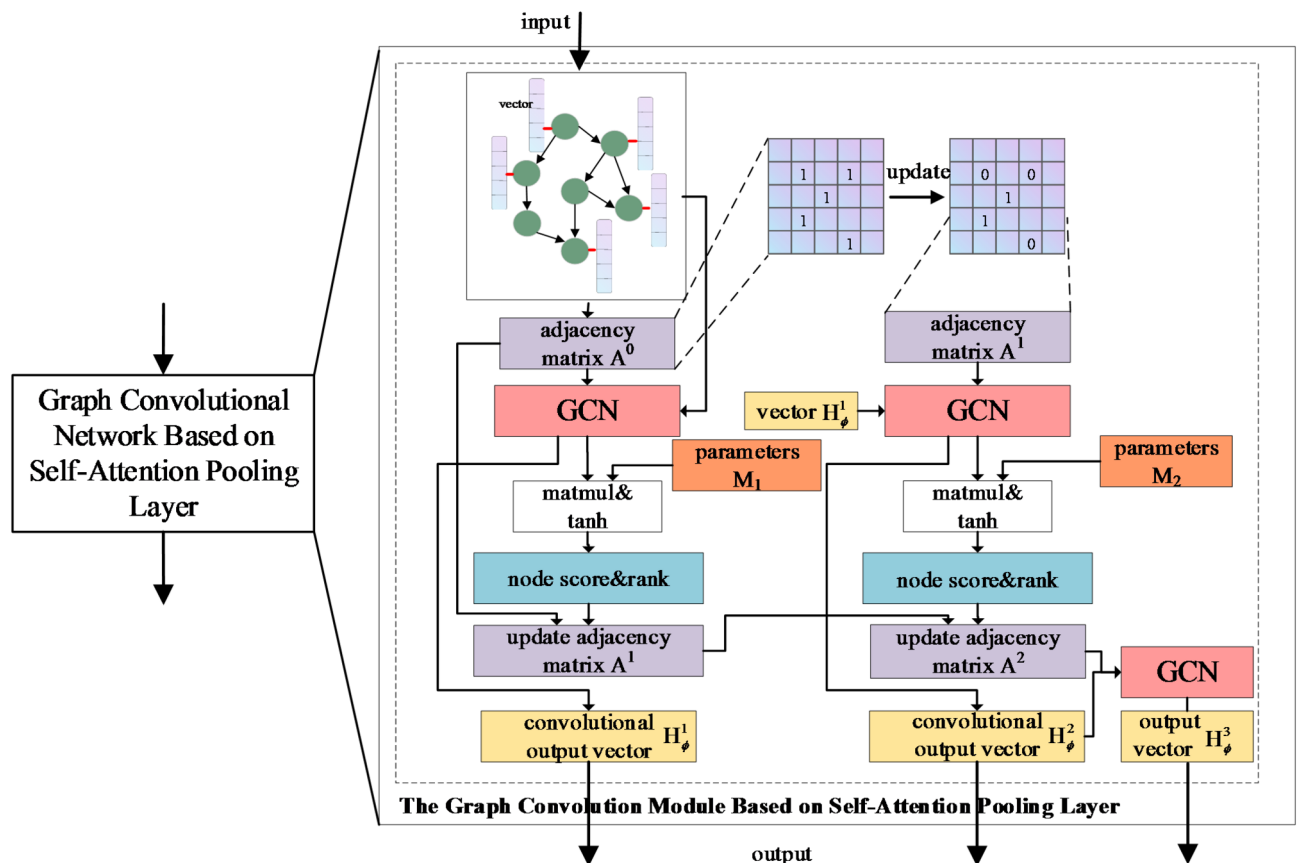


Fig. 3. Graph Convolution Module Based on Self-Attention Pooling Layer.



identity matrix;  $\tanh$  is the activation function;  $M_1 \in \mathbb{R}^{d \times 1}$  is the attention matrix;  $score_1 \in \mathbb{R}^{n \times 1}$  is the node importance score of the first layer.

To discard irrelevant node features and enhance the feature representation of entity nodes, after sorting the sequences  $score_1$ , the rows and columns of the adjacency matrix  $A^0$  corresponding to nodes with lower scores are set to 0. This reduces the influence of these nodes during global message passing in the next layer. The specific process is shown in Eq. (7) as follows:

$$\begin{cases} ID = \text{top\_rank}(score, [k \cdot n_1]) \\ A^1 = \text{change}(A^0, ID) \end{cases} \quad (7)$$

Where,  $\text{top\_rank}$  represents the sorting function;  $k$  denotes the ratio of the current layer's nodes to those in the previous layer;  $n_1$  represents the number of nodes in the maximal connected subgraph after the first update of the adjacency matrix;  $\text{change}$  is the adjacency matrix transformation function;  $A^1$  represents the updated adjacency matrix after the first round of graph message passing.

Similarly, the node embeddings output after the second round of message passing and the updated adjacency matrix are shown in Eq. (8) as follows:

$$\begin{cases} H_\phi^2 = \tilde{D}^{-\frac{1}{2}} \tilde{A}^1 \tilde{D}^{-\frac{1}{2}} H_\phi^1 W_2 \\ score_2 = \tanh(H_\phi^2 \times M_2) \\ A^2 = \text{change}(A^1, \text{top\_rank}(score_2, [k, n_2])) \end{cases} \quad (8)$$

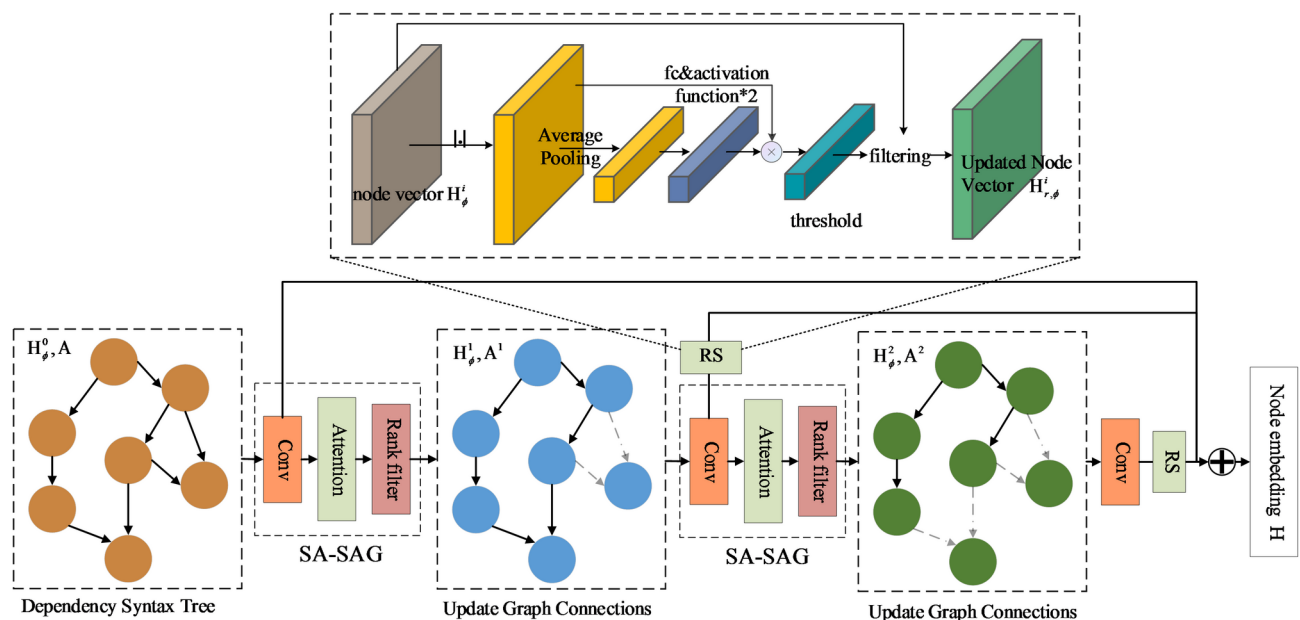
Finally, the node output vectors after the third round of message passing are shown in Eq. (9) as follows:

$$H_\phi^3 = \tilde{D}^{-\frac{1}{2}} \tilde{A}^2 \tilde{D}^{-\frac{1}{2}} H_\phi^2 W_3 \quad (9)$$

Where,  $H_\phi^2, H_\phi^3$  represents the output results after the 2nd and 3rd passes through the GCN network,  $M_2$  represents the attention parameter matrix,  $W_2$  and  $W_3$  represents the parameter matrices of the 2nd and 3rd GCN networks, and  $n_2$  represents the number of maximal connected subgraphs after the second message passing.

### Noise suppression method based on residual-dependency syntax graph

After the graph convolution, self-attention, and sorting operations, the filtered graph structure for each layer can be obtained, as shown in Fig. 4. The figure illustrates how the residual shrinkage network is applied in graph message passing. To suppress the influence of irrelevant node noise during graph message passing, this paper introduces a Residual Shrinkage (RS) network and proposes a noise suppression method based on a residual-dependency syntax graph. The RS network is embedded into the graph message passing process, with the output of each convolution stage being input to the network. By using a learnable threshold, the internal node vectors are subjected to soft thresholding, dynamically removing noise interference in the syntax graph, which enhances the extraction of valid information.



**Fig. 4.** Residual-Dependency Syntax Graph Noise Suppression Method Flowchart.

For the residual network, the node vectors are first processed by taking their absolute values to make the features positive. After that, a global average pooling operation is performed to obtain the average value of each node vector. Next, two fully connected layers along with corresponding activation functions are used to learn the threshold features, which are then mapped to the corresponding dimensional vector, resulting in the threshold feature matrix  $x$ , as represented by formula (10):

$$x_i = \text{sigmoid}(W_\delta(\text{relu}(W_\varepsilon(\text{Pooling}(|H_\phi^i|) + c_\varepsilon) + c_\theta)) \quad (10)$$

Where  $H_\phi^i \in \mathbb{R}^{n \times d}$  is the node vector after convolution; Pooling is the global average pooling function;  $W \in \mathbb{R}^{1 \times 1}$  and  $c \in \mathbb{R}^{1 \times 1}$  represent the learnable parameter matrices and bias of the linear layers, respectively; sigmoid and relu are the activation functions.

To prevent the issue of excessively large thresholds, the vector  $x_i$  is multiplied element-wise with the absolute value of vector  $H_\phi^i$  to obtain the final set of thresholds  $\delta$ . Then, the original node vector undergoes soft threshold, as shown in Eq. (11):

$$h_{r,\phi}^i = \begin{cases} h_\phi^i - \delta, & h_\phi^i > \delta \\ 0, & -\delta \leq h_\phi^i \leq \delta \\ h_\phi^i + \delta, & h_\phi^i < -\delta \end{cases} \quad (11)$$

Where,  $h_\phi^i$  and  $h_{r,\phi}^i$  represent the value of node vectors before and after the update within  $H_\phi^i$ , respectively.

The following pseudocode demonstrates an example of the batch-processing residual-dependency syntax graph noise suppression method:

Noise Suppression Method Based on Residual-Dependency Syntax Graph
<i>Input:</i> In batch processing, sentence $X = \{X_1, X_2, \dots, X_b\}$ ; each sentence corresponds to word vector $\{E_1, E_2, \dots, E_b\}$ ;
<i>Output:</i> In batch processing, node vector $H$ . Initialize the dependency syntax graph
for $X_i$ in $X$ do $H^i, A^i \leftarrow (\text{externalParser}(X_i), E_i)$ #Construct the node embeddings and A for each sentence.
$H_\phi^0 \leftarrow H^i, A^0 \leftarrow A^i$ # Construct the dependency syntax graph for each batch.
end for
for $m$ in $\text{conv\_num}$ do #conv_num = {1, 2, 3}
$H_\phi^m, A^m = \text{Self Attention Graph Pooling}(H_\phi^{m-1}, A^{m-1})$ # Graph Convolution Layer if $m \neq 1$ then
$H_{r,\phi}^m = \text{Residual Shrinkage Network}(H_\phi^m)$ # Residual Shrinking
end for
$H = H_\phi^1 + H_{r,\phi}^2 + H_{r,\phi}^3$ # Vector concatenation

## Syntax-semantic feature fusion for relation extraction

For each entity pair in the sentence, the entity's node vector is found in the dependency syntax graph using the id index, and then concatenated with the BERT word embedding vector to obtain the final entity vector representation  $v_{a,b} = [e_{a,b}; h_{a,b}]$ . Where  $v_{a,b}$  represents the feature vector of the entities with indices  $a$  and  $b$ ,  $h_{a,b}$  and  $e_{a,b}$  represent the node vector and word embedding vector, respectively. Then, a two-layer perceptron is used to map the entity vectors  $v_{a,b}$ , and the feature information between entities is obtained by concatenating entity pairs. A linear layer is then applied to map the entity feature information to relation classification. The specific calculation method is as follows:

$$M_{a,b} = \text{softmax}(W \cdot (\text{MLP}_1(v_{a,b}) * \text{MLP}_2(v_{a,b})) + c) \quad (12)$$

Where MLP represents the multi-layer perceptron;  $*$  denotes full concatenation between vectors;  $M_{a,b}$  represents the entity relationship classification value within the decoding matrix;  $W \in \mathbb{R}^{d \times n}$  and  $c \in \mathbb{R}^{d \times n}$  are the parameters of the linear layers.

Due to the inherent structural differences between the BERT model and GCN, directly combining them may lead to incomplete convergence, thereby affecting the accuracy of model extraction. The output of the BERT model is typically high-dimensional vectors, while the input of GNN is usually low-dimensional node features. This dimensionality mismatch can result in a sharp increase in computational complexity and may introduce redundant information. Additionally, the optimization objectives of BERT and GCN are inconsistent: BERT focuses on capturing global semantic information, whereas GNN emphasizes local structural information. This objective conflict further impacts the convergence of the model. In this paper, we employ an interpolation prediction method to enable the pre-trained model and the graph neural network to update simultaneously. Specifically, the output vectors  $N$  from the BERT model are divided into two purposes: one is used as input to the graph neural network, and after passing through the graph neural network, the vectors are combined with classification results to calculate the cross-entropy loss with the true labels. The other is used for independent prediction, where node vectors  $E$  are extracted directly from  $N$ , classified, and then used to calculate the cross-entropy loss with the true labels. The weights between the two parts are connected by a weighting coefficient, as shown in Eq. (13):

$$loss = w \cdot CEL_{BERT-GCN}(p_1, l) + (1 - w)CEL_{BERT}(p_2, l) \tag{13}$$

In this case,  $w$  represents the weight coefficient, when  $w = 1$ , it indicates that the interpolation prediction method is not used; CEL denotes the cross-entropy loss function;  $p_1$  and  $p_2$  are the prediction values output by the GCN and BERT, respectively; and  $l$  is the ground truth label.

At the same time, the model faces an issue of class imbalance when outputting label matrices, as the relationships between text entities are rare, resulting in a large number of label matrices but very few positive samples. To address this issue, we introduce entity types to restrict the relationship classification between entities. By predefining entity type relationships, we set a masking mechanism that enables the model to compute the loss value for specific entity relationships and specify classification vectors for unrelated entities. This is expressed as in formula (14).

$$\begin{cases} id\_mask = R < entity_1, entity_2 > \\ CEL_{BERT}(p_2, l) = CEL(p_2 \cdot id\_mask, l \cdot id\_mask) \\ CEL_{BERT-GCN}(p_1, l) = CEL(p_1 \cdot id\_mask, l \cdot id\_mask) \end{cases} \tag{14}$$

In this context,  $id\_mask$  represents the mask between entities,  $R$  is the function used to judge the predefined entity types, and  $entity$  refers to the entity categories.

Experiments and analysis  
Datasets

Articles The experiment uses two datasets for validation:1) Baidu DUIE 2.0 Relation Extraction Dataset<sup>32</sup>, which includes over 30,000 text samples, 130,000 triplet data, and 31 predefined relation types, with “unknown” representing no relationship between entities. 2) SemEval 2010 Task 8 Dataset<sup>33</sup>, consisting of 8,000 training samples and 2,717 test samples, containing 10 relation types, with “Other” indicating no relationship between entities.

Experimental setup and metrics

This paper uses the Ubuntu 18 operating system, Python 3.8 environment, and the PyTorch 1.9.0+cu111 framework. The GPU used is an A100-80G. The model parameters are set as follows: the number of iterations (epochs) are 20 and 30, the batch size is 32, the learning rate is 5e-3, the input text length limit is set to 256, the number of GCN layers is 3, the irrelevant node coefficient for SAG is 0.2, and the coefficient for interpolation prediction is 0.8. The pre-trained model used is bert-base-chinese, with a hidden vector size of 768 and 12 layers of Transformers. The experimental metrics are precision (Precise, P), recall (Recall, R), and F1 score, which are used to evaluate the performance of relation extraction.

Comparison and analysis of different models

To verify the effectiveness of the proposed method, the following baseline methods are selected for comparison:

R-BERT<sup>17</sup>: This method directly uses the position indices of entities in the sentence, utilizes BERT to extract entity vectors, and aggregates the CLS vector for relation classification.

BERT-LSTM<sup>21</sup>: Based on BERT, this method uses LSTM’s sequential characteristics to extract deeper features, aggregating vector information for classification.

BERT-GAT<sup>34</sup>: This approach constructs a graph based on entities and allocates different weights to the edges between entities using an attention mechanism. It then uses graph propagation to update entity vectors and finally decodes the relationship between the two entities.

GP-GNN<sup>22</sup>: This method uses Glove to extract vectors and combines them with positional information in an LSTM model to obtain entity vectors. It constructs a fully connected graph structure about entities and predicts the relationship between entities.

AGGCN<sup>24</sup>: This method converts the original dependency tree into a fully connected weighted graph. Based on GCN, it learns node representations using the correlation strength between nodes, ultimately obtaining the relationship between entities.

Based on the above methods, this paper tests the model’s ability for relation extraction on two datasets using F1 score, accuracy, and recall curves, as shown in Table 1.

Validation Model	DUIE			SemEval 2010		
	F1(%)	R(%)	P(%)	F1(%)	R(%)	P(%)
R-BERT	92.89	92.98	92.88	87.06	87.15	87.05
BERT-LSTM	92.28	92.57	92.47	87.47	87.56	87.54
BERT-GAT	93.19	93.19	93.22	87.17	87.28	87.14
GP-GNN	89.6	90.59	88.85	83.25	80.72	77.2
AGGCN	91.87	91.32	92.59	84.48	85.92	83.1
ours	93.25	93.27	93.29	88.41	88.46	88.44

Table 1. Comparison results.



Experimental results indicate that the proposed model outperforms most existing models in relation extraction tasks, achieving overall performances of 93.28% and 88.41% on the DUIE 2.0 and SemEval2010 Task 8 datasets, respectively, demonstrating the effectiveness of the model. Models such as R-BERT and BERT-LSTM, which all utilize BERT pre-trained models for encoding, apply different operations to the encoded vectors for recognition. However, the use of single word vectors fails to fully capture the interactions between entities, missing out on the analysis of other important features. Models like BERT-GAT, which incorporate graph neural networks, focus on the interaction between entities after encoding but lack guidance from the sentence's inherent syntax and semantic structures. In contrast, our model is based on BERT encoding and uses attention mechanisms combined with a residual shrinkage network to fully integrate dependency syntax graph information, thus enhancing the performance of relation extraction.

### The impact of text length on dependency relations

To visually demonstrate how the proposed model enhances the ability to capture entity dependency relationships as text length increases, Fig. 5 compares BERT without a graph neural network and our model across different sentence lengths on two datasets. In the upper part, the bar charts with F1, P, and R metrics use solid bars to represent the model without a graph neural network, while the counterparts indicate the proposed model. In the lower part, the line charts depict the variation of F1 scores with text length for each dataset: black lines represent the model without a graph neural network, whereas red lines represent our model. In the DUIE dataset, it is observed that for sentence lengths within the range of (0, 50), the metrics show little difference. For sentence lengths in the range of (50, 100), the metrics show a slight increase. However, for sentence lengths in the range of (100–256), the F1, Precision (P), and Recall (R) metrics improve by 1.34%, 1.7%, and 0.98%, respectively. In the SemEval 2010 dataset, for sentence lengths within three ranges, the F1 score significantly exceeds that of the previous model, with improvements of 0.19%, 1.11%, and 2.17%, respectively. This indicates that as the sentence length increases, the proposed model shows a more significant improvement in relation extraction performance. This demonstrates that by incorporating graph neural networks based on dependency syntax graphs, the model enhances the dependency relations between entities, leading to improved performance in relation extraction tasks.

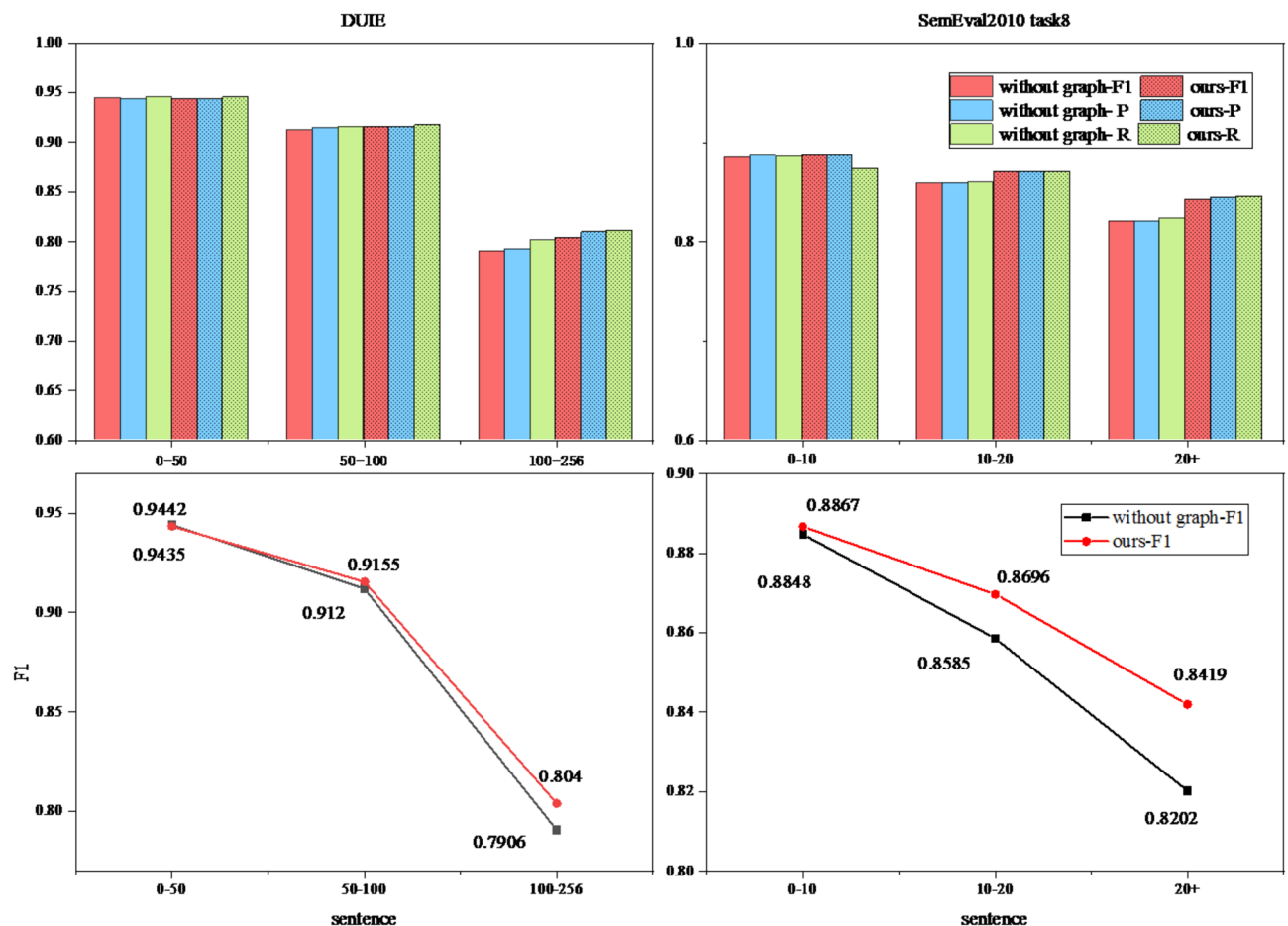


Fig. 5. The three metrics of the model at different sentence lengths.

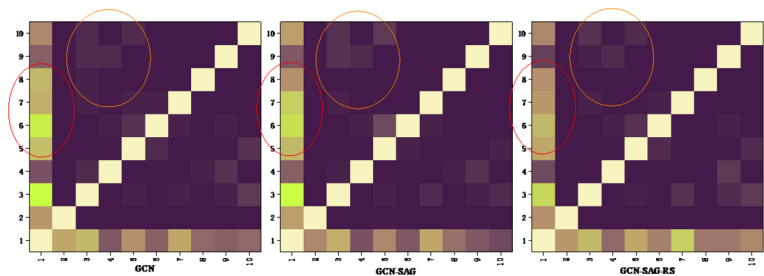


Fig. 6. Three confusion matrices.

Test Set	DUIE		SemEval 2010	
	F1(%)	Δ(%)	F1(%)	Δ(%)
BERT	93.08	–	87.31	–
GCN	92.69	-0.39	88.15	0.16
SAG	92.81	0.12	88.26	0.11
RS	93.28	0.23	88.41	0.15

Table 2. Ablation study.

Model noise suppression analysis

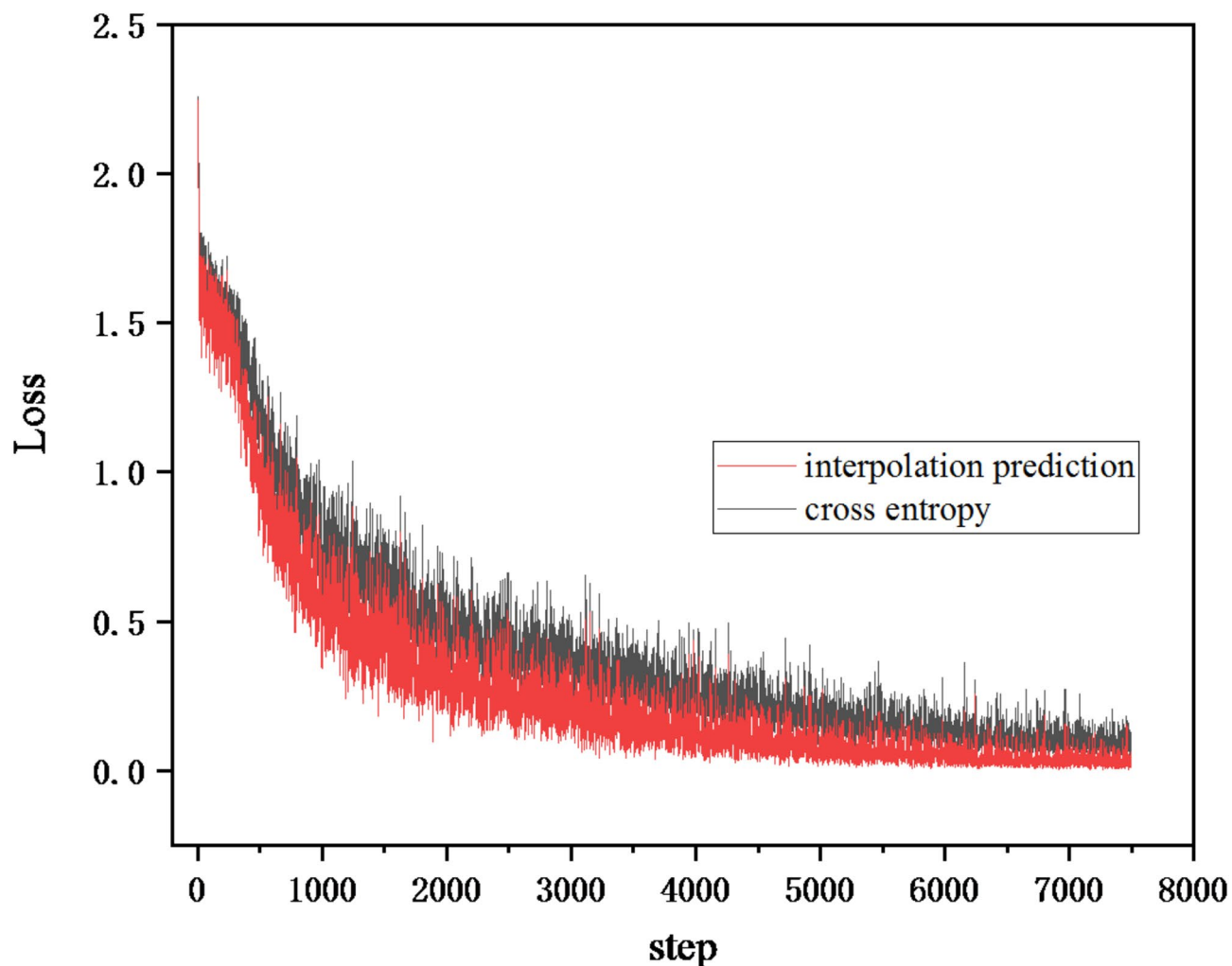
The method proposed in this paper primarily reduces the propagation of irrelevant node noise based on constructing a dependency syntax graph. To visually demonstrate the impact of adding the SAG and RS modules on the model’s denoising effect, we conduct denoising visualization on the SemEval2010 task8 dataset. First, we train the neural network without reducing noise to ignore the effect of noise on the model. Then, we sequentially add the SAG and RS modules, and use confusion matrices to more intuitively observe the impact of these modules on the overall performance.

As shown in Fig. 6, the diagonal elements of the matrix represent the correctly predicted relationship categories, while the other irrelevant elements show the strength of noise, with lighter colors indicating stronger noise. Compared to the other two confusion matrices, GCN has more noise outside the diagonal, while GCN-SAG-RS has the least noise. This indicates that by using the SAG module to learn the importance score of each node through the attention mechanism and remove useless nodes, and by using the RS module to learn soft thresholds through the residual shrinkage mechanism and update the node vectors, these two methods significantly improve the model’s performance.

Ablation study

To verify the effectiveness of the proposed method, eight ablation experiments were conducted on two datasets, with the results shown in Table 2. Each experiment added a corresponding module based on the previous one: using the BERT model for relation extraction (BERT), adding Graph Convolutional Networks (GCN), adding the Self-Attention Graph Pooling layer (SAG), and adding the Residual Shrinkage (RS) module. The experimental results show that, on some parts of the DUIE dataset, adding the dependency syntax tree resulted in a 0.39% decrease in the F1 score. This was due to excessive noise, which had a significant impact on the model’s overall performance. After applying two noise reduction methods, the F1 scores improved by 0.12% and 0.23%, respectively. On the SemEval 2010 dataset, after adding the dependency syntax tree, the model’s overall performance improved by 0.16% after GCN graph propagation. Compared to the DUIE Chinese dataset, the noise from building the syntax tree had a smaller impact on the model in the English dataset. After applying the SAG and RS modules, the F1 scores improved by 0.11% and 0.15%, respectively.

To further analyze the impact of the cross-entropy loss function and the interpolation prediction loss function on the model training process and final performance, this paper applies both loss functions (interpolation prediction loss and cross-entropy loss) to our model. To ensure the rationality of the experiment, we omit the second term in the model using interpolation prediction loss and unify the final model output as  $CEL_{BERT-GCN}(p, l)$  for comparison. The experiment is conducted on the SemEval2010 test dataset, and the results are shown in Fig. 7. The x-axis represents the training steps, while the left y-axis represents the training loss values. From the trend of the curves, it can be observed that the interpolation prediction loss outperforms the cross-entropy loss in terms of convergence speed and loss value updates. Moreover, the changes in three key evaluation metrics, as presented in Table 3, further validate the effectiveness of the interpolation prediction loss. This advantage is primarily attributed to its computational mechanism, which effectively enhances the information interaction between BERT and GCN, optimizes feature representation learning, and stabilizes parameter updates across different modules. Consequently, the model’s predictive performance is significantly improved.



**Fig. 7.** The impact of two loss functions on model performance.

Loss function	P(%)	R(%)	F1(%)
cross entropy	88.3	88.46	88.24
interpolation prediction	88.38	88.45	88.41

**Table 3.** The impact of different loss on model performance in addition, under the same experimental parameters, this paper analyzes the impact of different parameters on model performance. Table 4 shows the performance comparison of different parameters on the DUIE 2.0 dataset.

From Table 4, it can be seen that, with other parameters being the same, the learning rate has a significant impact on the model's performance. This is because a smaller learning rate makes it difficult for the model to reach optimal performance within the specified number of iterations. The text length, GCN layer count, and SAG parameter all have varying effects on the model's classification performance. The model performs best when the GCN layer count is 3 and the SAG coefficient is 0.2.

## Conclusions

This paper proposes a relation extraction model based on the BERT pre-trained model combined with an external syntactic parser and graph neural networks. By learning information from the sentence, it achieves the fusion of semantic and syntactic information, improving the accuracy of entity relationship classification. Furthermore, a graph convolutional network with a self-attention pooling layer and a residual shrinking network are employed to reduce the influence of irrelevant information in the text. Additionally, an interpolation prediction method is used to calculate the loss and refine the interaction between BERT and the graph neural network. Experimental results demonstrate significant improvements of the proposed method over the baseline models. This study highlights the potential of combining BERT with graph neural networks and integrating semantic and syntactic

Experimental parameters	P(%)	R(%)	F1(%)
Lr {5e-3,5e-4,5e-5}	{93.26,87.97,77.16}	{93.34,89.13,81.18}	{93.28,88.03,78.44}
length {64,128,256}	{94.35,93.45,93.26}	{94.45,93.32,93.34}	{94.37,93.35,93.28}
GCN layer count {2,3,4}	{93.0, 93.26,92.58}	{93.0, 93.34,92.44}	{92.97, 93.28,92.48}
Para-SAG {0.1,0.2,0.3}	{92.79,93.26,93.26}	{92.63,93.34,93.25}	{92.66,93.28,93.22}

**Table 4.** The impact of different parameters on model performance.

information to enhance relation extraction performance, providing valuable insights and directions for future research.

**Data availability**

The datasets used in this study are publicly available and can be accessed freely under open access licenses. Baidu DUIE 2.0 Relation Extraction Dataset: <https://gitee.com/open-datasets/DUIE2.0SemEval> 2010 Task 8 Dataset : <https://www.kaggle.com/datasets/cycloneboy/semEval-2010-task-8-dataset>.

Received: 21 February 2025; Accepted: 2 May 2025  
Published online: 06 May 2025

**References**

1. Wang, H. et al. Deep neural network-based relation extraction: an overview. *Neural Comput. Appl.* 1–21. (2022).

2. Wu, T. et al. Towards deep Understanding of graph convolutional networks for relation extraction. *Data Knowl. Eng.* **149**, 102265 (2024).

3. Bai, T. et al. Traditional Chinese medicine entity relation extraction based on CNN with segment attention. *Neural Comput. Appl.* 1–10. (2022).

4. Miwa, M. & Bansal, M. End-to-end relation extraction using Lstms on sequences and tree structures. *Arxiv Preprint Arxiv* 160100770 (2016).

5. Hou, J. et al. BERT-based Chinese relation extraction for public security. *IEEE Access.* **8**, 132367–132375 (2020).

6. Wan, Z. et al. Gpt-re: In-context learning for relation extraction using large Language models. *Arxiv Preprint Arxiv* 230502105 (2023).

7. Wang, H. et al. Deep neural network-based relation extraction: an overview. *Neural Comput. Appl.* 1–21. (2022).

8. Bastos, A. et al. RECON: relation extraction using knowledge graph context in a graph neural network[C]//Proceedings of the Web Conference 2021. 1673–1685. (2021).

9. Chen, G., Tian, Y. & Song, Y. Joint aspect extraction and sentiment analysis with directional graph convolutional networks. In: Proc. 28th international conference on computational linguistics. 272–279. (2020).

10. Yu, B. et al. Learning to prune dependency trees with rethinking for neural relation extraction. In: Proc. 28th international conference on computational linguistics. 3842–3852. (2020).

11. Yan, X. et al. Classifying relations via long short term memory networks along shortest dependency path. *arXiv preprint arXiv:1508.03720*, (2015).

12. Zhang, Y., Qi, P. & Manning, C. D. Graph Convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185* (2018).

13. Zeng, D. et al. Relation classification via convolutional deep neural network. In: Proc. COLING 2014, the 25th international conference on computational linguistics: technical papers. : 2335–2344. (2014).

14. Lee, J., Seo, S. & Choi, Y. S. Semantic relation classification via bidirectional Lstm networks with entity-aware attention using latent entity typing. *Symmetry* **11** (6), 785 (2019).

15. Nathani, D. et al. Learning attention-based embeddings for relation prediction in knowledge graphs. *arXiv preprint arXiv:1906.01195* (2019).

16. Devlin, J. et al. Bert: Pre-training of deep bidirectional Transformers for Language understanding. *arXiv preprint arXiv:1810.04805* (2018).

17. Wang, L. et al. A joint extraction method for fault text entity relationships in smart grid considering nested entities and complex semantics. *Energy Rep.* **11**, 6150–6159 (2024).

18. Wu, S. & He, Y. Enriching pre-trained language model with entity information for relation classification. In: Proc. 28th ACM international conference on information and knowledge management. 2361–2364. (2019).

19. Hou, J. et al. BERT-based Chinese relation extraction for public security. *IEEE Access.* **8**, 132367–132375 (2020).

20. Xu, S. et al. BERT gated multi-window attention network for relation extraction. *Neurocomputing* **492**, 516–529 (2022).

21. Shi, P. & Lin, J. Simple Bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255* (2019).

22. Zhu, H. et al. Graph neural networks with generated parameters for relation extraction. *arXiv preprint arXiv:1902.00756* (2019).

23. Tian, Y. et al. Dependency-driven relation extraction with attentive graph convolutional networks. In: Proc. 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). : 4458–4471. (2021).

24. Guo, Z., Zhang, Y. & Lu, W. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510* (2019).

25. Xue, F. et al. Gdpnet: Refining latent multi-view graph for relation extraction[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 35(16), 14194–14202. (2021).

26. Koroteyev, M. V. BERT: a review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943* (2021).

27. Jin, L. et al. Relation extraction exploiting full dependency forests. In: Proc. AAAI Conference on Artificial Intelligence. 34(05), 8034–8041. (2020).

28. Zou, Y. et al. Mining of Autism Intervention Methods based on PyLtp Package and TextRank Graph Algorithm. 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC). IEEE, : 609–613. (2020).

29. Chantapornchai, C. & Tunsakul, A. Information extraction tasks based on BERT and spacy on tourism domain. *ECTI Trans. Comput. Inform. Technol. (ECTI-CIT)*. **15** (1), 108–122 (2021).

30. Chiang, W. L. et al. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In: Proc. 25th ACM SIGKDD international conference on knowledge discovery & data mining. : 257–266. (2019).

31. Lin, Y. et al. Bertgc: transductive text classification by combining Gcn and bert. arXiv preprint arXiv:2105.05727 (2021).
32. Li, S. et al. Duie: A large-scale chinese dataset for information extraction. Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II 8. Springer International Publishing, : 791–800. (2019).
33. Hendrickx, I. et al. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. arXiv preprint arXiv:1911.10422, (2019).
34. Onan, A. Hierarchical graph-based text classification framework with contextual node embedding and BERT-based dynamic fusion. *J. King Saud University-Computer Inform. Sci.* **35** (7), 101610 (2023).

## Acknowledgements

This work was supported by the Jilin Province Science and Technology Development Plan Project. (grant 20230201067GX)

## Author contributions

L.W. and F.W. wrote the main manuscript text. X.L. and J.L. completed the experimental design and part of the experiments, while M.M. and Q.Z. were responsible for the review.

## Declarations

### Competing interests

The authors declare no competing interests.

### Additional information

**Supplementary Information** The online version contains supplementary material available at <https://doi.org/10.1038/s41598-025-00915-5>.

**Correspondence** and requests for materials should be addressed to F.W.

**Reprints and permissions information** is available at [www.nature.com/reprints](http://www.nature.com/reprints).

**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

**Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025