electronics

MDPI

Article

REEGAT: RoBERTa Entity Embedding and Graph Attention Networks Enhanced Sentence Representation for Relation Extraction

Fengze Cai [1,†], Qiang Hu [2,†], Renjie Zhou [2,*] and Neal Xiong [3,*]

[1] Zhuoyue Honors College, Hangzhou Dianzi University, Hangzhou 310018, China; caifengze@hdu.edu.cn
[2] College of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China; huqiang@hdu.edu.cn
[3] Department of Computer, Mathematical and Physical Sciences Sul Ross State University, Alpine, TX 79830, USA
* Correspondence: rjzhou@hdu.edu.cn (R.J.); neal.xiong@sulross.edu (N.X.)
† These authors contributed equally to this work.

Abstract: Relation extraction is one of the most important intelligent information extraction technologies, which can be used to construct and optimize services in intelligent communication systems (ICS). One issue with the existing relation extraction approaches is that they use one-sided sentence embedding as their final prediction vector, which degrades relation extraction performance. The innovative relation extraction model REEGAT (RoBERTa Entity Embedding and Graph Attention networks enhanced sentence representation) that we present in this paper, incorporates the concept of enhanced word embedding from graph neural networks. The model first uses RoBERTa to obtain word embedding and PyTorch embedding to obtain relation embedding. Then, the multi-headed attention mechanism in GAT (graph attention network) is introduced to weight the word embedding and relation embedding to enrich further the meaning conveyed by the word embedding. Finally, the entity embedding component is used to obtain sentence representation by pooling the word embedding from GAT and the entity embedding from named entity recognition. The weighted and pooled word embedding contains more relational information to alleviate the one-sided problem of sentence representation. The experimental findings demonstrate that our model outperforms other standard methods.

Keywords: information extraction; relation extraction; contrastive learning; graph attention network

Citation: Cai, F.; Hu, Q.; Zhou, R.; Xiong, N. REEGAT: RoBERTa Entity Embedding and Graph Attention Networks Enhanced Sentence Representation for Relation Extraction. *Electronics* **2023**, *12*, 2429. https://doi.org/10.3390/electronics12112429

Academic Editor: Simeone Marino

Received: 15 March 2023
Revised: 8 May 2023
Accepted: 9 May 2023
Published: 27 May 2023

1. Introduction

Information extraction is one of the fundamental capabilities of intelligent communication systems (ICS) to provide intelligent services to people. Information extraction is used in many different industries built upon ICS, such as web data retrieval systems [1,2], recommendation systems [3,4], the intelligent medical field [5,6], emotion monitoring systems [7], and the wireless sensing field [8,9]. Moreover, information extraction is utilized by government departments to gain insights into public opinion [10,11], etc. In the field of information extraction, relation extraction, or RE for short, is a critical task. With RE, relationships between entities are extracted from unstructured text and stored as structured data. The most common form of storage for structured data here is a triad, i.e., (head entity, relationship to tail entity, tail entity), where entity refers to a word containing a proper name such as a person, place, or organization, and the relation extraction technique can extract the relation between entities.

At first, information extraction depended on pattern matching, in which a specialist or academic in the relevant field constructed a recognition template using the properties of a sample dataset. KNN-BLOCK DBSCAN, presented by Chen et al. [12], is an efficient

clustering algorithm for large-scale datasets by incorporating KNN and block techniques to reduce computational complexity. Although this method effectively achieved high recognition rates within a specific domain, it lacked portability. As a result, scientists have steadily moved their attention to investigating machine learning techniques. Giorgi et al. [13] pointed out that joint learning based on NER and RE should be used for information retrieval and extraction. In the field of RE, traditional machine methods rely on statistical language models, and the extraction process is divided into a learning process and a prediction process [14], where the learning process involves training a relation extraction model suitable for the current corpus through a practical training method. The prediction process predicts the test text through the trained relation extraction model. Information extraction commonly employs traditional machine learning classification models such as hidden Markov models (HMM), maximum entropy (ME), decision trees (DT), support vector machines (SVM), etc.

Large-scale pre-trained language models have become more prevalent recently in various natural language processing tasks. These models, such as BERT [15], Albert [16], RoBERTa [17], and Xlnet [18], have proven to be dominant players in tasks such as text generation, text classification, reading comprehension, entity recognition, and POS tagging, thanks to the advancements in deep learning. These models are based on the transformer architecture [19], which uses a self-attention mechanism to record, keep, and track tokens' distant connections. Transformer encoders and decoders can perform self-supervised learning tasks, such as anticipating concealed and upcoming tokens, through pre-training on a significant text corpus.

RE is a classification problem about sentences. When performing a relational extraction task with the idea of text classification, to obtain the sentence embedding, the word embedding of the initial token in a sentence is typically pooled, and then the sentence embedding is used as the final prediction vector. The first token is chosen because it is more "meaningless" than the other word segmentation tokens, so it is more balanced to obtain information from the corresponding text words during the pooling operation. However, the relational extraction task needs to focus on the information about the entity words in the sentence. The information about the other words in the sentence needs to be addressed by the sentence embedding created by pooling the first token. If the information about the entity words is ignored, it will affect the model's predictive performance. Therefore, if only this sentence embedding vector is used as the final prediction vector, there is a specific problem of one-sidedness.

Therefore, we propose the REEGAT (RoBERTa Entity Embedding and Graph Attention networks enhanced sentence representation) model to alleviate the one-sidedness problem that exists. The first step is to add relationship information. In this step, we use GAT to weight the word embedding vector and the relational embedding vector in the text with each other. This makes the final obtained sentence embedding vector also contain relational information. The second step is to add entity word information. During this step, we extract entities using the WCL-BBCD model. Then, the entity embedding vector is stitched with the sentence embedding vector obtained in the first step, so that the sentence embedding vector contains both relationship and entity information so that it can alleviate the one-sidedness problem.

The following is a summary of this paper's significant contributions:

1. In most current models, word embedding vectors are directly used to obtain sentence embedding vectors. However, the resulting sentence embedding vectors lack relational information. Therefore, we use the attention mechanism in the graph attention network model to weight the word embedding vector of the text. We weight the word embedding vector trained by RoBERTa with the relational embedding term. The weighted word embedding vector contains relational information. Then, the sentence embedding vector is obtained through the word embedding vector. At this time, the sentence embedding vector includes relational information.

2. We extract entities from the training corpus with the help of the WCL-BBCD model. The entities are converted into word embedding vectors, i.e., entity embedding vectors. Moreover, the sentence embedding vector and the entity embedding vector are stitched together to form the input vector of the final fully connected layer. As a result, the sentence embedding vector contains relational and entity information, which gives it a richer meaning.
3. The REEGAT model is proposed to alleviate the one-sidedness problem based on the above approach. For the SemEval-2010 Task 8 and Wiki80 datasets, REEGAT surpassed the other models described in this study.

The remainder of this paper is organized as follows to present our work: Section 2 provides a review of related work in the field; Section 3 presents the suggested approach in full detail; Section 4 discusses and analyzes the experimental findings obtained through our system; and Section 5 summarizes and concludes the paper, highlighting the significance and implications of our research findings.

2. Related Work

2.1. Relation Extraction

In natural language processing, a relation is a connection between two or more entities. RE aims to identify and extract these relations from text, resulting in structured data representing a triad with the corresponding entities.

The three primary approaches to relation extraction are rule-based, deep-learning-based, and traditional machine-learning-based methods. Rule-based methods rely on the expertise of linguistic experts to develop grammar rules that facilitate relation classification. While this approach may be effective in some domains, it may not be as versatile as other methods. A potential way to improve the effectiveness of text pattern methods such as the one proposed by Nakashole et al. [20], is to incorporate prior knowledge, such as knowledge graphs, into the representation of entities and their relationships. This approach could enhance the accuracy of the extracted relationships and make them more useful for downstream applications.

The rule-based approach has the advantage of having high extraction accuracy in a constrained field. The disadvantage is that the construction of its rules requires a lot of human resources, and the portability of the completed regulations is poor.

Zhang et al. summarized, in the paper review [21], that the current traditional relationship extraction methods are mainly based on statistical language models. In general, there are two categories into which supervised classical machine-learning-based approaches can be divided: kernel-function-based methods [22] and feature-engineering-based methods [23]. Machine-learning-based methods that use feature engineering convert text features into vectors using machine learning algorithms, while methods based on kernel functions calculate entity similarity using kernel functions.

Most recently, deep-learning-based methods, such as graph neural networks (GNN) and remote supervision (distant supervision, DS), have received extensive attention from researchers. Wu et al. [24] summarized the development history, principles, applications, and research directions of graph neural networks. GNN were first proposed by Scarselli et al. [25]. GNN can effectively represent the relationship between entities in the field of relationship extraction. Remote supervision believes that if two entities retrieve a certain relationship in the knowledge base and appear in pairs in a sentence, the sentence must express the relationship between the two entities in a certain way. Graph convolutional networks [26] (GCN) belong to GNN, and GCN introduce convolution during training. Zhang et al. [27] presented a pruned-tree-based graph convolutional network (contextualized graph convolutional networks, C-GCN), which can calculate the shortest path between two entities that may have a relationship. Guo et al. [28] presented an attention-based graph convolutional network (attention guided graph convolutional networks, AGGCN), which can automatically learn and select subgraph structures that are helpful for relation extraction tasks. Jin et al. [29] believed that previous works are

limited to simultaneously identifying the relationship between two entities. The influence of different relationships between entities in this context will be ignored, so it is proposed to use GCN to learn the dependencies between relations. In addition, Wu et al., based on deep reinforcement learning, proposed a method for autonomous target search by UAVs in complex disaster scenes [30].

In general, rule-based, supervised, and semi-supervised methods based on traditional machine learning and supervised methods based on deep learning are well-suited for RE in restricted domains. On the other hand, remotely supervised and unsupervised methods are more appropriate for RE in open domains.

Relation extraction is commonly considered a form of text classification task [14], which can be performed using a text classification model. Large-scale pre-trained language representation models based on transformer [19], such as BERT [15] and RoBERTa [17], have demonstrated exceptional performance on several natural language processing tasks, including text classification. Consequently, these models are also well-suited for relation extraction.

The basic steps to perform relation extraction tasks with large-scale pre-trained language representation models are:

1. Given a text to be classified, each word's embedding in the text is obtained from a large-scale pre-trained language representation model.
2. The first token in the sentence's word embedding is pooled to produce the sentence embedding. Tokens are the output of the text input to the tokenizer. Often the tokenizer of these large-scale pre-trained language representation models will add a starting token to the first part of the sentence. For example, in BERT, it will add the "[cls]" token to the sentence, and in RoBERTa, it will add the "<s>" token to the sentence. The starting token is the first token in the sentence.
3. The final prediction vector for classification is obtained by feeding the sentence embedding from the first token into a wholly connected layer. Typically, the fully connected layer's output dimension is set to the number of relation types, and the resulting vector represents the score of each relation type's assigned input sentence. The relation extracted from the sentence is the one for which the relation type with the best performance was chosen.

2.2. BERT

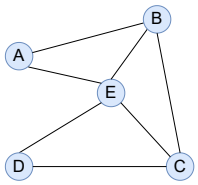
In 2018, Google introduced BERT, a cutting-edge pre-trained language representation model that has gained significant popularity in recent years [15]. The bi-directional transformer, the central component of the BERT architecture, uses self-attention mechanisms to capture complex word relationships [19]. Token, segment, and position embeddings are added together to create the input vector for BERT. The bi-directional transformer generates the resultant work by merging the vectors of the hidden layer. This architecture has demonstrated high performance in assorted tasks relating to natural language processing, including text classification, text generation, and relation extraction.

The introduction of BERT has marked a significant breakthrough regarding natural language processing. Among the most inventive features of the model is its pre-training method, which incorporates two fundamental techniques, namely masked language model (MLM) and next sentence prediction (NSP). MLM masks 10% of the tokens at random, with 10% left unmasked and 80% replaced with "[MASK]". NSP, on the other hand, focuses on the relationship between two sentences, which is essential for tasks such as natural language inference and question answering. Leveraging pre-trained BERT models with remarkable accuracy allows downstream tasks such as named entity recognition to be completed.

2.3. Graph Neural Networks

The graph is a data structure that can effectively handle unstructured data and can be applied in many fields, and Figure 1 shows three structures of graphs. For example,

in social networking, a social network graph can be formed by using users' social relationships to predict the type of users. In e-commerce, the interaction history between users and products can be analyzed to recommend products accurately for users. In citation networks, the citations and cited relationships between documents can be analyzed to classify documents. However, traditional neural networks, such as convolutional neural networks, are difficult to apply on top of graph-structured data due to the complex, diverse, and fickle nature of graph structures. Because each graph has different nodes and each node has a different number of neighboring nodes, the required translation invariance of convolutional neural networks is no longer satisfied, which leads to the fact that neither sampling nor pooling operations can be performed on graph data.



(a) Undirected Graph

	A	B	C	D	E
A	0	0	0	0	1
B	0	0	0	1	1
C	0	0	0	1	1
D	0	1	1	0	1
E	1	1	1	1	0

(b) Adjacency Matrix

	A	B	C	D	E
A	1	0	0	0	0
B	0	2	0	0	0
C	0	0	2	0	0
D	0	0	0	3	0
E	0	0	0	0	4

(c) Degree Matrix

Figure 1. Three representations of graph structures.

Graph-based neural networks have gained popularity in information extraction due to their ability to capture rich dependencies between entities. Specifically, GAT (graph attention network) [31] and GCN (graph convolutional network) [26] are two famous graph neural network models that have been applied in NER and RE. In NER, each word is treated as a graph node, and edges are added between nodes based on their entity types. By contrast, in RE, the graph treats each word as a node, with weighted edges representing the probability of the relationship between each word. The advantage of graph-based neural networks is that they can incorporate contextual information from the entire sentence, thus improving the accuracy of entity recognition and relation extraction.

3. Framework of the Proposed REEGAT Model

Relationship extraction techniques in the context of deep learning likewise no longer need to incorporate feature engineering techniques such as those used in machine learning. Named entity recognition needs to focus more on the meaning of the words themselves, while relationship extraction needs to focus more on the meaning of the sentences themselves. Relationship extraction requires giving more attention to the meaning of the sentence itself. In general, the sentence embedding vector obtained by pooling the word embedding vector of the first token in the sentence inevitably ignores other words' information. If the ignored information is about the entity word, it will have some impact on the prediction performance of the model. Therefore, if only this sentence embedding vector is used as the input vector of the fully connected layer in the final prediction process, there is a certain problem of one-sidedness.

To alleviate the problem of one-sidedness caused by using only the sentence embedding vector as the final fully connected layer input vector, this section proposes the relationship extraction model REEGAT (RoBERTa entity embedding with graph attention networks) based on a graph neural network. The REEGAT model uses the RoBERTa model as the word embedding encoder and the PyTorch embedding model as the relational embedding encoder. The general scheme of the model is shown in Figure 2. The REEGAT model is implemented in two parts to alleviate the one-sidedness problem. The first is to add relationship information to the sentence embedding vector, and the second is to add entity word information to the sentence embedding vector. The specific implementation of the two parts is as follows:

1. Adding relational information: This part first uses the attention mechanism in the graph attention networks (GAT) model to weight the word embedding vectors of the

text. The process is to weight the word embedding vector generated by the RoBERTa model with the relational embedding vector generated by the PyTorch embedding model. Then, the first weighted word embedding vector is used to weight the relational embedding vector. Finally, the first weighted relational embedding vector is used to weight the first weighted word embedding vector. The second weighted word embedding vector contains the relation information. Then, the sentence embedding vector is obtained by pooling the word embedding vector of the first token, which also contains the relation information accordingly.

2. Adding entity word information: The WCL-BBCD model extracts the entities in the sentences. It obtains the word embedding vector of the corresponding word from the quadratic weighted word embedding vector obtained in step (1) according to the entity, called the entity embedding vector. The entity embedding vector is stitched to the end of the sentence embedding vector containing the relational information obtained in step (1), as the new sentence embedding vector contains relational and entity information and has a richer meaning.

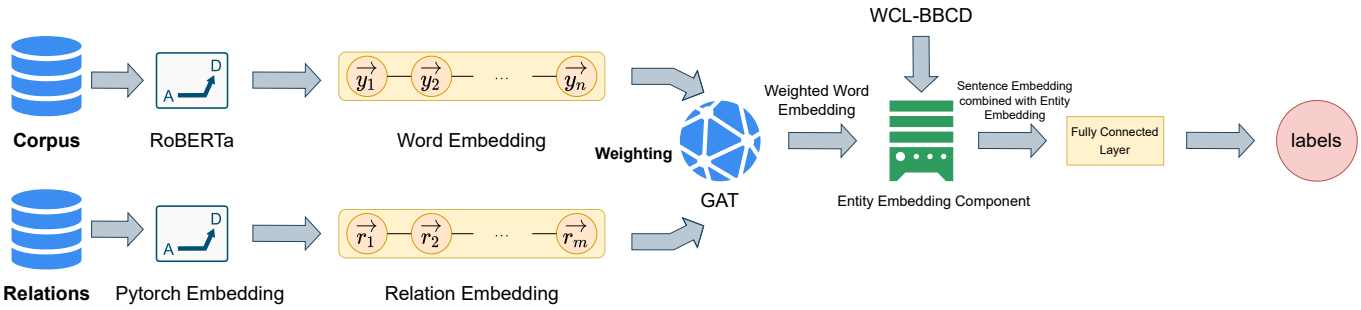


Figure 2. The flowchart of the REEGAT model.

It then uses the sentence embedding vector obtained in step (2) as the final input vector. Using the sentence embedding vector, with richer sentence meaning, to replace the original sentence embedding vector as the final input vector can alleviate the problem of one-sidedness to a certain extent.

The flowchart of the REEGAT model is shown in Figure 2. RoBERTa is employed to produce word embedding, which the GAT model uses as input. The semantic representation of the word embedding is enhanced by the multi-head attention mechanism used by the GAT model to assign weights to the word embedding. Then, the entity embedding component is utilized to obtain the sentence embedding, using the weighted word embedding as input and the specific pooling algorithm. Next, the corresponding entity and its embedding are obtained through the entity recognition result of our proposed WCL-BBCD [32]. Finally, the sentence embedding and entity embedding are stitched to form the final prediction vector.

3.1. Embedding Layer

The first component in the REEGAT model is the embedding layer. The primary purpose of using an embedding layer is to capture words' semantic and syntactic information in a dense, low-dimensional space, which can be easily fed into neural networks for further analysis.

The embedding layer consists of the RoBERTa model and the embedding model. The RoBERTa model is used as the word embedding encoder, and the PyTorch embedding model is used as the relational embedding encoder. The RoBERTa model is consistent with the BERT model, which is also based on the two-way transformer. We chose the RoBERTa model because it has two differences to the BERT model.

The first difference between RoBERTa and BERT is the word segmentation algorithm. BERT utilizes WordPiece as its word segmentation algorithm, while the word segmentation algorithm used by RoBERTa is BPE (byte pair encoding). BPE is implemented as follows:

(1) build a character dictionary based on the corpus; (2) count the most frequent neighboring character pairs in the corpus and update the neighboring pairs to the dictionary; (3) repeat step (2) until the specified number of iterations or the size of the dictionary is reduced to a specific size. The difference between BPE and WordPiece is that WordPiece merges the subwords with the most significant change in the likelihood value of the language model. In contrast, BPE directly merges the two subwords with the highest frequency of occurrence. RoBERTa uses BPE because the pre-training of RoBERTa uses a larger corpus than BERT, which reduces the overhead of calculating the likelihood value in WordPiece. In addition, BPE uses byte encoding to alleviate the "unknown" problem's occurrence effectively.

The second difference between RoBERTa and BERT is the pre-training task, MLM and NSP. The masked token is tagged with a "[MASK]" tag, performed in the data pre-processing stage, so it will only be executed once. In contrast, RoBERTa uses dynamic masking, where random masking is performed again each time a sequence is input to the model, so that even if a tremendous amount of data is input to the model, the model will try different masking strategies until the most suitable one is chosen for the input data.

In summary, these two modifications make the performance of RoBERTa better than that of BERT. Thus, we chose RoBERTa instead of BERT to generate the word embedding.

3.2. Weighting Layer

After the embedding layer comes the weighting layer, which assigns different weights to the features learned by the embedding layer. The primary purpose of using a weighting layer is to allow the model to assign more importance to certain features or words based on their relevance to the task at hand. We compared GAT and GCN in choosing the model components and finally chose GAT. In our requirement, neighboring nodes should have more reasonable dependencies. However, in GCN, it is impossible to assign different weights to the neighboring nodes of a node in the graph, which leads to the inability of GCN to capture spatial information effectively. Furthermore, using GCN requires knowledge of the graph structure before training. This leads to the poor performance of the trained GCN model on other graph structures, which means a poor generalization ability.

GAT (graph attention network) was proposed by Veličković in 2018. It uses the attention mechanism to determine the weighted sum of nearby node features, substituting the fixed function utilized in GCN. The benefit of introducing the attention mechanism is addressing the limitation of GCN, which cannot assign different weights to adjacent nodes of a graph node. By introducing the self-attention mechanism, information between adjacent nodes can be obtained without needing information from the entire graph. This makes the GAT more efficient and effective than GCN in capturing the graph's dependencies and relationships between nodes.

The REEGAT model which we propose comprises two graph attention layers. The graph attention layer uses the model's multi-headed attention mechanism to give equal weights to the relation embedding and word embedding. RoBERTa provides the word embedding, and PyTorch embedding provides the relation embedding. The graph attention layer first uses the relation embedding to weight the word embedding by the multi-headed attention mechanism, and its calculation formula is:

$$WE^1_{attn} = MultiHead(WE, RE, RE) \tag{1}$$

where WE represents the word embedding, RE represents the relation embedding, $MultiHead$ represents the multi-head attention mechanism, and WE^1_{attn} represents the output vector using the first multi-head attention mechanism.

Then, WE^1_{attn} and the word embedding WE are added to obtain the new vector WE_{attn}, and WE is spliced to obtain the new word embedding WE_{concat}. The formula is shown below:

$$\begin{cases} WE_{attn} = WE^1_{attn} + WE \\ WE_{concat} = Concat(WE_{attn}, WE) \end{cases} \tag{2}$$

Then, the dimensionality reduction operation of WE_{concat} is performed through the fully connected layer and activated by the sigmoid-activation function with the following equation:

$$weight = \sigma(Dense(WE_{concat})) \tag{3}$$

where *Dense* represents the fully connected layer, and *weight* represents the weights.

The output range of the sigmoid is (0, 1), so the range of the *weight* is also (0, 1), and the final output is the weighted word embedding WE^1, the calculation formula of which is:

$$WE^1 = weight * WE_{attn} + (1 - weight) * WE \tag{4}$$

The weighted relation embedding is computed similarly to the weighted word embedding, simply by switching the positions of WE and RE in Equation (1) with each other. Since the length of each sentence in a batch is inconsistent, padding complements the sentences that are not long enough. However, the word embedding used for padding impacts the calculation of weights, so adding a mask to the initial calculation of RE_{attn} is necessary. If the mask is set to 1, the multi-headed attention mechanism will change the corresponding part of the key to "-inf". The reason for setting it to "-inf" is that the weight of this part of the key will tend to be 0 after the softmax, which will not affect the weight calculation.

The first graph attention layer results are the weighted word embedding and the relation embedding. Then, the two weighted vectors are input to the second graph attention layer. The second attention layer uses the relation embedding to weight the word embedding by the multi-headed attention mechanism, and the word embedding, after reweighting, is the output of the second attention layer.

3.3. Entity Embedding Component

The entity embedding component is mainly proposed to acquire entity and sentence embedding. In NER, entity embedding is typically generated using methods such as WCL-BBCD, which extracts entities from the training corpus and transforms them into word embedding, thereby obtaining entity embedding. The sentence embedding is derived by pooling the word embedding computed by GAT. The algorithm is shown in Algorithm 1.

Algorithm 1 Pooling Algorithm

Require: WE^2
Ensure: PoE_i
1: $WE_0^2 = WE^2[:, 0, :]$ // Take the word embedding of the first token
2: $WE_0^2 = Dropout(WE_0^2)$
3: $WE_0^2 = Dense(WE_0^2)$
4: $WE_0^2 = tanh(WE_0^2)$
5: $PoE_i = Dropout(WE_0^2)$

The following is a description of the algorithm. The Pooling Algorithm is a method used for extracting features from word embeddings in natural language processing tasks. Given a matrix of word embeddings WE^2, the algorithm selects the embedding of the first token in the sequence, applies dropout regularization to it, and then passes it through a fully connected layer followed by a hyperbolic tangent activation function. Finally, the output is again subject to dropout regularization, resulting in the final pooled embedding PoE_i.

4. Experiments

4.1. Experimental Setup

In this work, we run experiments utilizing the SemEval-2010 Task 8 dataset [33] and the Wiki80 dataset [34]. The SemEval-2010 Task 8 dataset defines a task of relation extraction, that is, given two labeled words and sentences containing them, predict the proper relation

from a given list of relation types. The dataset defines a total of "9+1" relations, in which each relation in the "9" relations can be subdivided into 2 relations according to the order of labeling nouns. For example, in the "9" relations, the causal relation can be further subdivided into 2 relations: "A causes B (cause–effect (e1, e2))" or "B causes A (cause–effect (e2, e1))". In this paper, we continue to use the official divided training set for training and the divided test set for testing. There are 8000 pieces of data in the training set and 2717 pieces in the test set. The Wiki80 dataset is extracted from the FewRel dataset by the NLP team at Tsinghua University. This dataset is also appropriate for the relation extraction task and is refined by hand without noise. In the Wiki80 dataset, 80 relations are defined. The officially partitioned training set contains 630 training data for each relation, and the test set contains 70 test data for each relation.

We use *Precision*, *Recall*, and *F*1 score as the evaluation standards for the experimental data when assessing the results.

A single machine with a single GPU, an Intel(R) Xeon(R) CPU running at 2.20 GHz with 25 GB of memory, and an NVIDIA Tesla P100 with 16 GB of memory, make up the experimental hardware environment for this paper.

4.2. Evaluation Metrics

In relation extraction, the model's performance is typically evaluated using the *Precision*, *Recall*, and *F*1 score metrics. Based on the number of true positives (TP), false positives (FP), and false negatives (FN) in the data, these metrics are computed. TP is the proportion of samples that were successfully identified using the relation type, while FP is a measure of how many samples have been incorrectly associated with a relation type. FN is the amount of samples that should have been associated with a relation type but were not, in terms of number of samples. By computing these values, we can judge whether the model is accurate and thorough in classifying relation types.

The *precision* is the proportion of correctly identified samples to all samples in a set of data.

The *Precision* can be calculated using the following formula:

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Recall is defined as the proportion of correctly classified samples to all of the actual positive samples in a dataset. Hence, the calculation method for *Recall* can be obtained as follows:

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

According to the definition of the *F*1 score, it is a comprehensive index that balances the effects of both *Precision* and *Recall* and represents the harmonic mean of *Precision* and *Recall*. Thus, the following can be deduced as the *F*1 score calculation method:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{7}$$

For the above evaluation indicators, there are two calculation methods, namely "macro-averaged" and "micro-averaged". The *Precision*, *Recall*, and *F*1 score for each category individually are calculated for the macro-average, and then the average of all categories is found concerning the three indicators. Micro-average is used to calculate the *Precision*, *Recall*, and *F*1 score for the overall data, without distinction of the category. In this paper, the evaluation indicators used in RE are the macro-average *Precision*, *Recall*, and *F*1 score. The SemEval-2010 Task 8 dataset is evaluated using the script "semeval2010_task8_scorer-v1.2.pl".

4.3. Baseline Models

In order to verify the validity of the REEGAT model proposed in this paper, this subsection conducts comparison experiments on the SemEval-2010 Task 8 and Wiki80 datasets and analyzes the experimental results. The comparison models set up for the experiments are:

1. GCN [27]: GCN introduces convolution in the original graph neural network.
2. C-GCN [27]: C-GCN uses BiLSTM to extract vectors containing semantic and syntactic information and uses them as input to the GCN.
3. AGGCN [28]: AGGCN is a graph convolutional neural network (GCN) that utilizes a multi-headed attention mechanism to transform a traditional graph into a fully weighted connected graph, enabling it to obtain information about neighbor nodes with different weights.
4. BERT: Basic model.
5. RoBERTa: RoBERTa, used in the REEGAT model, is the word embedding model.
6. R-BERT [35]: R-BERT is based on BERT, to combine the target entity information to process the relation extraction task.
7. R-RoBERTa: R-RoBERTa is based on RoBERTa, to combine the target entity information to process the relation extraction task.
8. RIFRE [36]: RIFRE is based on BERT by adding a heterogeneous graph neural network to mine the possible relations between entities.
9. ROFRE: ROFRE is constructed from RIFRE by replacing the word embedding model from BERT with RoBERTa. ROFRE is constructed by us for the purpose of comparison, it has not been published elsewhere.
10. REEGAT: Model proposed in this paper.

4.4. Relation Extraction Results

The experiments related to RE were conducted in the aforementioned experimental environment. The obtained results on the SemEval-2010 Task 8 and Wiki80 datasets are presented in Table 1.

Table 1. Experimental results on the SemEval-2010 Task 8 and Wiki80 datasets.

Model	SemEval-2010 Task 8			Wiki80		
	P	**R**	**F1**	**P**	**R**	**F1**
GCN	81.58	84.14	82.84	76.29	75.66	75.97
C-GCN	83.02	85.86	84.42	79.73	78.84	79.28
AGGCN	83.90	87.48	85.65	68.30	62.96	65.52
BERT	88.05	89.87	88.92	64.17	64.84	64.50
RoBERTa	88.77	89.73	89.23	63.43	64.20	63.81
R-BERT	88.37	90.14	89.23	86.93	87.00	86.96
R-RoBERTa	89.56	90.11	89.81	87.27	87.27	87.27
RIFRE	90.21	92.26	91.21	86.66	86.73	86.69
ROFRE	91.12	91.45	91.27	87.04	87.07	87.05
REEGAT	**91.38**	**92.55**	**91.95**	**87.61**	**87.66**	**87.64**

REEGAT incorporates the target entity information obtained by WCL-BBCD, proposed in our previous paper [32], while REEGAT also combines the idea of adding graph neural networks in RIFRE, and changes the attention mechanism used in RIFRE from a self-attention mechanism to a multi-headed attention mechanism, and the above strategies effectively improve the prediction accuracy of RE. On the SemEval-2010 Task 8 dataset, the prediction accuracy of RIFRE is higher than that of R-BERT, the proposed REEGAT in

this work obtains the greatest *Precision*, *Recall*, and *F*1 scores, which are 1.30%, 0.31%, and 0.81% better than those of RIFRE, respectively. While on the Wiki80 dataset, the prediction accuracy of R-BERT is higher than that of RIFRE, and the *Precision*, *Recall*, and *F*1 scores of REEGAT proposed in this paper are improved by 0.78%, 0.76%, and 0.78%, respectively, compared with R-BERT.

Compared with the BERT model, the graph neural-network-based relation extraction model REEGAT proposed in this paper has three differences. First, it uses the RoBERTa model to replace the BERT model to train the word embedding vector; second, it proposes using the graph attention network model to weight the word embedding vector using the relational embedding vector; third, it proposes using the entity embedding component to stitch the entity embedding vector and the sentence embedding vector to form a new input vector. To verify the role of each component in REEGAT, ablation experiments are conducted in this paper. In addition, we add a ranking in the last column of the table, indicating the ranking of the total scores of *Precision*, *Recall*, and *F*1 score for each model in the previous section, to demonstrate the performance of GAT and the presence or absence of the entity embedding component on the improvement of the results. The components included in the comparative model in this paper are shown in Table 2. The following is an analysis of the experimental findings from the ablation experiments:

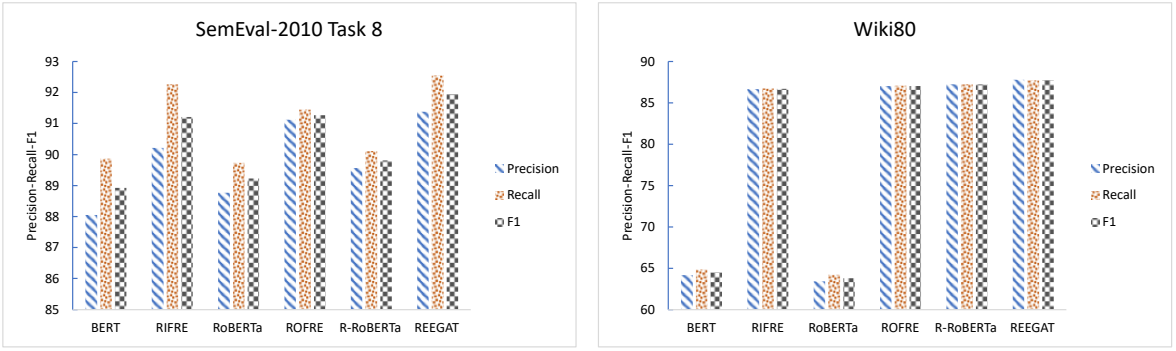
Table 2. The components included in the models and their performance.

Model	GAT	EEC *	SemEval-2010 Task 8			Wiki80		
			P	R	F1	P	R	F1
BERT	×	×	88.05	89.87	88.92	64.17	64.84	64.50
RoBERTa	×	×	88.77	89.73	89.23	63.43	64.20	63.81
R-BERT	×	√	88.37	90.14	89.23	86.93	87.00	86.96
R-RoBERTa	×	√	89.56	90.11	89.81	87.27	87.27	87.27
RIFRE	√	×	90.21	92.26	91.21	86.66	86.73	86.69
ROFRE	√	×	91.12	91.45	91.27	87.04	87.07	87.05
REEGAT	√	√	91.38	92.55	91.95	87.61	87.66	87.64

* EEC stands for entity embedding component.

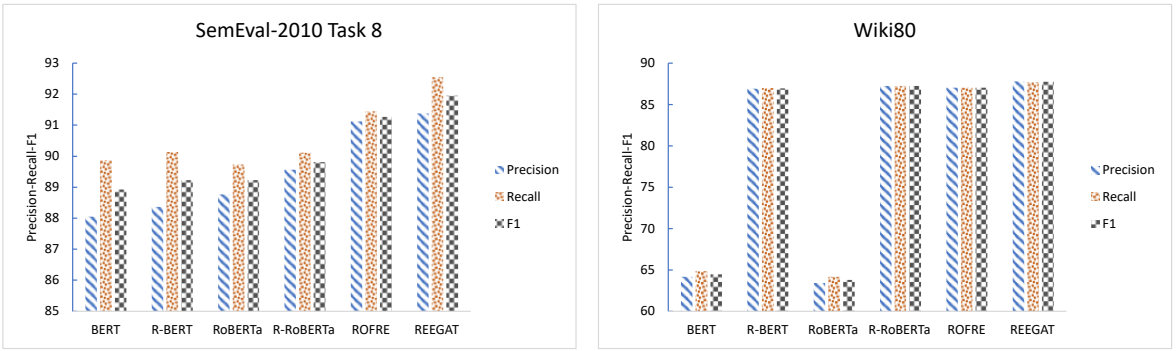
1. The experiment outcomes show that GAT can significantly increase the prediction accuracy. Specifically, RIFRE, which utilizes GAT, outperforms BERT. Similarly, the performance of ROFRE, which uses GAT, is better than that of RoBERTa, while the performance of REEGAT, which incorporates GAT, is better than that of R-RoBERTa. Compared with BERT, the *Precision*, *Recall*, and *F*1 indicators of RIFRE are improved by 2.45%, 2.66%, and 2.58% on the SemEval-2010 Task 8 dataset and 35.05%, 33.76%, and 34.40% on the Wiki80 dataset, respectively. Compared with RoBERTa, the *Precision*, *Recall*, and *F*1 indicators of ROFRE are improved by 2.65%, 1.92%, and 2.29% on the SemEval-2010 Task 8 dataset and 37.22%, 35.62%, and 36.42% on the Wiki80 dataset, respectively. Compared with R-RoBERTa, the *Precision*, *Recall*, and *F*1 of REEGAT are improved by 2.03%, 2.71%, and 2.38% on the SemEval-2010 Task 8 dataset and 0.39%, 0.45% and 0.42% on the Wiki80 dataset, respectively. The visualization results of this part of the experiment are shown in Figure 3.
2. The effectiveness of the entity embedding component in improving the prediction accuracy can be inferred from the experimental results. Specifically, R-BERT outperforms BERT, R-RoBERTa outperforms RoBERTa, and REEGAT outperforms ROFRE. These findings suggest that incorporating entity embeddings can improve the ability of models to identify and extract relationships between entities. Compared with BERT, the *Precision*, *Recall*, and *F*1 indicators of R-BERT are improved by 0.36%, 0.30%, and 0.35% on the SemEval-2010 Task 8 dataset and 35.47%, 34.18%, and 34.82% on the Wiki80 dataset, respectively. Compared with RoBERTa, the *Precision*, *Recall*,

and *F*1 indicators of R-RoBERTa are improved by 0.89%, 0.42%, and 0.65% on the SemEval-2010 Task 8 dataset and 37.58%, 35.93%, and 36.77% on the Wiki80 dataset, respectively. Compared with ROFRE, the *Precision*, *Recall*, and *F*1 of REEGAT are improved by 0.29%, 1.20%, and 0.75% on the SemEval-2010 Task 8 dataset and 0.65%, 0.68%, and 0.68% on the Wiki80 dataset, respectively. The visualization results of this part of the experiment are shown in Figure 4.



(a) SemEval-2010 Task 8 (b) Wiki80

Figure 3. Ablation experiment results of the GAT on the SemEval-2010 Task 8 and Wiki80 datasets.



(a) SemEval-2010 Task 8 (b) Wiki80

Figure 4. Ablation experiment results of the entity embedding component on the SemEval-2010 Task 8 and Wiki80 datasets.

In summary, the ablation experiments show that the model's prediction accuracy can be significantly improved by using both the GAT and the entity embedding component.

4.5. Hyperparameter Study

Different hyperparameters can significantly affect the model's prediction accuracy, and selecting appropriate hyperparameters is crucial for achieving optimal performance. In this section, we experiment with the SemEval-2010 Task 8 dataset to investigate the impact of hyperparameters on the experimental results. The experiments follow a controlled approach, where the number of heads in the multi-head attention mechanism is varied while keeping other hyperparameters constant. The aim is to observe the effects of this specific parameter on the model's performance. As shown in Figure 5, the results indicate that for the SemEval-2010 Task 8 dataset, the REEGAT model achieved the highest macro-average *F*1 score (91.95) when using the multi-head attention mechanism with two heads. Specifically, the macro-average *Precision* peaked (91.38) with two heads, and then decreased as the number of heads increased. In contrast, the macro-average *Recall* first decreased and then increased, peaking (93.07) when the number of heads was 16.

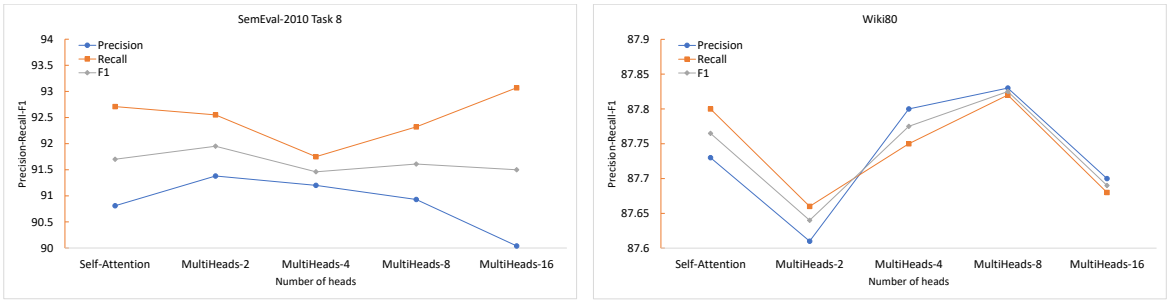


Figure 5. The effect of the number of heads on experimental outcomes.

As illustrated in Figure 5, the trends of the macro-average *Precision*, macro-average *Recall*, and macro-average *F*1 score of the Wiki80 dataset are consistent with an initial decrease, followed by an increase, and ultimately a decrease, as the quantity of heads grows. The peaks, which were 87.83, 87.82, and 87.82, respectively, were reached when the number of heads was eight.

5. Discussion

The proposed model has outperformed earlier state-of-the-art models and achieved the best results in all three evaluation metrics used in the experiments, compared to earlier models. This is true for both the SemEval-2010 Task 8 and Wiki80 datasets. These findings support our working hypothesis. Using the GAT model to strengthen the connection between the word embedding vector and the relational embedding vector and the method of entity embedding can alleviate the current one-sidedness problem. The experiments demonstrate that using these methods to alleviate the one-sidedness problem can effectively improve the accuracy of RE.

From a broader perspective, our study's practical contributions to the field of RE have important implications for various real-world applications. For instance, the proposed model's accuracy and efficiency can aid in extracting relationships between entities in diverse texts, such as biomedical literature, news articles, and social media posts. The improved performance of our model could also help to automate tasks that require analyzing a large volume of text data, such as sentiment analysis or topic modeling. Our research provides a valuable tool for professionals in various industries who rely on accurate and efficient RE.

Of course, the REEGAT model proposed in this paper still has certain limitations. The graph neural-network-based relation extraction model proposed in this paper currently only considers the relations between two entities. In practical applications, a sentence likely contains multiple entities and there are relationships between multiple entities. Therefore, to improve the generality of the model, it is necessary to consider the relationship extraction between multiple entities in the future.

In addition, Michel et al. [37] proposed a new method for selecting the optimal number of attention heads according to the task and available computing resources. Sometimes, the number of heads only affects the model's performance slightly. In this paper, different numbers of heads are tested in the experiment to obtain the results. We believe introducing the new method proposed by Paul can further optimize our results under limited computing power. In our future research, we will try to add this method to our experiments to obtain more efficient results.

Moving forward, there are several future directions for research in this field. One area of interest in NER research is using contrastive learning to enhance the quality of entity embeddings. Further exploration of the use of GAT in RE can be conducted, potentially incorporating additional features such as syntactic and semantic information. Finally, the representation of entities and their relationships can be improved using prior knowledge, such as knowledge graphs, in word embedding research.

In summary, our proposed model outperformed state-of-the-art models and achieved impressive results in all three evaluation metrics used in our experiments. Our findings reinforce the effectiveness of entity embedding and GAT in enhancing the accuracy of RE models. Moreover, our research's contribution to the field of RE extends to its potential to improve the performance of automated text analysis tasks in various industries. The impact of our research highlights the need for further exploration of entity embedding and GAT in RE and the potential for enhancing word embeddings with prior knowledge.

6. Conclusions

This paper proposes and verifies the effectiveness of the REEGAT model in performing relation extraction tasks. REEGAT mainly uses a GAT and an entity embedding component to enrich the meaning expressed in the sentence embedding as the prediction vector, effectively alleviating the problem of one-sidedness of using only the sentence embedding as the final prediction vector. The proposed model in this paper has been experimentally verified and compared, demonstrating superior performance on the SemEval-2010 Task 8 and Wiki80 datasets. Additionally, of the three evaluation indicators used in the experiment, it produced the best results.

In the future, we will investigate the potential of contrastive learning in the RE field. Additionally, we may be able to increase the efficiency of various intelligent information extraction services provided through intelligent communication systems, including named entity recognition and relation extraction, by incorporating knowledge graphs into word embedding representation.

Author Contributions: Conceptualization, Q.H., R.Z. and N.X.; Methodology, F.C., Q.H. and R.Z.; Validation, Q.H.; Formal analysis, Q.H., R.Z. and N.X.; Investigation, F.C.; Resources, N.X.; Writing—original draft, F.C. and Q.H.; Writing—review & editing, R.Z. and N.X.; Visualization, N.X.; Supervision, R.Z.; Project administration, R.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Technology Research and Development Program of China under grant no. 2022YFB3105401.

Data Availability Statement: Publicly available datasets were analyzed in this study. The SemEval-2010 Task 8 dataset can be found here: http://www.kozareva.com/downloads.html and the Wiki80 dataset can be found here: https://github.com/thunlp/OpenNRE/blob/master/benchmark/download_wiki80.sh.

Acknowledgments: The authors are grateful to the anonymous reviewers for their invaluable comments and suggestions.

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

Abbreviations

Abbreviated description of symbols.

Abbreviations	Meaning
RoE	*RoBERTa* input vector in the model about sentence S
WE^1_{attn}	*RoBERTa* model about the word embedding vector of sentence S
WE	*RoBEARTa* model about the word embedding vector of sentence S output
RE	Target relationship embedding vector generated based on the number of relationship categories
WE_{attn}	The vector obtained by summing WE^1_{attn} and WE
WE_{concat}	The combined word embedding vector obtained by splicing WE_{attn} and WE
$Concat$	Vector/matrix stitching by row
$Dense$	Full connectivity layer used
WE^1	The output vector of the word embedding vector of sentence S after one weighting calculation

WE^2	The output vector of the word embedding vector of sentence S after two weighting calculations
PoE_i	Sentence embedding vector
WE_0^2	Word embedding vector in WE_i^2 for the first "<s>" token of sentence S_i
$Dropout$	$Dropout$ mechanism

References

1. Arora, S.; Lewis, P.; Fan, A.; Kahn, J.; Ré, C. Knowledge Retrieval over Public and Private Data. 2023. Available online: https://knowledge-nlp.github.io/aaai2023/papers/003-PQA-oral.pdf (accessed on 8 May 2023).
2. Liang, W.; Yang, Y.; Yang, C.; Hu, Y.; Xie, S.; Li, K.; Cao, J. Pdpchain: A consortium blockchain-based privacy protection scheme for personal data. *IEEE Trans. Reliab.* **2022**. [CrossRef]
3. Ahmadian, S.; Ahmadian, M.; Jalili, M. A deep learning based trust-and tag-aware recommender system. *Neurocomputing* **2022**, *488*, 557–571. [CrossRef]
4. Zhou, R.; Liu, C.; Wan, J.; Fan, Q.; Ren, Y.; Zhang, J.; Xiong, N. A Hybrid Neural Network Architecture to Predict Online Advertising Click-Through Rate Behaviors in Social Networks. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 3061–3072. [CrossRef]
5. Li, Y.; Liang, W.; Peng, L.; Zhang, D.; Yang, C.; Li, K. Predicting drug-target interactions via dual-stream graph neural network. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2022**, 1–11. [CrossRef]
6. Guo, Z.; Nan, G.; Lu, W.; Cohen, S.B. Learning latent forests for medical relation extraction. In Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, Yokohama, Japan, 7–15 January2021; pp. 3651–3657.
7. Chen, Y.; Hou, W.; Li, S.; Wu, C.; Zhang, X. End-to-end emotion-cause pair extraction with graph convolutional network. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020; pp. 198–207.
8. Qiu, J.; Chai, Y.; Liu, Y.; Gu, Z.; Li, S.; Tian, Z. Automatic non-taxonomic relation extraction from big data in smart city. *IEEE Access* **2018**, *6*, 74854–74864. [CrossRef]
9. Yao, Y.; Xiong, N.; Park, J.H.; Ma, L.; Liu, J. Privacy-preserving max/min query in two-tiered wireless sensor networks. *Comput. Math. Appl.* **2013**, *65*, 1318–1325. [CrossRef]
10. Conlon, S.J.; Abrahams, A.S.; Simmons, L.L. Terrorism information extraction from online reports. *J. Comput. Inf. Syst.* **2015**, *55*, 20–28. [CrossRef]
11. Atkinson, M.; Piskorski, J.; Tanev, H.; van der Goot, E.; Yangarber, R.; Zavarella, V. Automated event extraction in the domain of border security. In *International Conference on User Centric Media*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 321–326.
12. Chen, Y.; Zhou, L.; Pei, S.; Yu, Z.; Chen, Y.; Liu, X.; Du, J.; Xiong, N. KNN-BLOCK DBSCAN: Fast clustering for large-scale data. *IEEE Trans. Syst. Man, Cybern. Syst.* **2019**, *51*, 3939–3953. [CrossRef]
13. Giorgi, J.; Wang, X.; Sahar, N.; Shin, W.Y.; Bader, G.D.; Wang, B. End-to-end named entity recognition and relation extraction using pre-trained language models. *arXiv* **2019**, arXiv:1912.13415.
14. Li, D.; Zhang, Y.; LI, D.; Lin, D. Review of Entity Relation Extraction Methods. *J. Comput. Res. Dev.* **2020**, *57*, 1424.
15. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2 June 2019; pp. 4171–4186.
16. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
17. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv* **2019**, arXiv:1907.11692.
18. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5753–5763.
19. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 6000–6010.
20. Nakashole, N.; Weikum, G.; Suchanek, F. PATTY: A taxonomy of relational patterns with semantic types. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, Republic of Korea, 12–14 July 2012; pp. 1135–1145.
21. Zhang, Q.; Chen, M.; Liu, L. A Review on Entity Relation Extraction. In Proceedings of the 2017 Second International Conference on Mechanical, Control and Computer Engineering (ICMCCE), Harbin, China, 8–10 December 2017; pp. 178–183. [CrossRef]
22. Zelenko, D.; Aone, C.; Richardella, A. Kernel methods for relation extraction. *J. Mach. Learn. Res.* **2003**, *3*, 1083–1106.
23. Xu, Y.; Hong, K.; Tsujii, J.; Chang, E.I.C. Feature engineering combined with machine learning and rule-based methods for structured information extraction from narrative clinical discharge summaries. *J. Am. Med. Inform. Assoc.* **2012**, *19*, 824–832. [CrossRef] [PubMed]

24. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P.S. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 4–24. [CrossRef]
25. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef]
26. Chiang, W.L.; Liu, X.; Si, S.; Li, Y.; Bengio, S.; Hsieh, C.J. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 257–266.
27. Zhang, Y.; Qi, P.; Manning, C.D. Graph convolution over pruned dependency trees improves relation extraction. *arXiv* **2018**, arXiv:1809.10185.
28. Guo, Z.; Zhang, Y.; Lu, W. Attention guided graph convolutional networks for relation extraction. *arXiv* **2019**, arXiv:1906.07510.
29. Jin, Z.; Yang, Y.; Qiu, X.; Zhang, Z. Relation of the relations: A new paradigm of the relation extraction problem. *arXiv* **2020**, arXiv:2006.03719.
30. Wu, C.; Ju, B.; Wu, Y.; Lin, X.; Xiong, N.; Xu, G.; Li, H.; Liang, X. UAV autonomous target search based on deep reinforcement learning in complex disaster scene. *IEEE Access* **2019**, *7*, 117227–117245. [CrossRef]
31. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
32. Zhou, R.; Hu, Q.; Wan, J.; Zhang, J.; Liu, Q.; Hu, T.; Li, J. WCL-BBCD: A Contrastive Learning and Knowledge Graph Approach to Named Entity Recognition. *arXiv* **2022**, arXiv:2203.06925.
33. Hendrickx, I.; Kim, S.N.; Kozareva, Z.; Nakov, P.; Séaghdha, D.Ó.; Padó, S.; Pennacchiotti, M.; Romano, L.; Szpakowicz, S. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In Proceedings of the 5th International Workshop on Semantic Evaluation, Los Angeles, CA, USA, 15–16 July 2010; pp. 33–38.
34. Han, X.; Gao, T.; Yao, Y.; Ye, D.; Liu, Z.; Sun, M. OpenNRE: An Open and Extensible Toolkit for Neural Relation Extraction. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations, Hong Kong, China, 3–7 November 2019; pp. 169–174.
35. Wu, S.; He, Y. Enriching pre-trained language model with entity information for relation classification. In Proceedings of the Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 2361–2364.
36. Zhao, K.; Xu, H.; Cheng, Y.; Li, X.; Gao, K. Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. *Knowl. Based Syst.* **2021**, *219*, 106888. [CrossRef]
37. Michel, P.; Levy, O.; Neubig, G. Are Sixteen Heads Really Better than One? In *Advances in Neural Information Processing Systems*; Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., Eds. Curran Associates, Inc.: New York, NY, USA, 2019; Volume 32.