Received: 24 January 2024 | Revised: 19 March 2024 | Accepted: 3 April 2024 | Published online: 8 April 2024

RESEARCH ARTICLE

Journal of Data Science and Intelligent Systems
2024, Vol. 00(00) 1–9
DOI: 10.47852/bonviewJDSIS42022524

BON VIEW PUBLISHING

Domain Knowledge-Driven Relation Extraction Methods

Boxuan Chen[1] and Guan Yuan[1,2,*]

[1]*School of Computer Science and Technology, China University of Mining and Technology, China*

[2]*Engineering Research Center of Mine Digitalization, China University of Mining and Technology, China*

Abstract: Relation classification is one of the important tasks in natural language processing, which aims to determine the relationship between entities given a sentence and their respective positions. Most of the existing methods for relation classification are based on neural networks using pre-trained models such as BERT. In recent years, models based on pre-trained models like BERT have achieved excellent results in relation classification tasks in general domains. However, these methods often struggle when applied to specialized domains due to the limitations of the corpora used for BERT pre-training. Most pre-trained models are trained on text corpora from general sources like Wikipedia, which cover a wide range of domains. As a result, the content of these corpora in specific domains is limited and lacks the necessary expertise, leading to subpar performance of relation classification models in specific domains. While providing a large number of domain-specific corpora to pre-trained models could potentially address this issue, it comes with limitations such as increased computational requirements and insufficient training of specialized vocabulary. This paper proposes a method inspired by the K-BERT pre-training model to incorporate triplet knowledge from domain knowledge graphs into sentence sequences. The triplets are transformed into sentence trees and then fed into the BERT pre-trained model using absolute and relative indices. Our model has achieved an accuracy of 93.06%, which is markedly higher than that of any other baseline approach. This approach allows the incorporation of domain knowledge without significantly increasing the computational complexity. Additionally, the paper introduces a partial input method that enables the computer to understand input sentences from multiple dimensions and hierarchical levels. Experimental results on a medical domain dataset for relation classification, which includes type labels, demonstrate significant advantages over other relation classification models in terms of Accuracy.

Keywords: domain knowledge graph, pre-trained model, relation classification

1. Introduction

Extracting entities from unstructured text, identifying relationships between entities, and forming (e_1, r, e_2) triplets are fundamental problems in knowledge extraction and important steps in building a knowledge graph. This task can be divided into two sub-problems: named entity recognition [1, 2] and entity relation extraction models [3]. Among them, the main task of entity relation extraction is to identify the semantic relationship between entity pairs.

During the construction of a general knowledge graph, existing pre-trained models for entity relation extraction have become relatively comprehensive and mature, achieving good results in most general relation classification tasks. However, these models often perform poorly in specific domains. This is because pre-trained models based on BERT [4] are trained on large-scale publicly available corpora to obtain general word embeddings, and they acquire domain knowledge only through parameter fine-tuning in specific downstream tasks. As a result, the models struggle to adapt to specific domains, such as entity relation extraction tasks in domains like healthcare and finance.

In domain knowledge extraction tasks, pre-trained models such as BERT, GPT [5], and XLNet [6] can only infer the meaning of specialized terms based on the context of the input corpus data and cannot directly incorporate domain knowledge into their embeddings. To address this issue, existing research primarily relies on training pre-trained models based on BERT using domain-specific corpora to adapt them for specific domain entity relation extraction tasks. For example, Lee et al. [7] pre-trained a BERT model using a biomedical dataset and achieved excellent results in downstream tasks in the biomedical domain. Similarly, Sci-BERT [8] pre-trained on computer science and biomedical literature corpora and demonstrated good performance in the respective downstream domains. However, these pre-trained models require a substantial number of domain-specific corpora, which are often scarce or difficult to obtain publicly compared to general domain corpora. Consequently, this training approach incurs high costs and may result in lower accuracy.

In specialized domain pre-trained language models, the higher the frequency and wider the distribution of specialized terms in the corpus, the better the model's recognition performance, as shown in Figure 1. However, domain-specific corpora often exhibit characteristics such as imbalanced distribution of specialized terms, making it challenging to ensure the effectiveness of model training with limited domain corpora. By introducing domain knowledge graphs as auxiliary information for

*Corresponding author: Guan Yuan, School of Computer Science and Technology and Engineering Research Center of Mine Digitalization, China University of Mining and Technology, China. Email: yuanguan@cumt.edu.cn

entity classification, it becomes possible to accurately identify domain-specific relationship features without relying solely on domain corpora, as shown in Figure 2. Currently, representation learning for knowledge graph embedding is mostly based on the combination of word2vec [9] and transE [10], transforming relationships and entities in the knowledge graph into vectors. On the other hand, BERT-based embedding methods primarily construct vector spaces based on text. Simply combining these two representation approaches does not fully leverage their respective strengths. The main challenge lies in the fact that incorporating a set of triplets into the input text may weaken the semantic expression of the original input text. Therefore, this paper introduces a mask matrix to constrain the attention calculation for each token, avoiding semantic drift when integrating the triplet set into the original text. Experimental results validate the effectiveness of using the mask matrix. Furthermore, to address the input mismatch issue, this paper combines the triplets from the knowledge graph with the text and original input text, enabling them to collectively participate in model training to obtain unified word embedding vectors.

Figure 1
Pre-training models require the occurrence of a specialized terminology multiple times in the text for effective learning

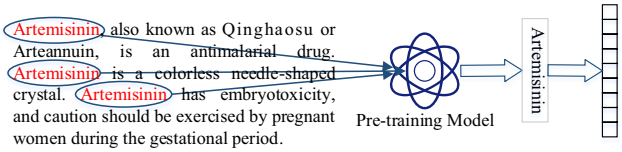
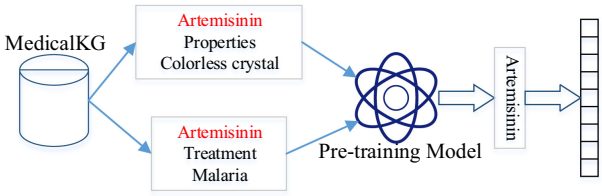


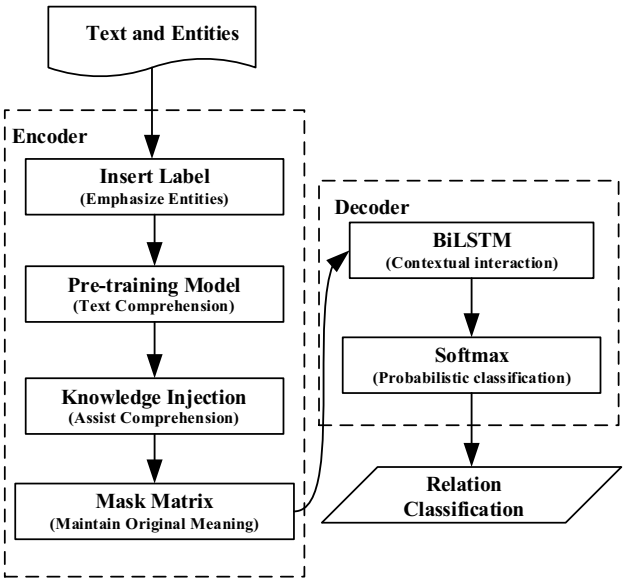
Figure 2
Incorporating triples to assist the pre-training model in understanding specialized terminology



Currently, a significant amount of work has been done to structure corpora and generate domain knowledge graphs, such as SNOMED-CT [11] in the medical field. These domain graphs contain a large number of specialized knowledge triplets that can be used for model training, greatly improving the performance of models in specific domains and reducing the computational cost of pre-training on large-scale corpora. Additionally, the incorporation of manually verified triplet knowledge enhances the interpretability of the trained models.

To enhance the training effectiveness of our model and emphasize the entities in the text, we employ a multi-level, multidimensional representation approach by separately inputting the text and entities into the model during training. We construct a BiLSTM network [12] and incorporate a fully connected layer, designing a model that is more suitable for domain relation classification tasks. The functions of each module are illustrated in Figure 3.

Figure 3
Functions of each module



Our contributions are as follows: (1) The method of step-by-step input, combined with the introduction of entity type labels at the text level, emphasizes and allows the model to understand domain-specific terms from multiple dimensions. (2) We apply domain knowledge graphs to entity relation extraction tasks and design an end-to-end framework for entity relation extraction.

2. Related Work

2.1. Relation extraction

Various deep learning-based entity relation extraction models have been proposed thus far. Prior to these methods based on deep learning, most approaches were based on statistical machine learning, where the performance heavily relied on the quality of extracted text features. Text features were typically obtained using existing natural language processing tools. However, this approach often led to the problem of error propagation, as incorrect feature extraction would limit the overall model performance. Zeng et al. [13] introduced a model that does not require complex preprocessing and obtains word-level features from given nouns. They used a convolutional neural network to extract sentence-level features and then concatenated the two to feedback them into a softmax layer for predicting the relationship between two entities. Experimental results demonstrated that this deep learning-based feature extraction method outperformed the feature extraction based on existing natural language processing tools. Socher et al. [14] addressed the problem of models being unable to capture information composed of long and short phrases, which hindered a deep understanding of textual meaning. The authors introduced a recursive neural network [15] model that could learn arbitrary syntactic types, lengths, and phrase and sentence vector representations. This model assigned a matrix or vector to each node in the parse tree, with the matrix capturing changes in the meaning of nearby words and phrases, while the vector captured the inherent meaning of the sentence composition. This allowed the proposed matrix-vector RNN model to encode a complete sentence representation from the bottom up based on the

tree structure. Experimental results demonstrated that this model performed well on large and noisy datasets. Yu et al. [16] proposed Factor-based Compositional Embedding Models, which construct sentence-level and substructure word embeddings using dependency trees and named entities, allowing for better handling of global information in text. dos Santos et al. [17] introduced the CR-CNN convolutional neural network model, which solves relation classification tasks by ranking the loss function. Experimental results demonstrated that this approach outperforms the CNN model with a softmax layer. Shen and Huang [18] utilized a CNN encoder combined with attention on target entities and words in the sentence to weight the words, enabling better identification of the most influential parts of the sentence for the two entities. Lee et al. [19] argued that attention-based neural network models did not fully leverage entity-related information, which could be crucial in relation classification tasks. To address this issue, they proposed an end-to-end recursive neural model that integrates entity-aware attention mechanisms and entity types. This model effectively utilizes entities and their potential types as features and also constructs word representations based on the sentence's symmetric similarity using self-attention. Wu and He [20] were the first to propose applying the pre-trained BERT model to relation extraction tasks. They added independent markers to target entities, transmitted information through the pre-trained architecture, and combined the encodings corresponding to the two entities.

2.2. Joint entity and relation extraction

Joint entity and relation extraction refers to the task of extracting both entities and their relationships from text simultaneously. Traditionally, these tasks were treated as separate sequential steps, where entity recognition was performed first, followed by relation extraction. However, joint entity and relation extraction models aim to tackle both tasks concurrently, leveraging the inherent dependencies between entities and their relationships.

Miwa and Bansal [21] proposed a method for joint entity and relation recognition, where the loss functions of entity recognition and relation extraction tasks are backpropagated and updated simultaneously. This approach reduces the problem of error propagation in pipeline models and addresses the issue of disjointed subtasks in relation extraction within pipeline models. Zheng et al. [22] achieved joint entity and relation extraction by transforming the tasks of named entity recognition and relation extraction into sequence labeling tasks, using pattern labeling for joint decoding of entity relations. However, this model cannot handle the issue of entity overlap. Dai et al. [23] introduced a position-attention mechanism that allows the model to generate different sentence representations for each query position, effectively solving the problem of overlapping relations. Shang et al. [24] generated candidate entities by enumerating word token sequences and designed a linking matrix for each relation to detect whether two candidate entities could form a triplet. They transformed the triplet extraction task into a relation-specific bipartite graph linking problem. Experimental results demonstrated that this method performed well in complex scenarios with different overlapping patterns.

2.3. Pre-training model

Since Google Inc. launched BERT in 2018, many endeavors have been made for further optimization. Baidu-ERNIE [25] and RoBERTa-wwm-ext [26] adopted a strategy of whole-word masking instead of the traditional BERT's single-character masking in their Chinese corpus BERT pre-training. This strategy enables better capturing of semantic information in Chinese vocabulary. SpanBERT [27], built upon BERT, masks continuous random segments and introduces segment boundary objectives. This method allows the model to learn the relationships between different segments, leading to a better understanding of sentence structure and semantics. RoBERTa [28] is an optimized model that further improves the pre-training process of BERT. It incorporates three strategies: removing the next sentence prediction task, dynamically changing the masking strategy, and training with more and longer sentences. The combination of these strategies allows RoBERTa to better capture contextual information during pre-training. Regarding the encoder aspect of BERT, Yang et al. [6] introduced Transformer-XL [29], which is more suitable for handling long sentences, as a replacement for the Transformer in BERT. This enhances the model's ability to comprehend long texts. THU-ERNIE modified the encoder of BERT and attempted to incorporate knowledge graph information into entities, making it an aggregator that integrates vocabulary and entities mutually. However, it did not consider the relationship information between entities. COMET [30] directly utilizes triples from the knowledge graph as the training corpus for the GPT [5] model to facilitate commonsense learning. However, this method is highly inefficient.

3. Methodology

3.1. Notation

The input consists of a sentence $X = \{x_1, x_2, x_3, \ldots, x_n\}$ composed of n word tokens and all entities $s_i = \{x_{START(i)}, x_{START(i)+1}, \ldots, x_{END(i)}\}$ along with their corresponding entity types $y_e(s_i) \in \varepsilon$. Here, $S = \{s_1, s_2, s_3, \ldots, s_m\}$ represents all possible contiguous spans in X with a length not exceeding L. $START(i)$ and $END(i)$ indicate the starting and ending positions of s_i respectively. ε denotes the predefined set of entity types, and $\mathcal{R}$ represents the predefined set of relation types. Each word token x_i is included in the vocabulary $\mathbb{V}$. The knowledge graph is denoted as $\mathbb{K}$ and consists of triples $\varepsilon = \{x_i, r_j, x_k\}$, where x_i and x_k are entity names, and $r_j \in \mathbb{V}$ represents the relation between entities. All triples exist in the knowledge graph $\mathbb{K}$, i.e., $\varepsilon \in \mathbb{K}$. Given each entity pair $s_i \in S$ and $s_j \in S$, the task is to predict the relationship $y_r(s_i, s_j) \in \mathcal{R}$ between the entity pairs. The output of the task is $Y_r = \{(s_i, r, s_i) : s_i, s_j \in S, r \in \mathcal{R}\}$.

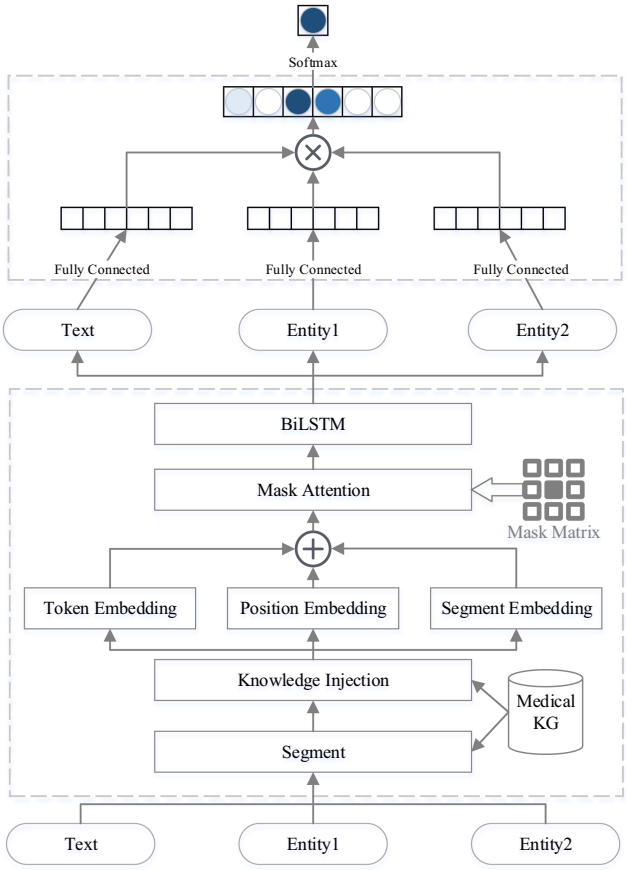
3.2. Model architecture

To make the model more suitable for the Chinese context, we employed the RoBERTa-wwm-ext [26] pre-trained embeddings when loading the pre-trained model. To enhance the model's attention to the crucial parts of the text, we input the text and two entities separately into the model. The model can capture the important part of the text and comprehend it from multiple perspectives.

This paper proposes a method that incorporates domain knowledge into pre-training, inspired by the idea of K-BERT [31]. The word embeddings are enriched with domain knowledge to improve the accuracy of information extraction in specialized domains. Following the approach of the PURE model, entity position and type information labels are added. Each word in the sentence is scanned and matched with entities in the domain knowledge graph. The triples contained in the domain knowledge graph are integrated into the input sentence in a tree structure and unfolded into sequential inputs for the encoding layer. The model

framework is illustrated in Figure 4. The symbol "⊕" denotes element-wise addition, and "⊗" represents vector concatenation operation.

Figure 4
Model architecture



3.3. Input layer

In order to independently handle each entity pair (s_i, s_j), it is necessary to extract the entity positions and provide entity type information. Therefore, the approach of inserting labels at the input layer is chosen. Given the input sentence X and entities s_i and s_j with entity types e_i and e_j, respectively, the text labels are defined as $< S : e_i >$, $< /S : e_i >$, $< O : e_j >$, and $< /O : e_j >$, and they are inserted before and after the two entities as shown in Figure 5. The inserted input with labels denoted as $\widehat{X}$ and is represented by

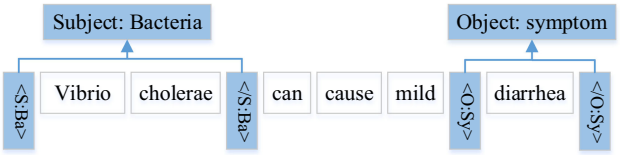
$$\widehat{X} = x_1, \ldots, < S : e_i >, x_{START(i)}, \ldots, x_{END(i)}, < /S : e_i >, \ldots,$$

$$< O : e_j >, x_{START(j)} \ldots, x_{END(j)}, < /O : e_j >, \ldots, x_n$$

3.4. Knowledge layer

For the input sentence $\widehat{X}$, we first use a Chinese word segmentation tool enhanced with domain knowledge to tokenize the sentence, improving the accuracy of domain-specific segmentation. After tokenization, we incorporate the triples ε from the domain knowledge graph $\mathbb{K}$ into the original sentence $\widehat{X}$. This transforms $\widehat{X}$ into a sentence tree that contains domain knowledge, which is then fed into the

Figure 5
Example of inserting label



embedding layer to generate a mask matrix. The mask matrix is used to control the selective attention mechanism of the word token sequence, preventing the model from distorting the original meaning of the sentence due to the injection of excessive knowledge.

Knowledge Integration serves two purposes: incorporating the triple knowledge from the domain knowledge graph into the sentence to enrich the domain information and unfolding the sentence tree structure into a sequence for the subsequent attention mechanism layer. Specifically, given the sentence $\widehat{X} = \{x_1, x_2, x_3, \ldots, x_n\}$ with indices sorted according to word tokens and the domain knowledge graph $\mathbb{K}$, after passing through the knowledge integration layer, which includes knowledge querying and knowledge injection steps, the output is the sentence tree $T = \{x_1, x_2, \ldots, x_i\{(r_{i0}, x_{i0}), \ldots, (r_{ik}, x_{ik})\}, \ldots, x_n\}$.

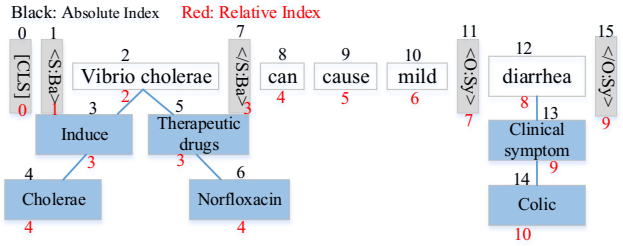
In the knowledge querying step, all entity names involved in the sentence $\widehat{X}$ are queried against the domain knowledge graph $\mathbb{K}$ to retrieve the corresponding triples. This can be represented by the formula $E = K\_Query\left(\widehat{X}, \mathbb{K}\right)$.

Next, in the knowledge injection step, the triples from E are concatenated at the corresponding positions of the sentence sequence $\widehat{X}$ to form the sentence tree T. The sentence tree can have multiple branches, but entities within a branch will not further iterate to create new branches, meaning the depth of the tree is 1. This can be represented by the formula $T = K\_Inject\left(\widehat{X}, E\right)$.

The function of the embedding layer is to transform the tree-structured sentence T into computationally understandable vectors through embedding representations. Drawing inspiration from the BERT model, the sentence tree T is embedded as a sum of text embeddings, position embeddings, and segment embeddings. In the BERT model, the input is a sequence of word tokens rather than a sentence tree. Therefore, the key challenge lies in how to convert the sentence tree T into the required sequence structure for word embeddings without losing the tree structure information.

In the text embedding part, the RoBERTa-wwm-ext pre-trained model, which is more suitable for Chinese classification tasks, is employed to generate word embedding vectors. Each word token in the sentence tree is transformed into a trainable embedding vector of dimension H. Similar to BERT, the [CLS] token is added at the beginning of the input sentence to represent the classification task, and the [MASK] token is used to mask word tokens. To convert the tree structure into the sequence structure required by the attention layer, the K-BERT approach is applied here. Each word token is assigned an absolute index and a relative index. The absolute index performs a depth-first traversal of the sentence tree starting from the root node [CLS] (indexed as x_0) to convert it into a sequence. The relative index represents the distance of each word token node from the root node [CLS]. The specific process is illustrated in Figure 6. By sorting the sentence tree based on the absolute index, the tree structure of the sentence is transformed into a sequential structure. However, this transformation also poses a challenge of unreadable sentences, which will be addressed in the position embedding layer.

Figure 6
Absolute and relative index



In the position embedding part, the attention mechanism does not capture the structural order information of the text sequence. For the BERT model, all the positional structural information of the sentence is encoded in the position embeddings. In the sentence sequence, the absolute indices "flatten" the sentence tree, making it unreadable. However, the relative indices retain the structural information of the tree. Therefore, the information of the relative indices is inputted to the position embedding layer, achieving the goal of preserving the sentence tree structure while inputting the sentence sequence. The embedding vectors are consistent with the Transformer model, and the specific calculation methods are shown in Formulas (1) and (2),

$$PE_{(index,2i)} = \sin\left(\frac{index}{10000^{\frac{2i}{H}}}\right) \tag{1}$$

$$PE_{(index,2i+1)} = \cos\left(\frac{index}{10000^{\frac{2i}{H}}}\right) \tag{2}$$

where PE represents the position embedding vector, $index$ corresponds to the relative index of the word token, i denotes the i-th dimension of the vector, and H represents the total dimensionality of the vectors. This encoding method effectively captures the relative distances between word tokens, preserving the structural information of the input sentence while avoiding limitations imposed by sentence length.

The segment embedding part is the same as in BERT, aiming to enable the model to recognize multiple input sentences. For example, given two sentences $X = \{x_1, x_2, x_3, \ldots, x_n\}$ and $Y = \{y_1, y_2, y_3, \ldots, y_n\}$, a [SEP] label is added as a separator in the input layer, resulting in a sequence $\{[CLS], x_1, x_2, x_3, \ldots, x_n, [SEP], y_1, y_2, y_3, \ldots, y_n\}$. To differentiate between different input sequences, the segment embedding for sentence X is defined as A, and for sentence Y as B. Therefore, the segment embedding sequence becomes $\{A, A, A, A, \ldots, A, B, B, B, B, \ldots, B\}$.

Incorporating domain knowledge into the input of the relation extraction task undoubtedly helps the model's understanding. However, it also introduces the problem of knowledge noise. When too much domain knowledge is injected into a sentence, it may dilute the focus of the original sentence or even distort its intended meaning. To alleviate this issue, a mask matrix is introduced in the attention mechanism to prevent the excessive influence of the added triplet tokens on the original sentence's meaning. Without this mitigation, the sentence may suffer from issues such as cluttered knowledge information, diluted emphasis, and distorted original meaning. To prevent such situations, we generate a mask matrix M from the sentence tree T, which ensures that the words in branch positions do not undergo attention calculations with other words in the main trunk positions. The mask matrix M is defined as shown in Formula (3),

$$M_{ij} = \begin{cases} 0 & x_i \ominus x_j \\ -\infty & x_i \oslash x_j \end{cases} \tag{3}$$

where the symbol $\ominus$ represents word tokens within the same branch, and $\oslash$ denotes word tokens that are not in the same branch. The indices i and j refer to the absolute index positions.

The attention mechanism incorporates a mask matrix into the self-attention part of the Transformer model [32]. The specific formula is shown as (4), (5) and (6),

$$Q^{i+1}, K^{i+1}, V^{i+1} = h^i W_q, h^i W_k, h^i W_V, \tag{4}$$

$$S^{i+1} = softmax\left(\frac{Q^{i+1}K^{i+1^T} + M}{\sqrt{H}}\right), \tag{5}$$

$$h^{i+1} = S^{i+1}V^{i+1} \tag{6}$$

where $W_q, W_k,$ and W_V represent trainable model parameters, h^i denotes the hidden layer of the i-th masked self-attention module, H represents the dimensionality of the embedding vectors, and M represents the mask matrix. From the formulas, it can be visually observed that when calculating the word tokens x_i and x_j situated in different branches, $M_{ij} = -\infty$. After applying the softmax function, S_{ij}^{i+1} becomes 0. This implies that the hidden state of x_i has no influence on x_j, reducing the risk of distorting the original sentence meaning due to excessive injection of knowledge.

By employing the attention mechanism, along with the BiLSTM network, we enhance the contextual modeling of the sentences. The output is denoted as $\widehat{T} = \{t_1, t_2, t_3, \ldots, t_n\} \in \mathbb{R}^{n \times H}$, where H represents the dimensionality of the embedding vectors.

3.5. Decoder layer

After the encoding layer, the output goes through fully connected and activation layers before being passed through a softmax layer for relationship classification. The specific formula is shown as (7),

$$T' = W\left(\tanh(\widehat{T})\right) + b \tag{7}$$

where $W \in \mathbb{R}^{L \times H}$, b is the bias vector, and L is the number of relationship types. To decode the input sentences in a multidimensional and hierarchical manner, we propose a partial input approach. We extract the two entities separately from the sentence and feed them back into the model, resulting in T_1' and T_2'. The outputs of the three inputs are concatenated and passed through a softmax layer to obtain the final output. The specific formula is shown as (8),

$$p = softmax\left(\left[T' : T_1' : T_2'\right]\right) \tag{8}$$

4. Experiments

4.1. Dataset and evaluation metric

The pre-training corpus used in this study consists of the Chinese Wikipedia corpus, WikiZh, and a large-scale, high-quality Chinese

question and answer (Q&A) corpus, WebtextZh. WikiZh has a size of 1.2 GB and includes one million well-formatted Chinese articles and 120 million sentences, which are used for pre-training the Chinese BERT model. WebtextZh, with a size of 3.7 GB, comprises 4.1 million articles covering 28,000 topics. For injecting knowledge, we utilized the domain knowledge graph called MedicalKG [31] specifically designed for the domain of Traditional Chinese Medicine. MedicalKG consists of four types of hypernyms (symptoms, diseases, parts, treatments) and contains a total of 13,864 triplets. These triplets were used for both word segmentation and knowledge incorporation purposes.

The dataset used in this study is a medical domain relation classification dataset, with text content sourced from Baidu encyclopedia. It consists of 5,500 sentences, encompassing 13 different relationship types, namely clinical symptoms, related diseases, applicable symptoms, causes of diseases, commonly used medications, applicable diseases, major causes, medical treatment symptoms, symptoms caused by, departments for medical consultation, diseases treated by, applicable departments, and examination items. Each sentence contains two entity nouns (e_1 and e_2) along with their corresponding entity types, including diseases, symptoms, bacteria, medical specialties, and others. The dataset was divided into about 3,500 training samples, 1,000 validation samples, and 1,000 test samples.

In multi-class tasks, there are two calculation methods for the F1-score: Micro-F1 and Macro-F1. The formulas for calculating the Recall and Precision are as follows:

$$Recall = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FN_i)} \tag{9}$$

$$Precision = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FP_i)} \tag{10}$$

In the formulas, TP_i represents the true positives for class i, which means a positive class is correctly predicted as positive. FN_i represents the false negatives for class i, which means a positive class is incorrectly predicted as negative. FP_i represents the false positives for class i, which means a negative class is incorrectly predicted as positive. n represents the number of classes. The calculation formula for Micro-F1 is as follows:

$$Micro - F1 = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{11}$$

This method assigns equal weights to each sample, making it suitable for datasets with relatively balanced classification samples.

Macro-F1 assigns equal weights to each class, making it unaffected by data imbalance in multi-class problems. However, it can be influenced by classes with high precision and recall rates. The calculation method involves first calculating the F1-score for each class and then taking the average of the F1-scores across all classes. The specific formula is as follows:

$$F1 - Score_i = 2 \times \frac{Recall_i \times Precision_i}{Recall_i + Precision_i} \tag{12}$$

$$Macro - F1 = \frac{\sum_{i=1}^{n} F1 - Score_i}{n} \tag{13}$$

In this formula, $Recall_i$ and $Precision_i$ represent the recall and precision of class i, respectively. In this study, we used the *Accuracy* provided by the sklearn package. The calculation method for *Accuracy* is to divide the number of correctly predicted samples by the total number of samples. The formula for *Accuracy* is as follows:

$$Accuracy = \frac{\sum_{i=1}^{n} TP_i}{Total} \tag{14}$$

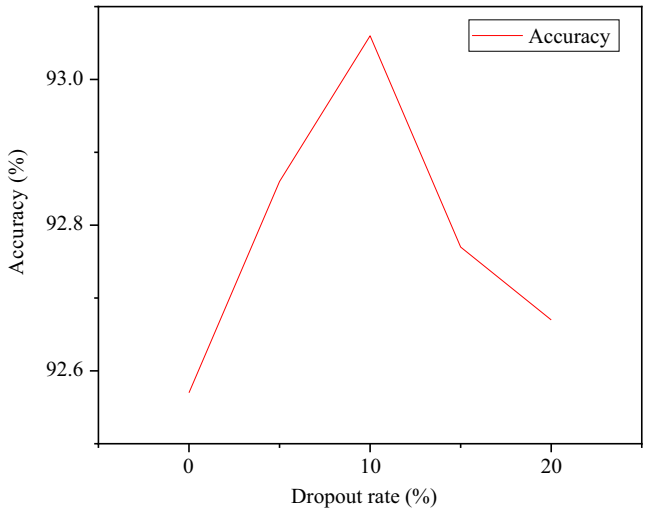
In this formula, TP_i stands for "True Positives" of the i^{th} sample. In classification tasks, a "True Positive" refers to an instance where the model correctly predicts a positive class sample as positive. The subscript i ranges from 1 to n, indicating an iteration over all samples, where n is the total number of samples. Total represents the total number of samples, which is the count of all the samples in the dataset.

4.2. Parameter settings

In order to conduct a fair comparison in our experiments, we configured the model in the pre-training phase with the same parameters as Google BERT. This includes using a self-attention layer with $L = 12$, a multi-head count of $A = 12$, a learning rate of $Lr = 0.00005$, and embedded vector hidden dimensions of $H = 768$. Our model has the same number of trainable parameters as BERT, which is 110 million, making our method and BERT compatible with each other in terms of parameters. We named our method R-KBERT.

We add dropout before each add-on layer, as shown in Figure 7. From the parameter analysis experiments, it can be observed that appropriate dropout, which randomly sets some neuron output values to zero, reduces the complex interactions between neurons, mitigates the risk of overfitting in the network, and improves the robustness and generalization ability of the network. However, when the dropout rate is too high, it may result in the model discarding useful textual features and increasing the risk of underfitting. Therefore, we set the dropout rate to 0.1.

Figure 7
Parameter analysis



4.3. Comparison with other methods

The baseline models include an SVM [33], which uses a support vector machine classifier to capture the context-entity relationships.

There is also an RNN [14] with short-term memory capabilities. Building upon the RNN model, we introduced the Matrix-Valued RNN (MVRNN) that alters the contextual semantics. The CNN and softmax [13] take word embeddings and positional features as input, concatenating them with lexical features. The FCM [16] constructs sentence-level and substructure word embeddings based on dependency trees and named entities. The CR-CNN [17] ranks the loss function. The Attention-CNN [18] utilizes CNN encoders combined with attention weights that are based on the target entities and words in the sentence. The Entity Attention BiLSTM [19] model integrates entity-aware attention mechanisms and entity types into an end-to-end recursive neural network. For the first time, we employ the BERT pre-training model for relation classification tasks, known as R-BERT [20]. Additionally, we incorporate fixed and floating markers, labeled as PL-Marker [34], to indicate the subject and object entities in the sentence. The results are presented in Table 1.

Table 1
Comparison with results in the literature

Method	Accuracy (%)
SVM [33]	82.26
MVRNN [14]	83.85
CNN + Softmax [13]	84.54
FCM [16]	85.23
CR-CNN [17]	87.31
Attention-CNN [18]	88.80
Entity Attention BiLSTM [19]	88.21
R-BERT [20]	90.78
PURE [35]	91.77
PL-Marker [34]	92.27
GDA [36]	92.47
R-KBERT	**93.06**

From Table 1, it can be observed that our *Accuracy* is 93.06%, which is significantly superior to all other baseline methods.

4.4. Ablation studies

To understand the contributions of each module to relation classification, we denote the absence of the BiLSTM layer as "NoLSTM," the absence of inserted type label information as "NoLabel," the absence of the entities input method as "NoEntity," using the absolute index as position embedding as "NoIndex," and all tokens make attention to each other as "NoMatrix." For specific details, refer to Table 2.

Table 2
Ablation experiment labels

Label	Detail
NoLSTM	Absence of the BiLSTM layer
NoLabel	Without inserting type label in text
NoEntity	Without additional input of two entities
NoIndex	Use absolute index as position embedding
NoMatrix	All tokens make attention to each other

Table 3 reports the results of the ablation study with the above configurations. We observed that the methods in the ablation experiments performed worse than our method. This demonstrates the effectiveness of each module in our model. These modules alleviate or address the corresponding problems. The BiLSTM layer addresses the issue of long-term dependencies in text. It is crucial to integrate entity boundaries and type information at the input layer, and by combining the approach of separately inputting entities, the key content in the text is explicitly identified. The position embedding utilizes relative indexes for computation, resolving the problem of text disorganization after incorporating knowledge. The performance of the model declined after removing the Mask Matrix, which demonstrates the necessity of constraining the computation of the attention.

Table 3
Comparison with the absence of different components

Methods	Pre-training model	Accuracy (%)
R-KBERT (NoLSTM)	BERT	91.38
R-KBERT (NoLabel)	BERT	91.28
R-KBERT (NoEntity)	BERT	91.67
R-KBERT (NoIndex)	BERT	91.38
R-KBERT (NoMatrix)	BERT	90.98
R-KBERT	BERT	92.67
R-KBERT (NoLSTM)	RoBERTa	91.58
R-KBERT (NoLabel)	RoBERTa	91.67
R-KBERT (NoEntity)	RoBERTa	92.17
R-KBERT (NoIndex)	RoBERTa	91.87
R-KBERT (NoMatrix)	RoBERTa	91.48
R-KBERT	RoBERTa	**93.06**

5. Conclusion

In this paper, we propose the application of domain knowledge graphs to relation classification tasks, aiming to enhance the model's understanding of domain-specific knowledge. Firstly, we inject domain knowledge into sentences, transforming them into knowledge-enriched sentence trees. Secondly, we utilize relative indices and a mask matrix to control the scope of knowledge attention, preventing the sentences from deviating from their original meaning due to an overwhelming amount of knowledge. Additionally, we employ a partial input approach to enable the model to comprehend input sentences from multiple angles and at multiple levels, achieving a more comprehensive understanding. We have achieved an improvement of 0.64% in accuracy on the medical domain dataset. Our method is applicable to domain-specific relation extraction tasks and improves accuracy without significantly increasing time complexity. As for future work, we are committed to expanding this method to achieve similarly good performance on distantly supervised datasets.

Ethical Statement

This study does not contain any studies with human or animal subjects performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest to this work.

Data Availability Statement

The data that support the findings of this study are openly available in [Github] at https://github.com/Soso6666/Classification-data

Author Contribution Statement

Boxuan Chen: Methodology, Software, Validation, Resources, Data curation, Writing – original draft, Visualization. **Guan Yuan:** Conceptualization, Methodology, Formal analysis, Investigation, Resources, Data curation, Writing – review & editing, Supervision, Project administration.

References

[1] Ratinov, L., & Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning,* 147–155.

[2] Sang, E. T. K., & de Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL*, 142–147.

[3] Mooney, R. J., & Bunescu, R. (2005). Mining knowledge from text using information extraction. *ACM SIGKDD Explorations Newsletter*, *7*(1), 3–10. https://doi.org/10.1145/1089815.1089817

[4] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 4171–4186.

[5] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. *Papers with Codes Preprint 2018*.

[6] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems,* 5753–5763.

[7] Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2020). BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, *36*(4), 1234–1240. https://doi.org/10.1093/bioinformatics/btz682

[8] Beltagy, I., Lo, K., & Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 3615–3620. https://doi.org/10.18653/v1/D19-1371

[9] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv Preprint:1301.3781*. https://doi.org/10.48550/arXiv.1301.3781

[10] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems, 1–9*.

[11] Bodenreider, O. (2008). Biomedical ontologies in action: Role in knowledge management, data integration and decision support. *Yearbook of Medical Informatics*, *17*(01), 67–79. https://doi.org/10.1055/s-0038-1638585

[12] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* 260–270.

[13] Zeng, D., Liu, K., Lai, S., Zhou, G., & Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2335–2344.

[14] Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 1201–1211.

[15] Socher, R., Lin, C. C., Manning, C., & Ng, A. Y. (2011). Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning*, 129–136.

[16] Yu, M., Gormley, M., & Dredze, M. (2014). Factor-based compositional embedding models. *NIPS Workshop on Learning Semantics*, *411*, 95–101.

[17] dos Santos, C., Xiang, B., & Zhou, B. (2015). Classifying relations by ranking with convolutional neural networks. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, 1*, 626–634. https://doi.org/10.3115/v1/P15-1061

[18] Shen, Y., & Huang, X. J. (2016). Attention-based convolutional neural network for semantic relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2526–2536.

[19] Lee, J., Seo, S., & Choi, Y. S. (2019). Semantic relation classification via bidirectional LSTM networks with entity-aware attention using latent entity typing. *Symmetry*, *11*(6), 785. https://doi.org/10.3390/sym11060785

[20] Wu, S., & He, Y. (2019). Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2361–2364. https://doi.org/10.1145/3357384.3358119

[21] Miwa, M., & Bansal, M. (2016). End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, 1*, 1105–1116.

[22] Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., & Xu, B. (2017). Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, 1*, 1227–1236.

[23] Dai, D., Xiao, X., Lyu, Y., Dou, S., She, Q., & Wang, H. (2019). Joint extraction of entities and overlapping relations using position-attentive sequence labeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 6300–6308. https://doi.org/10.1609/aaai.v33i01.33016300

[24] Shang, Y. M., Huang, H., Sun, X., Wei, W., & Mao, X. L. (2022). Relational triple extraction: One step is enough. *arXiv Preprint:2205.05270*. https://doi.org/10.48550/arXiv.2205.05270

[25] Sun, Y., Wang, S., Li, Y., Feng, S., Chen, X., Zhang, H., . . . , & Wu, H. (2019). ERNIE: Enhanced representation through knowledge integration. *arXiv Preprint:1904.09223*. https://doi.org/10.48550/arXiv.1904.09223

[26] Cui, Y., Che, W., Liu, T., Qin, B., & Yang, Z. (2021). Pre-training with whole word masking for Chinese BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *29*, 3504–3514. https://doi.org/10.1109/TASLP.2021.3124365

[27] Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., & Levy, O. (2020). Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, *8*, 64–77. https://doi.org/10.1162/tacl_a_00300

[28] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., . . . , & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv Preprint:1907.11692*. https://doi.org/10.48550/arXiv.1907.11692

[29] Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q., & Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2978–2988. https://doi.org/10.18653/v1/P19-1285

[30] Bosselut, A., Rashkin, H., Sap, M., Malaviya, C., Celikyilmaz, A., & Choi, Y. (2019). COMET: Commonsense transformers for knowledge graph construction. *arXiv Preprint:1906.05317*. https://doi.org/10.48550/arXiv.1906.05317

[31] Liu, W., Zhou, P., Zhao, Z., Wang, Z., Ju, Q., Deng, H., & Wang, P. (2020). K-BERT: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, *34*(03), 2901–2908. https://doi.org/10.1609/aaai.v34i03.5681

[32] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . , & Polosukhin, I. (2017). Attention is all you need. In *31st Conference on Neural Information Processing Systems*.

[33] Rink, B., & Harabagiu, S. (2010). Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, 256–259.

[34] Ye, D., Lin, Y., Li, P., & Sun, M. (2021). Packed levitated marker for entity and relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, 1,* 4904–4917.

[35] Zhong, Z., & Chen, D. (2021). A frustratingly easy approach for entity and relation extraction. In *2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 50–61.

[36] Hu, X., Liu, A., Tan, Z., Zhang, X., Zhang, C., King, I., & Yu, P. S. (2023). GDA: Generative data augmentation techniques for relation extraction tasks. In *Findings of the Association for Computational Linguistics*, 10221–10234. https://doi.org/10.18653/v1/2023.findings-acl.649

How to Cite: Chen, B. & Yuan, G. (2024). Domain Knowledge-Driven Relation Extraction Methods. *Journal of Data Science and Intelligent Systems*. https://doi.org/10.47852/bonviewJDSIS42022524