symmetry

MDPI

Article

A Feature Combination-Based Graph Convolutional Neural Network Model for Relation Extraction

Jinling Xu [1,†], Yanping Chen [1,*,†], Yongbin Qin [1], Ruizhang Huang [1] and Qinghua Zheng [2]

[1] College of Computer Science and Technology, Guizhou University, Guiyang 550025, China; jinlingxu06@outlook.com (J.X.); ybqin@gzu.edu.cn (Y.Q.); rzhuang@gzu.edu.cn (R.H.)
[2] School of Automation Science and Engineering, Xi'an Jiaotong University, Xi'an 710049, China; qhzheng@mail.xjtu.edu.cn
* Correspondence: ypench@gmail.com
† These authors contributed equally to this work.

Abstract: The task to extract relations tries to identify relationships between two named entities in a sentence. Because a sentence usually contains several named entities, capturing structural information of a sentence is important to support this task. Currently, graph neural networks are widely implemented to support relation extraction, in which dependency trees are employed to generate adjacent matrices for encoding structural information of a sentence. Because parsing a sentence is error-prone, it influences the performance of a graph neural network. On the other hand, a sentence is structuralized by several named entities, which precisely segment a sentence into several parts. Different features can be combined by prior knowledge and experience, which are effective to initialize a symmetric adjacent matrix for a graph neural network. Based on this phenomenon, we proposed a feature combination-based graph convolutional neural network model (FC-GCN). It has the advantages of encoding structural information of a sentence, considering prior knowledge, and avoiding errors caused by parsing. In the experiments, the results show significant improvement, which outperform existing state-of-the-art performances.

Keywords: relation extraction; feature combination; graph neural network; nlp

Citation: Xu, J.; Chen, Y.; Qin, Y.; Huang, R.; Zheng, Q. A Feature Combination-Based Graph Convolutional Neural Network Model for Relation Extraction. *Symmetry* **2021**, *13*, 1458. https://doi.org/10.3390/sym13081458

Academic Editors: Mariano Torrisi and Jan Awrejcewicz

Received: 15 June 2021
Accepted: 4 August 2021
Published: 9 August 2021

1. Introduction

Relation extraction means extracting entity-relationship triples from unstructured text. For example, in "`teams of nurses and doctors were seen in packed emergency rooms attending to the wounded`", the entity "`packed emergency rooms`" and the entity "`the wounded`" belong to a "`physical (PHYS)`" relationship. Relation extraction can transform unstructured text into structured data. It can promote the automatic construction of a knowledge base [1], understand user query intentions, and improve the search efficiency of search engines.

At present, neural network models are used by more and more researchers. It can well-capture the semantic information in a sentence. Because a sentence usually contains several named entities, it is also important to capture structural information of a sentence for supporting relation extraction. For example, He et al. [2] used latent structural information and semantic features to optimize the representation of discourse arguments, which can enhance the semantic understanding of implicit relationships. More recently, more and more researchers have been using graph neural network models for relation extraction, and have made a lot of progress. One of the important steps of using graph neural networks for relation extraction is to generate an adjacency matrix. Most researchers use dependency trees to generate adjacent matrices for encoding structural information of a sentence. For instance, Sun et al. [3] converted a sentence-based dependency tree into a directed graph to consider the structural information in the tree.

However, on the one hand, this kind of graph neural network model based on dependency trees has two problems. One is that it often has poor performance by inaccurate

chunking or parsing; the other is that the prior knowledge that can be obtained through our previous experience is not convenient to be used. On the other hand, in our normal cognitive process, when we see a sentence, our brain will form a graph structure based on prior knowledge of the sentence. Then, after relevant processing by our brain, we can get the meaning of the sentence. More specifically, if this sentence contains two entities that we want to judge the relationship, we will get the relationship between these two entities in this sentence.

Moreover, a sentence is structuralized by two named entities, which precisely segment a sentence into several parts. Different features can be obtained by prior knowledge and experience, which are not only effective to initialize a graph for graph neural networks, but also capture the structural information of the sentence. Additionally, it is easy to cause the problem of feature sparseness for the relation extraction basically based on a sentence with a few words. In order to solve the problem, Chen et al. [4] constructed a set space model, which uses language characteristics to combine the features in the sentence into different sets. Thus, in this paper, we also use this method to obtain more features of the sentence. Moreover, we make some improvements on the basis of the atomic features and combined features in Chen et al. [4]. Next, we use the combined features to initialize a graph and apply it to the graph convolutional neural network (GCN). Therefore, a feature combination-based graph convolutional neural network model for relation extraction is proposed by us. In this way, we not only make good use of sentence structural information and prior knowledge, but also avoid the above-mentioned problems caused by the dependency trees.

We tested the performance of the model on the ACE05 English dataset, CoNLL04 dataset, and SciERC dataset. Experiments show that our model can achieve better performance for these datasets.

The contributions of this paper can be divided into the following points:

- Our methods to generate combined features are used to initialize an adjacent matrix for a graph neural network. It is effective at capturing the structural information of a sentence.
- Based on a graph convolutional neural network, a deep architecture is designed to support relation extraction. It outperforms existing state-of-the-art performance.

The rest of this paper is organized as follows. Section 2 shows related work. Section 3 elaborates on the construction process of our model. The details related to the experiment are shown in Section 4. Section 5 presents our conclusion and future work.

2. Related Work

At present, the neural network model has achieved great improvement in relation extraction tasks. The main neural networks used are the Convolutional Neural Network (CNN) [5–7] and Recurrent Neural Network (RNN) [8–10]. Moreover, according to whether the extracted entity pairs span sentences, the relation extraction model can be divided into a sentence-level relation extraction model [11] and cross-sentence relation extraction model [12]. There are also some researchers that perform document-level relation extraction [13,14]. Because an entity pair may have multiple relationships, some people conduct research on an entity pair corresponding to only one relationship, while some people take into account the existence of multiple relationships in the entity pair for relation extraction [15]. In this paper, we only consider one entity pair corresponding to one relationship.

In the deep learning framework, because the neural network model is weak in obtaining the structural information of the sentence, position information is often used to capture sentence structural information. For example, Zeng et al. [7] added position information for distant supervision relation extraction. Veyseh et al. [10] not only considers the word embedding and position embedding, but also adds the entity tag embedding for relation extraction. Shen and Huang [16] combines the attention mechanism and position information to obtain sentence structural information.

More recently, many researchers have used sentence-based dependency trees to generate a graph for encoding structural information of the sentence, then to apply a graph neural network to perform relation extraction. One of the important steps to construct a graph neural network model is to construct an adjacency matrix, or in other words, to obtain a graph. At present, Guo et al. [17], Vashishth et al. [18] and Fu and Ma [19] inputted the entire tree into graph convolutional neural networks to perform relation extraction. Zhang et al. [20] used two entities in the Least Common Ancestor (LCA) subtree of the dependency tree to construct the graph with nodes that do not exceed the K distance. In view of the above construction of a complete graph based on the dependency tree, the structural information in the dependency tree is not considered. Sun et al. [3] established a directed weighted graph while retaining the structure of the information depending on the tree.

In addition to using dependency trees to construct graphs of sentences, there are others who directly construct graphs on the sentences. Vashishth et al. [21] regard entities as nodes, and construct a multi-relation graph to predict multiple relations between entities. Sun et al. [22] use entities and entity pairs to construct a bipartite graph for joint extraction of entities and relationship types. However, the above-mentioned processes of constructing a graph cannot make good use of prior knowledge. Moreover, when the graph is obtained using a dependency tree, because the dependency tree is generated by an external tool (e.g., StanfordCoreNLP) which is error-prone, this may lead to poor performance of the model. Therefore, in our experiments, we do not use dependency trees to build graphs, but to build undirected graphs based on features related to two entities, which avoid the problems caused by dependency trees.

3. Model

In this part, we will introduce the FC-GCN model in detail. The overall model diagram is shown in Figure 1.

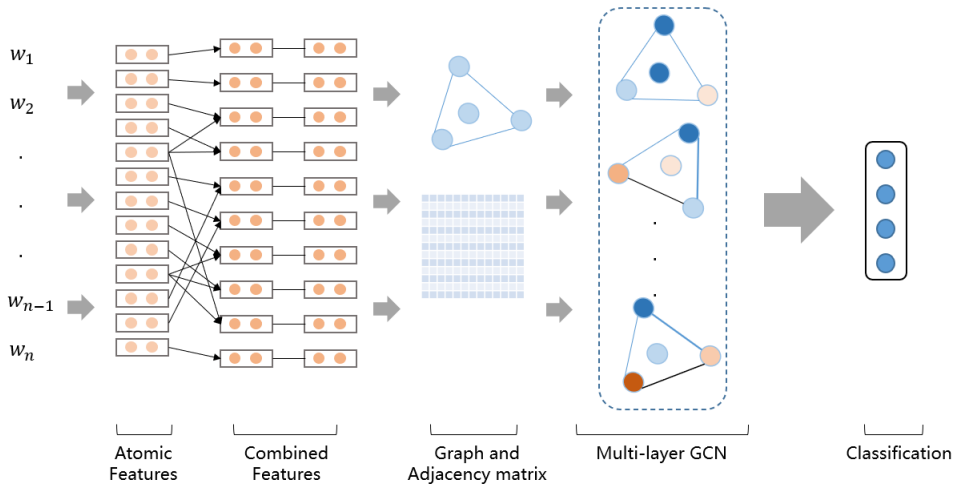

Figure 1. Overview of our FC-GCN model.

3.1. The Atomic and Combined Features

Combined features can well-capture the structural information in sentences [23] and are widely used in relation extraction tasks. It is generally constructed by using syntactic and semantic rules. In this paper, we have made further improvements on the basis of the features related to two entities in Chen et al. [4]. Below, we first introduce the features which we used in Chen et al. [4], and it includes the entity type, entity subtype, part of speech (POS) of the word on the left and right of the entity, entity heads tag, and relative positions of two entities. Additionally, we also regard these features as atomic features.

Given a sentence $s = [w_1, w_2, ...]$ (w_j is a word) and two entities e_1 and e_2 in it, let e_i represent the i-th entity (e_1 or e_2). In order to obtain the above-mentioned atomic features,

we assume that the functions $\texttt{RightPosOf}(e_i)$, $\texttt{LeftPosOf}(e_i)$, $\texttt{TypeOf}(e_i)$, $\texttt{SubTypeOf}(e_i)$, $\texttt{HeadOf}(e_i)$, $\texttt{PositionBetween}(e_1, e_2)$, respectively, can obtain the POS tags of the word on the right and left of e_i, the type of e_i, the subtype of e_i, the head tag of e_i, and the relative position of e_1 and e_2. In that way, we can get 11 atomic features for every two entities as follows:

$$\begin{aligned} x_1 &= RightPosOf(e_1) & x_7 &= SubTypeOf(e_1) \\ x_2 &= LeftPosOf(e_1) & x_8 &= SubTypeOf(e_2) \\ x_3 &= RightPosOf(e_2) & x_9 &= HeadOf(e_1) \\ x_4 &= LeftPosOf(e_2) & x_{10} &= HeadOf(e_2) \\ x_5 &= TypeOf(e_1) & x_{11} &= PositionBetween(e_1, e_2) \\ x_6 &= TypeOf(e_2) \end{aligned}$$

where x_k represents the k-th atomic feature. In addition to POS tags and relative position features, the remaining other features are marked in the corresponding corpus. As for the POS tags and relative position features, we obtained them through external tools (e.g., nltk) and manual annotations, respectively. Moreover, there are four kinds of relative position information, which are e_1 in front of e_2 (marked as 1), e_2 in front of e_1 (marked as 2), e_1 contains e_2 (marked as 3), and e_2 contains e_1 (marked as 4).

Because we consider that two entities are also crucial for judging the relationship between entities, we also regard the two entities as atomic features. Moreover, we assume that the functions $\texttt{Entity}(e_1)$ and $\texttt{Entity}(e_1)$, respectively, can obtain e_1 and e_2. Then we can get the following two new atomic features.

$$\begin{aligned} x_{12} &= Entity(e_1) \\ x_{13} &= Entity(e_2) \end{aligned}$$

According to Chen et al. [4], based on the above atomic features and prior knowledge, the eight combined features are designed as follows:

$$\begin{aligned} X_1 &= RightPosOf(e_1) \oplus TypleOf(e_1) \\ X_2 &= LeftPosOf(e_1) \oplus TypleOf(e_1) \\ X_3 &= RightPosOf(e_2) \oplus TypleOf(e_2) \\ X_4 &= LeftPosOf(e_2) \oplus TypleOf(e_2) \\ X_5 &= TypleOf(e_1) \oplus TypleOf(e_2) \\ X_6 &= SubTypeOf(e_1) \oplus SubTypeOf(e_2) \\ X_7 &= HeadOf(e_1) \oplus HeadOf(e_2) \\ X_8 &= PositionBetween(e_1, e_2) \end{aligned}$$

where $\oplus$ is the connection symbol, and X_t represents the t-th combined feature. According to each combination feature, we can get the corresponding combination rule.

In this paper, we directly regard x_{12} and x_{13} as combined features. It is equivalent to using these two atomic features to construct a node of the graph without constructing an edge among it with any other node. Then the two new combined features are as follows:

$$\begin{aligned} X_9 &= Entity(e_1) \\ X_{10} &= Entity(e_2) \end{aligned}$$

In summary, we can get 13 atomic features $(x_1, ..., x_{13})$ and 10 combined features $(X_1, ..., X_{10})$ for every two entities. So far, we can see that x_{11}, x_{12} and x_{13} are both atomic and combined features. In this paper, these atomic features and combined features are our prior knowledge. Furthermore, rules or patterns to generate combined features are

also regarded as "prior knowledge". However, if there are not so many atomic features in the dataset, this number may be reduced, but the construction process of the entire model remains unchanged.

3.2. Graph Based on Combined Features

After we get the atomic features and combined features related to the two entities, the graph based on the combined features can be constructed. Specifically, we use these atomic features as nodes and construct edges between nodes according to the combination rules $(X_1, ..., X_{10})$ to obtain the graphs based on combined features. Then we use graph convolutional neural networks to extract higher-dimensional feature representations of this graph. For example, in this way, a total of 13 atomic features $(x_1, ..., x_{13})$ can be generated on the ACE05 English dataset. Then we can construct an undirected graph with 13 vertices. The sample diagram is shown in Figure 2.

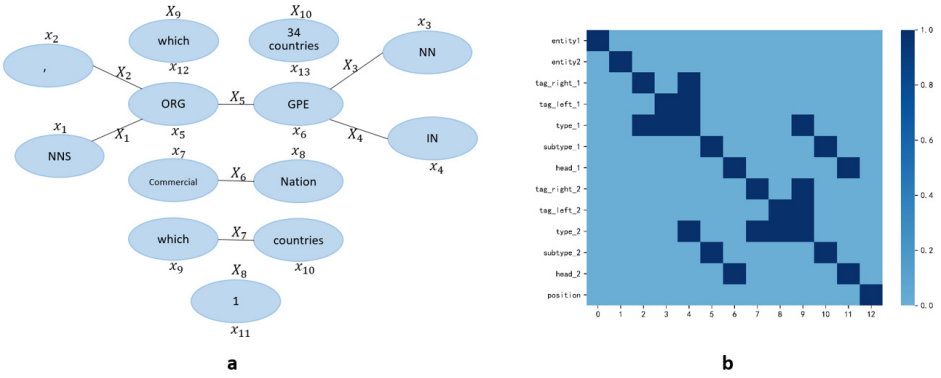

Figure 2. (**a**) The graph **a** constructed from the sentence ", which operates in 34 countries". The two entities contained in this sentence are "which" (e_1) and "34 countries" (e_2), where X_t and x_r represents the t-th combination feature and the r-th atomic feature, respectively. From the positional relationship of these two entities in the sentence, it can be seen that e_1 is before e_2, so its relative position relationship (x_{11}) is marked as 1. (**b**) The values on the horizontal axis of the adjacency matrix **b** correspond to the atomic features on the vertical axis.

Formally, we define V as the set of nodes and E as the set of edges. Whether an edge is constructed between two nodes depends on the combination rules. In other words, if two nodes belong to one of the combined features, then they are connected with undirected edges. Otherwise, the edges are not constructed. Then, we get the undirected graph $G = (V, E)$. Moreover, we assume that $\mathbf{A}$ is the adjacency matrix of the graph. If there exist an edge from node i to node j, then we define $\mathbf{A}_{ij} = \mathbf{A}_{ji} = 1$, otherwise, $\mathbf{A}_{ij} = \mathbf{A}_{ji} = 0$. We add a self-loop for each node in the graph, so the values on the diagonal of $\mathbf{A}$ are all 1.

3.3. GCN Module

Inspired by the convolutional neural network, Kipf and Welling [24] built a graph convolutional neural network on the graph, which can update the information of the current node according to the information of the neighbor nodes. After multiple layers of GCN, we can think that each node has collected information about other nodes in the entire graph. Specifically, given the undirected graph $G = (V, E)$ based on combined features with n nodes, in an L-layer GCN, each layer can be expressed as a nonlinear equation:

$$\mathbf{H}^{(l+1)} = \sigma(\widetilde{\mathbf{D}}^{-\frac{1}{2}}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}), \tag{1}$$

where $\mathbf{H}^{(l+1)}$ is the node representation of the l-th layer, $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$. $\mathbf{I}$ is the identity matrix. $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathbf{A}}_{ij}$ is the degree matrix. $\mathbf{W}$ is the parameter matrix participating in training. σ is the activation function (e.g., relu). Until now, we have obtained a graph based on combined

features and a symmetric adjacency matrix **A**. Next, we embed the vertices $(x_1, ..., x_{13})$ in the graph and denote it with **X**.

$$\mathbf{X} = \mathbf{Embedding}([x_1, ...x_{13}]) \tag{2}$$

Then, $\mathbf{H}^{(0)} = \mathbf{X}$. $\mathbf{H}^{(0)}$ is the vector representation of the initial input to GCN. The output $\mathbf{H}^{(L)}$ of the last layer is the updated feature matrix of each node.

In order to obtain a higher-dimensional feature representation, we can use multi-layer GCN:

$$\mathbf{H}^{(l+1)} = \mathbf{GCN}(\mathbf{H}^{(l)}) \tag{3}$$

In this way, each vertex can obtain information about other vertices in the entire graph. Finally, we get the feature representation $\mathbf{h}_{gcn}$ of our graph based on combined features.

$$\mathbf{h}_{gcn} = \mathbf{H}^{(L)} \tag{4}$$

3.4. Prediction

After applying the GCN model to the graph based on combined features, we get the feature representation ($\mathbf{h}_{gcn}$) of the vertex. Given the feature representation, our purpose is to determine what kind of relationship the two entities belong to in this sentence. Therefore, we apply a feedforward neural network (FFNN) and softmax layer on $\mathbf{h}_{gcn}$ to make the prediction:

$$\widetilde{\mathbf{y}} = SoftMax(FFNN(\mathbf{h}_{gcn})), \tag{5}$$

where $\widetilde{\mathbf{y}}$ refers to the predicted category probability distribution. The loss function **L** is designed to minimize the cross entropy value:

$$\mathbf{L}(\mathbf{\Theta}) = -\mathbf{y}log(\widetilde{\mathbf{y}}) - (1 - \mathbf{y})log(1 - \widetilde{\mathbf{y}}) \tag{6}$$

where **y** is the true category probability distribution, and $\mathbf{\Theta}$ is the parameter to be learned in the entire neural network.

4. Experiment

4.1. Datasets

In the experiment, we used three datasets: ACE05, CoNLL04, and SciERC. All datasets are divided into the training set, validation set, and test set, according to the ratio of 8:1:1. The data analysis of each dataset is summarized in the Table 1. Below, we give a detailed introduction of each dataset:

Table 1. Details of datasets used. Please see Section 4.1 for more details.

Split	ACE05	CoNLL04	SciERC
train	83293	15264	19540
dev	13779	1908	2443
test	13780	1908	2443

ACE05: The ACE2005 English dataset is a standard dataset used for relation extraction, which is obtained from broadcasting, news, and web logs. It contains a total of seven entity types and six relationship types, and each entity has subtypes and head tags in addition to the type. The English dataset contains 506 documents and 6583 positive data. After adding negative data, we get 110,852 data.

CoNLL04: The data in the CoNLL04 dataset [25] are obtained from news articles. Each sentence contains entities and their corresponding relationships, and each entity is

also marked with a type. There are four types of entities and five types of relationships in total. This dataset contains a total of 2048 positive data. After we add negative examples, there are a total of 19,080 data.

SciERC: The SciERC dataset [26] comes from abstracts of 500 AI papers. Like the CoNLL04 dataset, each sentence in it is marked with entities and their corresponding relationships, and each entity is also marked with types. There are six entity types and seven relationship types. There are 4648 positive cases, and 24,426 positive and negative cases.

4.2. Experimental Setting

In the experiment of the ACE05 dataset, because it contains the subtype and the head tag of the entity, there are 13 nodes in the final graph based on this dataset. In the CoNLL04 and SciERC datasets, because they do not contain these two features, the graph based on these two datasets has only nine nodes. Whether it is a thirteen-node graph or a nine-node graph, they all construct edges according to the combination rules described above to construct the graph.

In the experiment, we adjust the hyperparameters based on the validation set. Finally, the maximum sentence length is set to 100, the number of convolution kernels in CNN is 60, and the size of the convolution kernel is set to 7, the dropout ratio is set to 0.5, the L2 regularization lambda is set to 1e-5, the initial learning rate is 1, and the decay rate reduced by 0.9 can get the best results. We also use the standard precision rate (P), recall rate (R) and F1 value as the evaluation indicators of the model. For each dataset, we have published its macro average F1 value.

4.3. Results on the Three Datasets

In this section, we show the multi-classification results of our model on ACE05, CoNLL04, and SciERC datasets in Tables 2 and 3. In order to obtain information about the sentence itself containing two entities, we also add sentence features to our model. The sentence features are obtained through a convolutional neural network. From the Tables 2 and 3, on the one hand, there are some improvements in the three data sets after adding sentence features. On the other hand, it can be seen from these two tables that the F1 value of the category "PHYS", "Feature-of" are the lowest among all categories on the ACE05 and SciERC datasets, respectively. Additionally, the F1 value of the category "OrgBased_In" is the highest among all categories in the CoNLL04 dataset. Thus, for these features we used, they can well-capture the information of the entity pairs belonging to the category "OrgBased_In", so as to well-separate the entity pairs belonging to this category on the CoNLL04 dataset. However, for the entity pair belonging to the category "PHYS" and "Feature-of", it cannot extract the information well, resulting in the F1 value in this category being much lower than other categories on the ACE05 and SciERC datasets, respectively. Therefore, in future research, we need to pay more attention to the entity pairs in these categories, observe the characteristics of them, and see if there are other methods to extract the information of the entity pairs in this category.

In addition, by observing the results on the three datasets, it can be seen that the results on the ACE05 and CoNLL04 datasets are higher than the results on the SciERC dataset, and the results of the ACE05 and CoNLL04 datasets are not much different. This may be caused by the different sources of the three datasets. Through Section 4.1, we can find that the data in the ACE05 and CoNLL04 datasets are news-related, while the data in the SciERC dataset are abstracts from AI papers.

Table 2. Multi-classification results of ACE05 dataset.

Dataset	Relationship Type	FC-GCN			FC-GCN+sen		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
ACE05	PHYS	68.22	44.79	54.07	71.70	46.63	56.51
	ART	87.18	53.12	66.02	93.55	45.31	61.05
	GEN-AFF	69.77	44.78	54.55	61.29	56.72	58.91
	ORG-AFF	85.63	77.30	81.25	91.22	72.97	81.08
	PART-WHOLE	73.63	77.01	75.28	75.58	74.71	75.14
	PER-SOC	88.89	70.00	78.32	91.67	68.75	78.57
	total	78.89	61.17	**68.91**	80.83	60.85	**69.43**

Table 3. Multi-classification results of on the CoNLL04 and SciERC datasets.

Dataset	Relationship Type	FC-GCN			FC-GCN+sen		
		P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
CoNLL04	Work_For	62.79	87.10	72.97	65.00	83.87	73.24
	Kill	70.59	80.00	75.00	71.88	76.67	74.19
	OrgBased_In	75.51	77.08	76.29	82.61	79.17	80.85
	Live_In	63.93	69.64	66.67	59.09	69.64	63.93
	Located_In	65.85	62.79	64.29	55.56	81.40	66.04
	total	67.74	75.32	**71.33**	66.83	78.15	**72.05**
SciERC	Used-for	35.05	27.31	30.70	27.67	35.34	31.04
	Feature-of	00.00	00.00	00.00	08.00	10.53	09.09
	Hyponym-of	76.19	33.33	46.38	66.67	37.50	48.00
	Evaluate-for	55.17	41.03	47.06	44.68	53.85	48.84
	Part-of	16.67	04.55	07.14	19.44	31.82	24.14
	Compare	44.44	20.00	27.59	20.00	20.00	20.00
	Conjunction	52.83	47.46	50.00	32.35	37.29	34.65
	total	40.05	24.81	**30.64**	31.26	32.33	**31.79**

4.4. Comparison with Benchmark Models

We make a comparison with two benchmark models: An atomic feature model based on a neural network (M_{ato}) and a combined feature model based on a neural network (M_{com}).

M_{ato} directly uses the atomic features mentioned above when a neural network performs relation extraction. It differs from the FC-GCN model by only one GCN module. First, we get the atomic features like the FC-GCN model. Then, we use the classifier to predict the relationship category. Since the difference between the M_{ato} and FC-GCN models is whether to use the GCN module, by comparing with this model, the importance of the GCN module to the FC-GCN model can be reflected.

M_{com} uses the neural network to perform relation extraction after combining atomic features according to the combination rules. The difference between the M_{com} model and the M_{ato} model is whether to combine the atomic features according to the above-mentioned combination rules to obtain the combined features. First, like the M_{ato} model, we get the atomic features. Then we use combination rules to transform atomic features into combination features. Finally, we use the classifier to predict the relationship category. Compared with the FC-GCN model, it lacks the GCN module, but has an additional feature combination step. Because the GCN module in the FC-GCN model has a certain function of combining features, comparing the M_{com} model with the FC-GCN model can reflect that the GCN module has stronger capabilities, not just the ability to combine features.

Table 4 shows the comparison results of our model, the M_{atomic} model, and $M_{combined}$ model. It can be seen from the Table 4 that our model exceeds the M_{atomic} model and the $M_{combined}$ model in the three datasets. It indicates that the GCN module cannot only

extract features in the graph effectively, but also extract more hidden information than simply combine features. More importantly, our model not only has good superiority in the F1 value, but also reach new heights in the accuracy and recall rate. In order to obtain information about the sentence itself containing two entities, we also add sentence features to our model. The sentence features are obtained by implementing a convolutional neural network on a sentence containing two entities. Then, the output is concatenated with the final feature representation and $\mathbf{h_{gcn}}$ for classification. The result is shown in Table 4. After adding sentence features to our model, the results are not only improved, but the distribution of results on all models is the same as without sentence features.

Table 4. Comparison results with the $\mathbf{M}_{ato}$ model and $\mathbf{M}_{com}$ model on three datasets. The meaning of "+sen" is to add a sentence feature containing two entities.

Model	ACE05			CoNLL04			SciERC		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
$\mathbf{M}_{ato}$	77.01	50.48	60.98	71.30	65.02	68.02	52.18	19.26	28.14
$\mathbf{M}_{com}$	72.11	54.45	62.05	72.28	65.63	68.79	61.15	19.71	29.81
FC-GCN	78.89	61.17	68.91	67.74	75.32	71.33	40.05	24.81	30.64
$\mathbf{M}_{ato}$+sen	64.65	56.62	60.37	69.04	67.12	68.07	33.41	22.13	26.62
$\mathbf{M}_{comb}$+sen	74.72	51.36	60.88	65.66	72.95	69.11	30.20	24.33	26.95
FC-GCN+sen	80.83	60.85	**69.43**	66.83	78.15	**72.05**	31.26	32.33	**31.79**

4.5. Analysis

In summary, no matter how the adjacency matrix changes, our model is always better than the M_{com} model and the M_{com} model. It shows that the method of constructing the feature into a graph and then using the graph convolutional neural network to extract the features of the graph is effective.

4.6. Comparison with Related Works

In this part of the experiment, we compared the FC-GCN model with other strategies. Additionally, we show the results of the model (BERT_FC-GCN+sen) after BERT [27] pre-training. In the "BERT_FC-GCN+sen" model, the node's vector representation $\mathbf{X}$ and the sentence-embedding containing two entities are both the outputs passing through the BERT layer.

Table 5 shows the comparison results between our model and other models on the ACE05 dataset. DRPC indirectly uses the dependency tree to predict the dependency relationship between words and the relationship between entities, so as to facilitate the capture of text information outside of syntactic information and the generalization of cross-domains. GCN(D+H) regards entities and entity pairs as nodes and constructs a bipartite graph. BERT-Z&H adds tags on both sides of the entity for relation extraction. Through this table, we can find that our model is better than these models. More specifically, our model is at least 1.91% higher than the above model in F1 value. It shows that the use of the graphs based on combined features can better extract the information between atomic features, so as to better extract the information between entity pairs, and finally make better decisions.

For the CoNLL04 and SciERC datasets, we show the results of comparison with other methods in Table 6. It can be seen from this table that our model is also higher than other methods. Additionally, the F1 value is at least 4.83% and 1.06% higher on the CoNLL04 and SciERC datasets, respectively.

Table 5. Comparison results with other models on the ACE05 dataset.

Model	P(%)	R(%)	F1(%)
K [28]	63.50	45.20	52.80
G&J [29]	77.20	60.70	68.00
FCM [30]	71.52	49.32	58.26
DRPC [31]	72.10	63.49	67.52
GCN(D+H) [22]	68.7	65.4	67.00
BERT-EA [32]	-	-	67.46
BERT-Z&H [33]	-	-	73.10
FC-GCN	78.89	61.17	68.91
FC-GCN+sen	80.83	60.85	**69.43**
BERT_FC-GCN+sen	85.64	75.95	**80.50**

Table 6. Comparison results with other models on the CoNLL04 and SciERC datasets.

Model	CoNLL04			SciERC		
	P(%)	R(%)	F1(%)	P(%)	R(%)	F1(%)
Multi-channel [34]	74.53	58.38	64.94	22.78	18.37	20.29
$\mathbf{F}_{atomic}$ [35]	46.89	73.43	57.23	20.31	18.30	19.25
$\mathbf{F}_{combined}$ [35]	72.87	62.39	67.22	36.16	26.72	30.73
FC-GCN+sen	66.83	78.15	72.05	31.26	32.33	31.79
BERT_FC-GCN+sen	81.75	77.39	**79.51**	62.85	55.71	**59.07**

Because relation extraction is implemented at sentence level, where a sentence usually contains a limited number of words, it suffers from a serious feature sparsity problem. The BERT embedding is pre-trained on external resources with the unsupervised method. It is effective to learn word semantic representations. They are helpful to reduce the feature sparsi-ty problem. Therefore, on these three data sets, the best results have been achieved after adding the BERT embedding.

In this section, we analyze the effect of different layers of GCN on the FC-GCN model. Specifically, we test the GCN layer numbers of 1, 2, and 3 on the three datasets. Additionally, the experimental results are shown in Figure 3. Firstly, it can be seen from the figure that no matter which dataset is used, the result of one layer of GCN is always higher than that of multi-layer GCN. Specifically, it can be clearly seen from the figure that the gap between one layer of GCN and two layers of GCN is the largest on the SciERC dataset, with a difference of 3.82% in F1 value. However, on the ACE05 and CoNLL04 datasets, the largest difference in F1 values is the two layers of GCN and the three layers of GCN, with a difference of 2.81%, 14.74% in F1 value, respectively. Moreover, compared with the results of the ACE05 and CoNLL04 datasets, the difference between the results on the SciERC dataset is not very large. This may be because the sources of these three datasets are different. Additionally, the information contained in them is also different. Thus, the hidden features needed to analyze the information are also different.

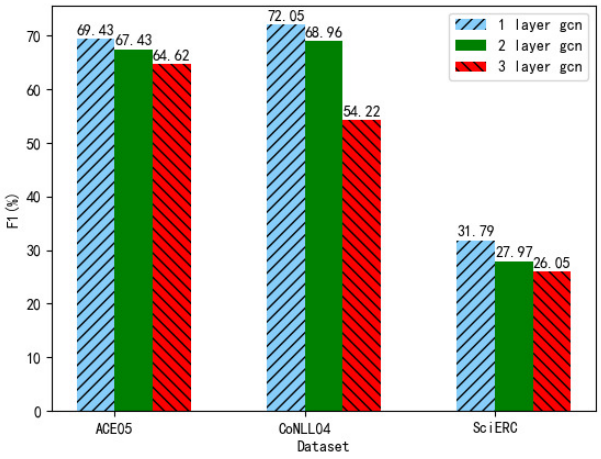

Figure 3. The results of different gcn layers on each dataset.

5. Conclusions and Future Work

In this paper, we proposed a novel feature combination-based graph convolutional neural network model (FC-GCN) for relation extraction, which no longer uses dependency trees to build graphs and reduces problems caused by external tools. Instead, we created an undirected graph based on combined features, which used atomic features as nodes and constructed edges between nodes according to the combination rules. Then, we applied a graph convolutional neural network to this graph and extracted high-dimensional information of this graph. Our model not only makes good use of sentence structural information and prior knowledge, but also avoids the problems caused by the dependency trees. Additionally, we demonstrated its superiority on three datasets. It outperforms existing state-of-the-art performances. Since we constructed a static graph in this paper, the adjacency matrix is fixed, and we will try to construct a dynamic graph in future research and test whether it will improve the experimental results. In addition, we will explore more ways to construct graphs in future research to better capture the structural information of sentences.

Author Contributions: J.X.: data curation, investigation, methodology, resources and writing (original draft); Y.C.: conceptualization, formal analysis, methodology, supervision and writing (review) and editing; Y.Q.: formal analysis, funding acquisition, project administration and supervision; R.H.: funding acquisition and supervision; Q.Z.: supervision and writing (review) and editing. All authors have read and agreed to the final version of the manuscript

Funding: This work is supported by the Joint Funds of the National Natural Science Foundation of China under Grant No. U1836205, the Major Research Program of National Natural Science Foundation of China under Grant No. 91746116, National Natural Science Foundation of China under Grant No. 62066007 and No. 62066008, the Major Special Science and Technology Projects of Guizhou Province under Grant No. [2017]3002, the Key Projects of Science and Technology of Guizhou Province under Grant No. [2020] 1Z055 and Project of Guizhou Province Graduate Research Fund (Qianjiaohe YJSCXJH[2019]102).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

References

1. Zhang, Y.; Zhong, V.; Chen, D.; Angeli, G.; Manning, C.D. Position-aware attention and supervised data improve slot filling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017.
2. He, R.; Wang, J.; Guo, F.; Han, Y. Transs-driven joint learning architecture for implicit discourse relation recognition. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020.

3. Sun, K.; Zhang, R.; Mao, Y.; Mensah, S.; Liu, X. Relation extraction with convolutional network over learnable syntax-transport graph. *Proc. AAAI Conf. Artif.* **2020**, *34*, 8928–8935. [CrossRef]
4. Chen, Y.; Wang, G.; Zheng, Q.; Qin, Y.; Chen, P. A set space model to capture structural information of a sentence. *IEEE Access* **2019**, *7*, 142515–142530. [CrossRef]
5. Zeng, D.; Liu, K.; Lai, S.; Zhou, G.; Zhao, J. Relation classification via convolutional deep neural network. In *Proceedings of the COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*; Dublin City University and Association for Computational Linguistics: Dublin, Ireland, 2014; pp. 2335–2344. Available online: https://www.aclweb.org/anthology/C14-1220 (accessed on 1 April 2021).
6. Xu, K.; Feng, Y.; Huang, S.; Zhao, D. Semantic relation classification via convolutional neural networks with simple negative sampling. *Comput. Sci.* **2015**, *71*, 941–949.
7. Zeng, D.; Liu, K.; Chen, Y.; Zhao, J. Distant supervision for relation extraction via piecewise convolutional neural networks. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015.
8. Yan, X.; Mou, L.; Li, G.; Chen, Y.; Jin, Z. Classifying relations via long short term memory networks along shortest dependency paths. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), Lisbon, Portugal, 17–21 September 2015.
9. Zhang, Z.; Shu, X.; Yu, B.; Liu, T.; Guo, L. Distilling knowledge from well-informed soft labels for neural relation extraction. *Proc. AAAI Conf. Artif.* **2020**, *34*, 9620–9627.
10. Veyseh, A.P.B.; Dernoncourt, F.; Dou, D.; Nguyen, T.H. Exploiting the syntax-model consistency for neural relation extraction. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020.
11. Alt, C.; Gabryszak, A.; Hennig, L. Probing linguistic features of sentence-level representations in neural relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 1534–1545. Available online: https://www.aclweb.org/anthology/2020.acl-main.140 (accessed on 1 April 2021).
12. Yu, D.; Sun, K.; Cardie, C.; Yu, D. Dialogue-based relation extraction. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020.
13. Zhou, W.; Huang, K.; Ma, T.; Huang, J. Document-level relation extraction with adaptive thresholding and localized context pooling. *arXiv* **2020**, arXiv:2010.11304.
14. Jain, S.; van Zuylen, M.; Hajishirzi, H.; Beltagy, I. Scirex: A challenge dataset for document-level information extraction. *arXiv* **2020**, arXiv:2005.00512.
15. Wei, Z.; Su, J.; Wang, Y.; Tian, Y.; Chang, Y. A novel cascade binary tagging framework for relational triple extraction. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020.
16. Shen, Y.; Huang, X.-J. Attention-based convolutional neural network for semantic relation extraction. In Proceedings of the COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 2526–2536.
17. Guo, Z.; Zhang, Y.; Lu, W. Attention guided graph convolutional networks for relation extraction. *arXiv* **2019**, arXiv:1906.07510.
18. Vashishth, S.; Joshi, R.; Prayaga, S.S.; Bhattacharyya, C.; Talukdar, P. Reside: Improving distantly-supervised neural relation extraction using side information. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018.
19. Fu, T.J.; Ma, W.Y. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. *ACL* **2019**, 1409–1418. [CrossRef]
20. Zhang, Y.; Qi, P.; Manning, C.D. Graph convolution over pruned dependency trees improves relation extraction. *arXiv* **2018**, arXiv:1809.10185.
21. Vashishth, S.; Sanyal, S.; Nitin, V.; Talukdar, P.P. Composition-based multi-relational graph convolutional networks. *arXiv* **2019**, arXiv:1911.03082.
22. Sun, C.; Gong, Y.; Wu, Y.; Gong, M.; Duan, N. Joint type inference on entities and relations via graph convolutional networks. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019.
23. Chen, Y.; Zheng, Q.; Zhang, W. Omni-word feature and soft constraint for chinese relation extraction. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 22–27 June 2014.
24. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
25. Roth, D.; Yih, W.-T. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004, Boston, MA, USA, 6–7 May 2004*; Association for Computational Linguistics: Boston, MA, USA, 2004; pp. 1–8. Available online: https://www.aclweb.org/anthology/W04-2401 (accessed on 1 April 2021).
26. Luan, Y.; He, L.; Ostendorf, M.; Hajishirzi, H. Multi-task identification of entities, relations, and coreferencefor scientific knowledge graph construction. *arXiv* **2018**, arXiv:1808.09602.
27. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.

28. Kambhatla, N. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive Poster and Demonstration Sessions, ACLdemo '04*, Barcelona, Spain, 21–26 July 2004; Association for Computational Linguistics: Boston, MA, USA, 2004. [CrossRef]
29. Zhou, G.; Su, J.; Zhang, J.; Zhang, M. Exploring various knowledge in relation extraction. In Proceedings of the ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, MI, USA, 25–30 June 2005.
30. Gormley, M.R.; Yu, M.; Dredze, M. Improved relation extraction with feature-rich compositional embedding models. *arXiv* **2015**, arXiv:1505.02419.
31. Veyseh, A.P.B.; Nguyen, T.H.; Dou, D. Improving cross-domain performance for relation extraction via dependency prediction and information flow control. *arXiv* **2019**, arXiv:1907.03230.
32. Wang, H.; Tan, M.; Yu, M.; Chang, S.; Wang, D.; Xu, K.; Guo, X.; Potdar, S. Extracting multiple-relations in one-pass with pre-trained transformers. *arXiv* **2019**, arXiv:1902.01030.
33. Zhong, Z.; Chen, D. A frustratingly easy approach for joint entity and relation extraction. *arXiv* **2020**, arXiv:2010.12812.
34. Chen, Y.; Wang, K.; Yang, W.; Qing, Y.; Huang, R.; Chen, P. A multi-channel deep neural network for relation extraction. *IEEE Access* **2020**, *8*, 13195–13203. [CrossRef]
35. Chen, Y.; Yang, W.; Wang, K.; Qin, Y.; Huang, R.; Zheng, Q. A neuralized feature engineering method for entity relation extraction. *Neural Netw.* **2021**, *141*, 249–260. [PubMed]