


Article

Syntax-Informed Self-Attention Network for Span-Based Joint Entity and Relation Extraction

Haiyang Zhang ^{*}, Guanqun Zhang ^{*} and Yue Ma

School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China; mayue@bupt.edu.cn

^{*} Correspondence: zhhy@bupt.edu.cn (H.Z.); zhanggq9616@bupt.edu.cn (G.Z.)

Abstract: Current state-of-the-art joint entity and relation extraction framework is based on span-level entity classification and relation identification between pairs of entity mentions. However, while maintaining an efficient exhaustive search on spans, the importance of syntactic features is not taken into consideration. It will lead to a problem that the prediction of a relation between two entities is related based on corresponding entity types, but in fact they are not related in the sentence. In addition, although previous works have proven that extract local context is beneficial for the task, it still lacks in-depth learning of contextual features in local context. In this paper, we propose to incorporate syntax knowledge into multi-head self-attention by employing part of heads to focus on syntactic parents of each token from pruned dependency trees, and we use it to model the global context to fuse syntactic and semantic features. In addition, in order to get richer contextual features from the local context, we apply local focus mechanism on entity pairs and corresponding context. Based on applying the two strategies, we perform joint entity and relation extraction on span-level. Experimental results show that our model achieves significant improvements on both Conll04 and SciERC dataset compared to strong competitors.



Citation: Zhang, H.; Zhang, G.; Ma, Y. Syntax-Informed Self-Attention Network for Span-Based Joint Entity and Relation Extraction. *Appl. Sci.* **2021**, *11*, 1480. <https://dx.doi.org/10.3390/app11041480>

Academic Editor: Ricardo Colomo-Palacios
Received: 1 January 2021
Accepted: 3 February 2021
Published: 6 February 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: entity recognition; relation extraction; dependency tree; self-attention

1. Introduction

Relation Extraction (RE) is the task of aiming to find semantic relationships between pairs of the entity mentions from unstructured texts based on entity recognition. As an essential task in information extraction, which is widely applied in question answering [1], populate knowledge base [2], and other knowledge-based tasks. The relational facts obtained from RE are often expressed in form of knowledge triplet $\{h, r, t\}$, where h and t represent head and tail entity, respectively, and r represents the relation between the entities. One annotated example is illustrated in Figure 1.

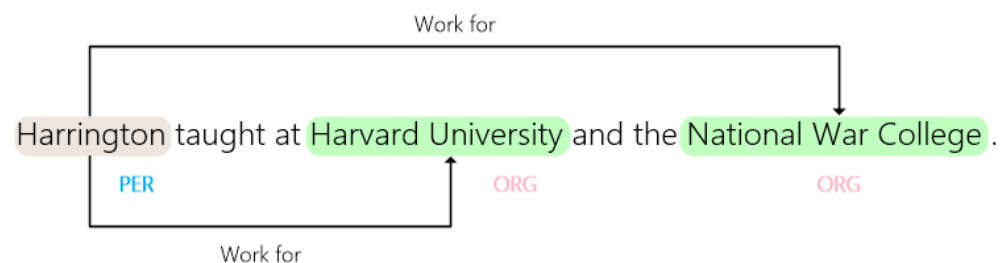


Figure 1. Annotated result in a sample sentence (ORG: organization; PER: person).

Previous works can be divided into two categories: one is in pipelined setting, the other is in joint setting. The pipelined method [3–5] divides the task into two steps: named entity recognition (NER) and relation extraction (RE). Although this kind of method seems reasonable, it ignores relevance between the two tasks. For example, by identifying entity

mentions, we can classify “Michael Jackson” and “Chicago” as *person* and *locate* type, respectively. It is easy to find that the type of above two entities is helpful for identifying relation “Live-in”. To tackle with this problem, joint method is proposed. Different from pipelined models, joint models can extract both entities and relations with shared parameters in an end-to-end architecture. It can effectively integrate the feature of entities and relations and achieve better results in the task.

For the problem of extracting relations, the relational facts in sentences are often complicated: a sentence may contain multiple entities, an entity may participate in multiple relational triplets, and triplets may have overlapped entity pairs. In addition to this, the nesting issue may occur in these entity mentions. Refs. [6–8] only can cover the situation where none of triplets have overlapped entities in a sentence. After that, many works like [9–11] can cover the situation where triplets have overlapped entity pairs and an entity participates in different triplets. Among these models, each token corresponds to a single fixed representation and we usually refer to them as token-based models. However, some tokens may participate in different spans, such as “the minister of the department of education” and “the department of education”. It’s hard for token-level models to model overlapping entities due to their natural limitation that one token has a single fixed representation. To tackle this issue, span-based models are proposed. Refs. [12,13] solve the problem of entity nesting by searching all possible spans exhaustively to complete RE task. At present, span-based joint entity and relation extraction models can cover the most complex situations.

However, despite span-based model can cover more cases, it still has many shortcomings that affect the quality of extraction. First, while maintaining an efficient exhaustive search on spans, the importance of syntactic features is not taken into consideration. It will lead to a problem that the prediction of a relation between two entities is related based on corresponding entity types, but in fact they are not related in the sentence. Employing additional syntactic features is helpful for this issue. Second, although previous state-of-the-art work like [13] has proven that extracting local context is superior to global context for RE task, but simply conducting a pooling method over the context directly to align feature dimensions can result in the loss of some critical information. For the processing of local context, in-depth learning of contextual features is needed.

Recently, Transformer [14] is a popular model applied on a wide range of Natural Language Processing (NLP) tasks. Multi-head self-attention is the core component of Transformer, by dividing semantic features into multiple subspaces and performing multiple attention functions to compute attention scores for each contextual word, it is possible to learn richer contextual features in the sentence. However, recent works [15–18] show that the power of multiple heads is not fully exploited. Therefore, how to make full use of these heads is a valuable question. In addition, BERT [19] is a new language representation model based on a multi-layer bidirectional Transformer, which achieved impressive improvements on many NLP tasks, such as sentiment classification [20], question answering [21], reading comprehension [22], and so on.

Based on the above observations, we propose a joint model based on pre-trained BERT, which combines syntax-informed and local context attention mechanism. In NER task, we map span-based entity mentions to specified types and then get candidate entity pairs. In RE task, based on candidate entity pairs, the above two attention mechanisms are applied. We use pruned dependency trees as our syntactic features, and then analyze the effect of different pruning methods on results.

To recap, the main contributions of this work are as follows:

- We propose to incorporate syntax knowledge into multi-head self-attention by employing part of the heads to focus on syntactic parents of each token from pruned dependency trees. In addition, we use it to model the global context to fuse syntactic and semantic features. Then, we compare the effects of several different pruning methods on the results;

- Based on the first point, according to the different positions of each pair of candidate entities from the NER task, we mask part of content of the sentence dynamically, just keeping the entity pair and content between them, and perform a local focus mechanism on the context to learn richer contextual features in the RE task;
- We employ BERT as our pre-trained language model and fine-tune it during training, and experimental results show that our model achieves significant improvements on both Conll04 and SciERC dataset compared to strong competitors.

2. Related Works

2.1. Syntax-Based Relation Extraction Model

Syntactic features are beneficial for RE tasks because they can capture dependencies between different components in a sentence that may be ambiguous on the surface. In the past, at the core of relation extraction approaches are statistical classifiers, many of which find that syntactic features play an important role. For traditional methods, Ref. [23] proposed a tree-based kernel approach and introduced kernels defined over shallow parse representations of text. Ref. [24] introduced the simplified dependency path to tree kernels. They proved that such feature can improve the performance of extraction. For neural-based models, Ref. [25] first proposed to learn relation representations from Shortest Dependency Path (SDP) through Convolutional Neural Network (CNN), and Refs. [26,27] tried to employ LSTM to encode SDP, which proved the effectiveness for syntactic feature encoding with neural models. In addition, they removed irrelevant information by trimming the trees and achieved better results. In addition to these preliminary attempts, some works modeled syntax trees by improving feature extractors. Refs. [6,28] used Tree-LSTM [29] and Tree-GRU models, respectively, which were an extended form of RNNs over dependency trees and had stronger modeling capabilities. After that, since the dependency tree structure is inherently graph-like, with the emergence of Graph Convolution Network (GCN) [30], Refs. [11,31] employed GCN to generate syntactic representations. Experimental results proved that sequence models had complementary strengths to syntax-based models and combining them can achieve impressive performance on their benchmarks. Ref. [32] used dependency trees to extract the syntax-based importance scores for the words with Ordered-Neuron Long-Short Term Memory Networks [33] (ON-LSTM) and injected these features into the model. Their experimental results showed that capturing the syntactic importance of the words was beneficial for RE task and had greater generalization ability.

In addition, with the introduction of Transformer [14], although the multi-head self-attention mechanism has been proven to have excellent performance in some NLP tasks, it lacks a hierarchical structure of the input sentence. Refs. [34,35] employed absolute structural position and relative structural position to encode structural features, and integrated them into self-attention, respectively, which had better performance on machine translation tasks. In summary, it is feasible to incorporate syntactic feature into self-attention and provide more possibilities for the expansion of syntax-based model in other NLP tasks.

2.2. Joint Entity and Relation Extraction Model

Early joint models were mainly feature-based [36–38]. They often rely on the quality of feature construction heavily, which is time-consuming and lacks sufficient generalizability. With success of deep learning on many NLP tasks, neural-based models make up for the shortcomings of feature-based models, and have many applications in the field of RE tasks. Ref. [6] proposed to employ RNN-LSTM with Tree-LSTM for joint entity and relation extraction, and modeled syntactic and sequence information simultaneously. Ref. [39] proposed a hybrid neural network that employed BiLSTM for entity recognition and CNN for relationship extraction without any handcrafted features. Refs. [40,41] proposed to model the task as a table filling problem, but it would cause redundant information and increase the possibility of errors. Ref. [8] first proposed to model the task as a sequence labeling problem on BIOES scheme. They synthesized entity types and relations as labels, and artificially features were not adopted. To some extent, although the problem of error

propagation is avoided, it has shortcomings on identification of the overlapping relations. After that, Ref. [9] proposed to divide situations into three categories: Normal (none of its triplets have overlapped entities), Single Entity Overlap (SEO), and Entity Pair Overlap (EPO). They viewed the task as a triple generation process based on seq2seq with a copy mechanism to cover all above situations. Refs. [6–8] only solved the problem of the Normal situation. After that, some works began to cover SEO and EPO situations. Ref. [10] employed BiLSTM to encode sequence information and modeled the RE task as a multi-head selection problem. In addition, Conditional Random Field (CRF) was used to decode so that the model can identify entities and all possible relations between them to solve overlapping relations issue. Based on the architecture of [10], Refs. [42,43] incorporated different types of attention, but still based on a sequence labeling scheme. Ref. [11] adopted GCN to encode dependency trees and BiLSTM as their sequence model, and integrated them into an end-to-end architecture. Ref. [44] formulated the joint extraction as a token pair linking problem and introduced a novel handshaking tagging scheme to align the boundary tokens of entity pairs under each relation type. In addition, the results showed that their method had good performance on relation extraction task. Due to the entity nesting issue, span-based method becomes popular. Refs. [12,13] completed the NER task by exhaustively searching spans and made these spans into candidate entity pairs, and let the model cover situations not considered in [9]. Ref. [45] dynamically constructed span graphs, by selecting the most confident entity spans and linking these nodes with confidence-weight. Through propagation of the graph, it can refine the span representation iteratively. On the basis of [45], Ref. [46] replaced the RNN encoder with BERT, and get better extraction results. Ref. [47] was also based on the Transformer and modeled the task as a multi-turn QA problem in an innovative manner, but it required constructing question templates manually, which was time-consuming. Overall, span-based method has achieved the best performance on multiple benchmarks of joint models and using the Transformer for feature extraction usually has better performance.

3. Methodology

In this section, we present our joint model illustrated in Figure 2. We employ BERT as our language representation model and fine-tune it during training. First, the input sequence will be mapped into BERT embeddings, and then we adopt pooling strategy to group the subword units of each word and obtain its representation on the word level (see Section 3.2). Second, due to the span-level, the NER task can better identify nested entity mentions. We use feature embeddings from the first step to construct span-level representation, then complete the NER task on the span level and get candidate entity pairs (see Section 3.3). Third, in order to let the model attend to syntactic features, we adopt syntax-informed multi-head self-attention to model the global context (see Section 3.4). Fourth, we leverage these candidate entity pairs from the second step to extract context between them, and local context representation is constructed based on the features from the previous step. Then, we apply the local focus mechanism on it. The attention mechanism of local context can help us to learn in-depth contextual features (see Section 3.5). Finally, we get final representations of entity pairs and corresponding context to complete the RE task (see Section 3.6).

3.1. Task Definition

Our model is a unified framework to classify named entities and relations in entity pairs. The input of our model represents a sequence of tokens $W = \{w_0, w_1, \dots, w_n\}$. For the input sequence, we search for all possible spans that satisfy the specified requirements and then get the candidate span set S . For each span $s \in S$, we compute a vector of entity type scores and choose the category with the highest score as its type. For the span whose prediction is *none*, we will filter it out. Then, from candidate entity pair set $S \times S$ and pre-defined relation set R , we extract corresponding semantic relationship based on context.

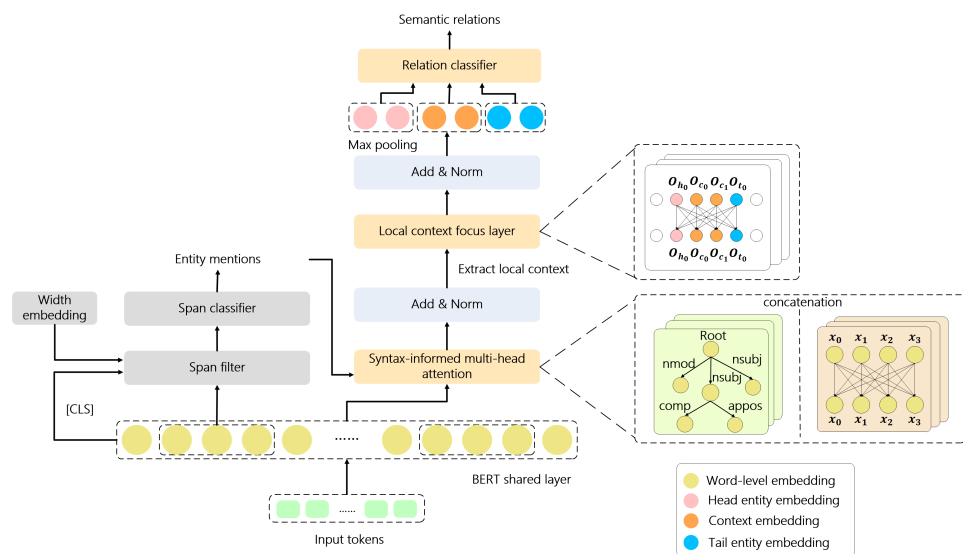


Figure 2. The overall architecture of our joint model.

3.2. Embedding Layer

Given a sentence $W_{raw} = \{w_0, w_1, \dots, w_n\}$ as a sequence of tokens, the embedding layer will map these tokens into feature vectors. In this layer, we employ BERT as our pre-trained language model. Before mapping these tokens, we transform the input as a sequence $W = \{[CLS], w_0, w_1, \dots, w_n, [SEP]\}$, where [CLS] is a special token added in front of every input sample, and [SEP] is a special separator token.

Since the BERT tokenizer is created with Wordpiece [48], which decomposes infrequent words into frequent subwords for unsupervised tokenization of the input token, but since syntactic features can only be represented on word-level, we need to align the subword units with their corresponding origin word. Therefore, if a word has multiple subword units, we will adopt the average pooling to group these units and obtain the word-level representation. After that, we can get feature sequence $X = \{x_0, x_1, \dots, x_l\}$, where l is the length of the input sequence.

3.3. Span-Based Named Entity Recognition

First, we find all candidate spans from the input sentence. Since too many spans will take up too much memory, we set a threshold t to limit the length of candidate spans, and spans whose length exceed t will be discarded. We denote a span as $s = \{s_0, s_1, \dots, s_n\}$, where $n \leq t$. For span s , we apply max pooling over it to create its representation. Since width is a distinguishing feature for span, we will integrate it into representation of span and denote w as the width feature. In addition, after BERT encodes the input sentence, the vector corresponding to token [CLS] will be generated, and it represents contextual features of the whole sentence. For entity classification, contextual information usually plays an important role, so we will incorporate it into span representation and denote the [CLS] vector as c . Finally, we concatenate all above features to get the final representation s' of a candidate span:

$$s' = MaxPool(s_0, s_1, \dots, s_n) \oplus w_n \oplus c \tag{1}$$

where \oplus represents the feature concatenation operator.

After getting the final representation s' , we classify candidate spans into specific types. Let E be a set of pre-defined entity types. We then employ a fully connected layer for classification of entities. For each entity type $e \in E$, we have

$$P^e = softmax(W^e \times s' + b^e) \tag{2}$$

From (2), we can get the score of each type of the span. In addition, each span will be assigned the entity type corresponding to the highest prediction score. Then, all spans are classified as *none* will be filtered out. Others will be combined into candidate entity pairs for the future RE task.

3.4. Syntax-Informed Multi-Head Self-Attention

Multi-head self-attention is the core component of Transformer [14], which not only allows the model to jointly attend to information from different feature subspaces at different positions, but also can be parallelized. Each of the heads learns a distinct attention function to attend to all of tokens in the sequence, and the feature of these heads will be concatenated to create the final representation.

Many experimental results have shown that it has a powerful ability to learn context features and achieves impressive performance in neural machine translation benchmark [14,49]. However, there are many studies find that the multi-head self-attention's capability has not been fully exploited [15–18], and we expect to incorporate syntactic features into multi-head self-attention architecture to enhance its modeling capabilities. Therefore, we propose to incorporate syntax knowledge into multi-head self-attention by employing part of the heads to focus on syntactic parents of each token from pruned dependency trees, and use the rest of the heads to focus on semantic information in order to exploit the ability of the heads even further. In this layer, we will apply this mechanism on the global context.

For semantic feature, we let h_1 attention heads to attend to it. For each head, we map the input matrix X of N tokens representations into query Q_h , key K_h , and value V_h with h different learnable matrices:

$$\begin{cases} Q_h = \text{Linear}_h^Q(X) = X * W_h^Q \\ K_h = \text{Linear}_h^K(X) = X * W_h^K \\ V_h = \text{Linear}_h^V(X) = X * W_h^V \end{cases} \quad (3)$$

After projecting, our Q_h , K_h , and V_h are of dimensions $N \times d_Q$, $N \times d_K$ and $N \times d_V$, $0 \leq h \leq h_1$, and d_Q, d_K, d_V are dimensions of one head. Then, we use scaled dot-product to calculate attention weights, which is formulated as follows:

$$A_h^{Sem} = \text{softmax}\left(\frac{Q_h K_h^T}{\sqrt{d_h}}\right) \quad (4)$$

where $\sqrt{d_h}$ is a scaled weight, and the value is a square root of its embedding dimension.

Then, semantic attention weights will be multiplied by V_h for each token, and we will get the semantic-attended token representations:

$$M_h^{Sem} = A_h^{Sem} \times V_h \quad (5)$$

Then, we concatenate the results of these heads, and get semantic representation M^{Sem} :

$$M^{Sem} = [M_0^{Sem}, \dots, M_{h_1}^{Sem}] \quad (6)$$

For syntactic features, we let h_2 attention heads to attend to it and map the input matrix into query, key, and value as (4). We use external tools to generate a dependency tree of the input sentence, and prune the tree with different methods according to the candidate entity pairs generated from NER task. Similar to [6], we test three methods for pruning:

- **ALL:** Keep the entire syntax tree without any pruning operations;
- **SDP (Shortest Dependency Path):** Keep the shortest dependency path between entities in the syntax tree;
- **LCA (Lowest Common Ancestors):** Keep the subtree under the lowest common ancestor of the entities pair.

Then, we convert the prune tree into an adjacency matrix, and denote it as A_{prune} . The attention weights are calculated as

$$A_h^{Syn} = A_{prune} \tag{7}$$

where A_{prune} is a $N \times N$ one-hot matrix, and $0 \leq h \leq h_2$.

Similarly, syntactic attention weights will be multiplied by V_h for each token, and we will get the syntax-attended token representations M^{Syn} :

$$M_h^{Syn} = A_h^{Syn} \times V_h \tag{8}$$

$$M^{Syn} = [M_0^{Syn}, \dots, M_{h_2}^{Syn}] \tag{9}$$

After that, we concatenate syntactic and semantic representation together, and denote it as M :

$$M = M^{Syn} \oplus M^{Sem} \tag{10}$$

Finally, we employ a residual connection to our model. A layer of linear transformation is performed on the feature representation M and then added to the input matrix X (see Section 3.2). After that, the result followed by a layer normalization [50] operation, and get output representation O :

$$O = LayerNorm(X + (W^{SA}M + b^{SA})) \tag{11}$$

where W^{SA} and b^{SA} are parameters of the linear transformation, and dimensionality of the W^{SA} is $d_f \times d_f$, where d_f is the dimension of word embedding. We can also treat the transformation as a convolution operation with a kernel size of 1.

3.5. Local Context Focus Layer

In this layer, our goal is to capture internal correlation of the local context and let the model focus on features that are beneficial for the RE task.

As shown in Figure 3, local context is part of the input sentence. Based on O from Section 3.4, it will be divided into three parts: head entity $s_{head} = \{O_{h_0}, O_{h_1}, \dots, O_{h_{l_1}}\}$, tail entity $s_{tail} = \{O_{t_0}, O_{t_1}, \dots, O_{t_{l_2}}\}$, and context between the two entities $c_{rel} = \{O_{c_0}, O_{c_1}, \dots, O_{c_{l_3}}\}$, where l_1, l_2, l_3 represents the length of corresponding sequence, respectively. Other parts of the sentence will be masked.

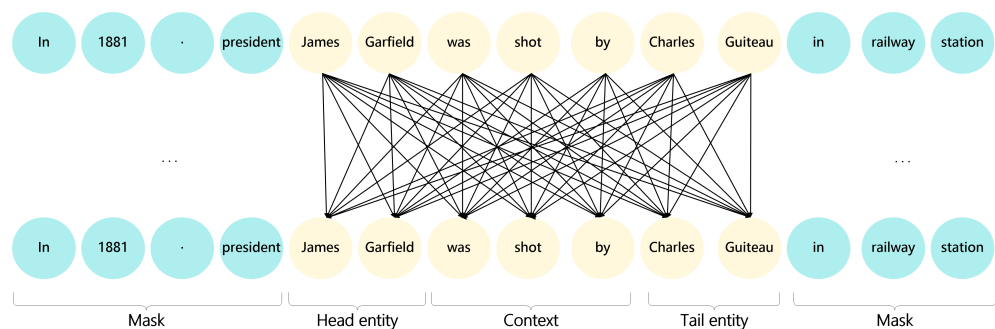


Figure 3. The example of local context focus mechanism, which shows the context-focused process between different tokens.

For local context attention, we calculate the score between each token and other tokens in parallel, and multiply the score with corresponding value feature. In local context, we apply multi-head self-attention (MHSA) to highlight the features that are beneficial to the

task. The specific calculation method can be seen in (3)~(6) and (11). Finally, we can get the local concerned context representation m :

$$m = \text{MHSA}(s_{\text{head}} \oplus c_{\text{rel}} \oplus s_{\text{tail}}) \quad (12)$$

3.6. Relation Extraction

In this section, we complete RE task. Based on m from Section 3.5, we also divide it into three parts $m_{\text{head}}, m_{\text{ctx}}, m_{\text{tail}}$ as well. We use a layer of full connection to classify relations in candidate entity pairs. Due to different lengths of the above three pieces of information, we need to aggregate them so that they can have a fixed length. Here, we use max pooling to aggregate each part, and concatenate them as a whole representation. In addition, we also need span width feature w of two entities. Then, we will get

$$m_{\text{loc}} = \text{MaxPool}(m_{\text{head}}) \oplus \text{MaxPool}(m_{\text{ctx}}) \oplus \text{MaxPool}(m_{\text{tail}}) \quad (13)$$

$$m' = m_{\text{loc}} \oplus w_{\text{head}} \oplus w_{\text{tail}} \quad (14)$$

$$P^r = \text{sigmoid}(W^r \times m' + b^r) \quad (15)$$

Since there may be multiple relations between two entities, we regard our RE task as a multi-label binary classification problem. From (15), each pair of candidate entities will generate probabilities corresponding to all the predefined relations, and then we set a threshold to filter out the results with low confidence.

3.7. Model Training

We employ BERT as our pre-trained language model, and its parameters will be fine-tuned during training. Since the joint model contains two tasks, the loss function will be designed as the sum of the two task losses.

Both parts of the loss function, use the form of cross entropy as follows:

$$L = - \sum_{i \in S} y_i^e \log P_i^e - \sum_{i \in S'} \sum_{j \in S'} y_{ij}^r \log P_{ij}^r (i \neq j) \quad (16)$$

The first part is the loss calculated from classifying all candidate spans, and the second part is obtained by classifying relations from candidate entity pairs.

4. Experiments

4.1. Datasets and Experimental Settings

We conduct experiments on two relation extraction datasets and achieve significant improvements on both corpora.

The first dataset is Conll04 [51], which contains sentences with annotated name entities and relations from news articles. We follow the processing method of the dataset in [40] and split the data into training set, validation set, and test set. The test set consists of 288 sentences. The training set and validation set contain 1153 sentences in total, and we use 20% of them for validation. For Conll04, it defines four entity types (*Person, Organization, Location, and Other*) and five relation categories (*Live-in, Work-for, OrgBased-in, Kill, and Located-in*).

The second dataset is SciERC [52], which contains 500 scientific abstracts that come from 12 AI conference/workshop proceedings in four AI communities. We process the dataset according to [52]. There are 551 sentences for the test set, and 1861 and 275 sentences for the training and validation sets, respectively. For SciERC, it defines six entity types (*Method, Metric, Task, Material, Generic, and Other-ScientificTerm*) and seven relation types (*Feature-of, Used-for, Compare, Part-of, Hyponym-Of, Conjunction, and Evaluate-for*).

Some statistical data of the two datasets are shown in Table 1.

Table 1. Statistics of experimental datasets.

	Conll04	SciERC
Entity types	4	6
Relation types	5	7
Training tuples	1283	3219
Test tuples	422	974

In our experiments, we measure NER and RE performance by computing precision, recall, and F-measure score on the test set. To evaluate the experimental results, we adopted the standard formula to calculate the scores as (17)–(19):

$$Precision = \frac{\text{correct positive predict}}{\text{all samples predicted as positive}} \quad (17)$$

$$Recall = \frac{\text{correct positive predict}}{\text{all positive samples}} \quad (18)$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (19)$$

Following previous works, for Conll04, we report both micro average and macro average F1 score. For micro-F1, we calculate the metric globally by counting the total true positives, false negatives, and false positives. For macro-F1, we calculate the metric for each label, and get their unweighted mean. A relation is correct if its type and corresponding two entities are all correct and an entity is considered correct if its span and type are both right. For SciERC, we report micro-F1 score as previous works and, in the RE task, we do not consider whether the entity type is correct.

In this work, we employ BERT as our pre-trained model, and its parameters will be fine-tuned during training. For the Conll04 dataset, we will adopt BERT-base-cased from HuggingFace’s Transformers [53]. In addition, for SciERC, the SciBERT [54] model will be adopted, which is trained on papers from the corpus of *semanticsholar.org* specifically. For the acquisition of syntactic features, we use the Stanford dependency parser of the Stanford CoreNLP toolkit [55] to generate dependency structures. We train the model with a learning rate of 5×10^{-5} , and adopt the Adam optimizer with weight decay as 0.01, using a linear warmup schedule. The dimension of width embedding is 25, and the batch size is set to 2. The contextual embedding size from BERT is 768, and the threshold for filtering relations is 0.55. For all layers with multi-head self-attention, the number of heads is 8, and we use half of the heads to focus on syntactic features. In addition, for span pruning, we limit spans to a max length of 10. All of the hyper-parameters are tuned on the validation set.

4.2. Overall Performance Comparison

Table 2 demonstrates the overall experimental results. For our joint model, we report both named entity recognition and relation extraction task performance on the test dataset. By applying syntax-informed self-attention mechanism with the LCA pruning method and local context focus mechanism, we observe that our model outperforms all listed joint models and achieves significant improvements on relation extraction tasks. For Conll04, we find that our model outperforms the previous state-of-the-art model [14] by 1.70 micro-F1 and 1.61 macro-F1. To evaluate the generalizability of our method, we also train and evaluate the model on the SciERC dataset. For SciERC, we find that our model has 1.67 micro-F1 more than previous state-of-the-art work. In addition, our proposed model improves upon other models in both precision and recall. All results demonstrate that our model is very effective in joint entity and relation extraction tasks.

Table 2. Comparison of our model with other previous state-of-the-art models. Due to some previous works that use different evaluation calculation metrics, we will indicate the specific calculation method in the table.

		Entity			Relation		
Dataset	Model	Precision	Recall	F1	Precision	Recall	F1
Conll04	Table-filling * [36]	81.20	80.20	80.70	76.00	50.90	61.00
	Multi-head Selector ‡ [10]	83.75	84.06	83.90	63.75	60.43	62.04
	Global Optimization † [41]	-	-	85.60	-	-	67.80
	Multi-turn QA † [47]	89.00	86.60	87.80	69.20	68.20	68.90
	SpERT †/‡ [13]	88.25/85.78	89.64/86.84	88.94/86.25	73.04/74.75	70.00/71.52	71.47/72.87
	Our Model †/‡	88.05/85.29	90.6/87.36	89.31/86.29	75.38 / 76.46	71.09 / 72.64	73.17 / 74.48
	SciIE † [52]	67.20	61.50	64.20	47.60	33.50	39.30
SciERC	DyGIE † [45]	-	-	65.20	-	-	41.60
	DyGIE++ † [46]	-	-	67.50	-	-	48.40
	SpERT † [13]	70.87	69.79	70.33	53.40	48.54	50.84
	Our Model †	69.65	71.10	70.37	55.28	50.00	52.51

Metrics: micro-average = †, macro-average = ‡, not stated = *.

4.3. Analysis of the Pruning Effect of Syntax Trees

Syntactic features are beneficial for RE tasks because it can capture dependencies between different components of a sentence and usually expresses as tree structure. When we focus on extracting the semantic relationship between two entities in a sentence, some irrelevant information in the dependency tree may interfere with our analysis. Therefore, we choose several different methods for pruning the syntax tree to remove irrelevant information as much as possible. For details about the implementation, see Section 3.4.

The comparison results are shown in Tables 3 and 4. We find that, for both Conll04 and SciERC, the LCA method shows better performance than ALL and SDP. Intuitively, retaining the entire tree may keep too much irrelevant information, and keeping only the shortest dependency path will be too aggressive for removing information. Therefore, using the LCA method can retain effective features more reasonably. Our experiments have also verified this.

Table 3. Comparison of different pruning methods on the Conll04 validation dataset.

Dataset	Pruning Method	Relation F1
Conll04 †/‡	ALL	71.84/73.13
	SDP	72.64/73.58
	LCA	74.66/76.13

Note: we show both micro-average (†) and macro-average (‡) F1 to evaluate relation extraction performance following previous experimental settings.

Table 4. Comparison of different pruning methods on the SciERC validation dataset.

Dataset	Pruning Method	Relation F1
SciERC †	ALL	47.53
	SDP	46.77
	LCA	51.35

Note: we show micro-average (†) F1 to evaluate relation extraction performance following previous experimental settings.

4.4. Ablation Tests

In this section, we conduct ablation tests on Conll04 and SciERC validation datasets reported in Tables 5 and 6 to analyze the effectiveness of different components of our proposed model. We test three variants of the model as follows:

- **All Model:** Use the complete model proposed above for testing;
- **Local Context Focus:** Remove the local attention module for context, and then explore its contribution to overall performance improvement of the model;
- **Syntactic Feature Fusion:** Remove syntactic features from the multi-head self-attention module and then analyze its contribution.

When we remove the local context focus mechanism, the performance of the relation extraction task decreases for both datasets. It contributes 1.39 micro-F1 and 1.44 macro-F1 for Conll04, and 2.45 micro-F1 for SciERC. For the global context, we find that the performance of relation extraction becomes worse if we remove all syntactic heads. It contributes 2.59 micro-F1 and 2.84 macro-F1 for Conll04, and 1.34 micro-F1 for SciERC.

All results described above demonstrate that using part of the heads to focus on syntactic feature is effective. In addition, the pruned syntax tree is an important feature for RE tasks. In addition, by attending to the semantic information of local context, it is possible to exploit more contextual features for the RE task.

Table 5. Ablation test on the Conll04 validation dataset.

Dataset	Settings	Precision	Recall	Relation F1
Conll04 †/‡	All model	77.85/79.00	71.72/73.72	74.66/76.13
	- Local context focus	75.54/76.28	71.14/73.45	73.27/74.69
	- Syntactic feature fusion	75.08/75.65	66.76/68.81	70.68/71.85

Note: we show both micro-average (†) and macro-average (‡) F1 to evaluate relation extraction performance following previous experimental settings.

Table 6. Ablation test on the SciERC validation dataset.

Dataset	Settings	Precision	Recall	Relation F1
SciERC †	All model	55.03	48.13	51.35
	- Local context focus	51.72	46.37	48.9
	- Syntactic feature fusion	53.42	42.86	47.56

Note: we show micro-average (†) F1 to evaluate relation extraction performance following previous experimental settings.

5. Discussion

In this section, we evaluate the prediction results of our proposed model and select several representative samples to discuss advantages and shortcomings of our model. In these cases, the blue part is the correct result, and the red one is wrong. The content in the lower right corner of the right parenthesis denotes “entity pair number-left (L)/right (R) entity-relationship”.

Case 1 (Effect of syntactic feature):

- **Without syntactic feature:** Ambassador Miller is also scheduled to meet with Crimean Deputy [Yevhen Saburov]^{E1-L-Work for} and [[Black Sea Fleet]^{E2-R-Work for}]^{E1-R-Work for} Commander [Eduard Baltin]^{E2-L-Work for}.
- **With syntactic feature:** Ambassador Miller is also scheduled to meet with Crimean Deputy Yevhen Saburov and [Black Sea Fleet]^{E1-R-Work for} Commander [Eduard Baltin]^{E1-L-Work for}.

Case 1 represents a kind of situation where the prediction of a relation between two entities is related based on corresponding entity types, but in fact they are not related in the sentence. We draw a pruned syntax tree of the example by the LCA method illustrated in Figure 4. The subtree in the bottom right corner represents the LCA tree corresponding to the two entities with correct results. For “Black Sea Fleet”, we see that “Eduard Baltin” has a direct modified relationship with it, but “Yevhen Saburov” does not. From the dependency tree, we can obtain a structural modified relationship between entities, which has a positive impact on relation extraction performance. In addition, it is easy to find out that, if we adopt the SDP method, “Commander” will be off the tree. However, “Commander” has

strong instructions for extracting the relation “Work-for”. Therefore, the LCA method can keep more important information.

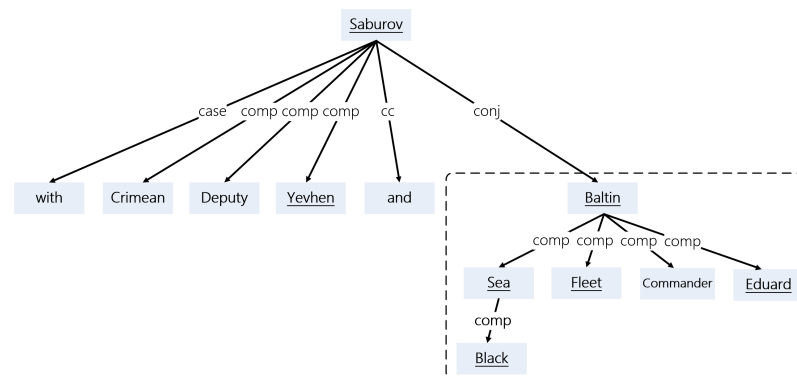


Figure 4. Pruned syntax tree of the example, the underlined parts represent entities information.

Case 2 (Effect of local context focus):

- **Without local context focus:** “If they tell us to get out, we get out,” said [Lutie Dyson]_{E1-L-Live in}, 62, who with her husband and about 65 others took shelter in a school in [Lake Charles]_{E1-R-Live in}.
- **With local context focus:** “If they tell us to get out, we get out,” said [Lutie Dyson]_{E1-L-Live in}, 62, who with her husband and about 65 others took shelter in a school in [Lake Charles]_{E1-R-Live in}.

Case 2 represents a kind of situation where the sentence contains clue words. Although previous work like [13] has proved that extracting local context is superior to the global context for the task, it still lacks in-depth learning of contextual features in local context. In the sentence, we can easily find that the phrase “took shelter in” is an important clue for relation extraction. When we add local focus layer, the model can classify the relation as “Live-in” correctly, but if we do not add this layer and simply use a pooling method over local context to align feature dimensions, the model cannot identify corresponding relations.

Case 3 (Negative sample):

- **Requires simple semantic inferring :** [Oswald]_{E1-L-Kill} was captured in the theater on 22 November 1963, only a few hours after [Kennedy]_{E1-R-Kill} was shot to death on a Dallas street.

Case 3 is a negative example, indicating that there are no straightforward clue words to indicate that there is a relationship “Kill” between entities “Oswald” and “Kennedy”. However, by reading the context, the two events “Kennedy was killed” and “Oswald was arrested” are logically causally connected, and there should be a relationship “Kill” between the two entities.

Although our method makes the model achieve better performance on relation extraction tasks, it is obvious that our model lacks sufficient linguistic inference abilities. From Case 3, it can be found that semantic relational reasoning is an important issue because it allows our model to extract knowledge facts from the text more comprehensively. In addition, the relational reasoning problem usually involves the semantic synthesis of multiple sentences.

In this research field, Ref. [56] proposed a dataset for document-level relation extraction tasks, and summarized several types of relational reasoning like logical, coreference, common sense, and so on. Recently, many works like [57,58] have begun to tackle the problem of cross-sentence relational reasoning, which is an important research direction in the future.

6. Conclusions and Future Work

Enabling machines with cognitive and logical capabilities will be a hot topic in future research, and the knowledge graph will be the “brain” of intelligent applications. The entity and relation extraction task is an important step in constructing knowledge graphs, and comprehensively and accurately extracting knowledge facts from texts has important research and economic value.

In this paper, we propose a model to extract entities and relations from unstructured text, which incorporates syntax-informed multi-head self-attention and a local context focus mechanism in end-to-end framework. In order to integrate syntactic features into a span-based joint extraction model, combined with the observation that the capability of many attention heads in the multi-head attention mechanism is not fully exploited, we propose to employ part of the heads to focus on syntactic parents of each token from dependency trees with different pruning methods to fuse syntactic and semantic features. In addition, we apply a local focus mechanism on entity pairs and corresponding context to get richer contextual features from the local context. Based on pre-trained BERT, results from the experiments on Conll04 and SciERC datasets confirmed that our model achieves significant improvements compared to strong competitors. In addition, we find that using the LCA method to prune a syntax tree in the syntax-informed multi-head self-attention mechanism has the greatest impact on the relation extraction task.

In addition, there are some limitations of this work worthy of further discussion. At present, our model can only extract knowledge facts on sentence-level. However, in fact, a large number of knowledge facts are expressed in multiple sentences. In future, we plan to study how to improve the accuracy of document-level relation extraction tasks. In addition, some semantic relationships need to be inferred by reading and synthesizing multiple sentences in context. Letting the model have powerful linguistic inference ability will be the highlight of our future research.

Author Contributions: Conceptualization, H.Z. and G.Z.; methodology, G.Z.; software, G.Z.; validation, G.Z., H.Z., and Y.M.; formal analysis, G.Z.; investigation, H.Z. and G.Z.; resources, H.Z. and Y.M.; writing—original draft preparation, G.Z.; writing—review and editing, H.Z. and Y.M.; visualization, G.Z.; supervision, H.Z. and Y.M.; project administration, H.Z. and Y.M.; funding acquisition, H.Z. and Y.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (NSFC) via No. 61871046.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Acknowledgments: The authors thank all of the reviewers for their comments on this article.

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

References

1. Yu, M.; Yin, W.; Hasan, K.S.; dos Santos, C.; Xiang, B.; Zhou, B. Improved Neural Relation Detection for Knowledge Base Question Answering. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 571–581.
2. 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; pp. 35–45.
3. 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, Ireland, 25–29 August 2014; pp. 2335–2344.
4. Zhou, P.; Shi, W.; Tian, J.; Qi, Z.; Li, B.; Hao, H.; Xu, B. Attention-based bidirectional long short-term memory networks for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Berlin, Germany, 7–12 August 2016; pp. 207–212.

5. Wang, L.; Cao, Z.; De Melo, G.; Liu, Z. Relation classification via multi-level attention cnns. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 1298–1307.
6. Miwa, M.; Bansal, M. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 1105–1116.
7. Katiyar, A.; Cardie, C. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 917–928.
8. Zheng, S.; Wang, F.; Bao, H.; Hao, Y.; Zhou, P.; Xu, B. Joint Extraction of Entities and Relations Based on a Novel Tagging Scheme. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 1227–1236.
9. Zeng, X.; Zeng, D.; He, S.; Liu, K.; Zhao, J. Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15–20 July 2018; pp. 506–514.
10. Bekoulis, G.; Deleu, J.; Demeester, T.; Develder, C. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Syst. Appl.* **2018**, *114*, 34–45. [[CrossRef](#)]
11. Fu, T.J.; Li, P.H.; Ma, W.Y. GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1409–1418.
12. Dixit, K.; Al-Onaizan, Y. Span-Level Model for Relation Extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 5308–5314.
13. Eberts, M.; Ulges, A. Span-based Joint Entity and Relation Extraction with Transformer Pre-training. In Proceedings of the 24th European Conference on Artificial Intelligence, Santiago de Compostela, Spain, 29 August–5 September 2020.
14. 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 30th Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4 December–9 December 2017; pp. 5998–6008.
15. Li, J.; Tu, Z.; Yang, B.; Lyu, M.R.; Zhang, T. Multi-Head Attention with Disagreement Regularization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2897–2903.
16. Voita, E.; Talbot, D.; Moiseev, F.; Sennrich, R.; Titov, I. Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 5797–5808.
17. Michel, P.; Levy, O.; Neubig, G. Are Sixteen Heads Really Better than One? In Proceedings of the 32th Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8 December–14 December 2019; pp. 14014–14024.
18. Vig, J. A Multiscale Visualization of Attention in the Transformer Model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Florence, Italy, 28 July–2 August 2019; pp. 37–42.
19. 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, 2–7 June 2019; pp. 4171–4186.
20. Du, C.; Sun, H.; Wang, J.; Qi, Q.; Liao, J. Adversarial and domain-aware bert for cross-domain sentiment analysis. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 4019–4028.
21. Yamada, I.; Asai, A.; Shindo, H.; Takeda, H.; Matsumoto, Y. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. *arXiv* **2020**, arXiv:2010.01057.
22. He, P.; Liu, X.; Gao, J.; Chen, W. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. *arXiv* **2020**, arXiv:2006.03654.
23. Zelenko, D.; Aone, C.; Richardella, A. Kernel methods for relation extraction. *J. Mach. Learn. Res.* **2003**, *3*, 1083–1106.
24. Bunescu, R.C.; Mooney, R.J. A shortest path dependency kernel for relation extraction. In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. Association for Computational Linguistics, Vancouver, BC, Canada, 6–8 October 2005; pp. 724–731.
25. Xu, K.; Feng, Y.; Huang, S.; Zhao, D. Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 536–540.
26. Xu, Y.; Mou, L.; Li, G.; Chen, Y.; Peng, H.; 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, Lisbon, Portugal, 17–21 September 2015; pp. 1785–1794.
27. Cai, R.; Zhang, X.; Wang, H. Bidirectional recurrent convolutional neural network for relation classification. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, 7–12 August 2016; pp. 756–765.
28. He, Z.; Chen, W.; Li, Z.; Zhang, M.; Zhang, W.; Zhang, M. SEE: Syntax-aware entity embedding for neural relation extraction. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.

29. Tai, K.S.; Socher, R.; Manning, C.D. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, 26–31 July 2015; pp. 1556–1566.
30. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the International Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017.
31. Zhang, Y.; Qi, P.; Manning, C.D. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2205–2215.
32. 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; pp. 8021–8032.
33. Shen, Y.; Tan, S.; Sordoni, A.; Courville, A. Ordered neurons: Integrating tree structures into recurrent neural networks. In Proceedings of the International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May 2019.
34. Shaw, P.; Uszkoreit, J.; Vaswani, A. Self-Attention with Relative Position Representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers), New Orleans, LA, USA, 1–6 June 2018; pp. 464–468.
35. Wang, X.; Tu, Z.; Wang, L.; Shi, S. Self-Attention with Structural Position Representations. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 1403–1409.
36. Miwa, M.; Sasaki, Y. Modeling joint entity and relation extraction with table representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1858–1869.
37. Li, Q.; Ji, H. Incremental joint extraction of entity mentions and relations. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 22–27 June 2014; pp. 402–412.
38. Ren, X.; Wu, Z.; He, W.; Qu, M.; Voss, C.R.; Ji, H.; Abdelzaher, T.F.; Han, J. Cotype: Joint extraction of typed entities and relations with knowledge bases. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 1015–1024.
39. Zheng, S.; Hao, Y.; Lu, D.; Bao, H.; Xu, J.; Hao, H.; Xu, B. Joint entity and relation extraction based on a hybrid neural network. *Neurocomputing* **2017**, *257*, 59–66. [[CrossRef](#)]
40. Gupta, P.; Schütze, H.; Andrassy, B. Table filling multi-task recurrent neural network for joint entity and relation extraction. In Proceedings of the COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, Japan, 11–16 December 2016; pp. 2537–2547.
41. Zhang, M.; Zhang, Y.; Fu, G. End-to-end neural relation extraction with global optimization. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; pp. 1730–1740.
42. Chi, R.; Wu, B.; Hu, L.; Zhang, Y. Enhancing Joint Entity and Relation Extraction with Language Modeling and Hierarchical Attention. In Proceedings of the Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data, Chengdu, China, 1–3 August 2019; Springer: Berlin, Germany, 2019; pp. 314–328.
43. Nguyen, D.Q.; Verspoor, K. End-to-end neural relation extraction using deep biaffine attention. In Proceedings of the European Conference on Information Retrieval, Cologne, Germany, 14–18 April 2019; Springer: Berlin, Germany, 2019; pp. 729–738.
44. Wang, Y.; Yu, B.; Zhang, Y.; Liu, T.; Zhu, H.; Sun, L. TPLinker: Single-stage Joint Extraction of Entities and Relations Through Token Pair Linking. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 13–18 September 2020; pp. 1572–1582.
45. Luan, Y.; Wadden, D.; He, L.; Shah, A.; Ostendorf, M.; Hajishirzi, H. A general framework for information extraction using dynamic span graphs. 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, 2–7 June 2019; pp. 3036–3046.
46. Wadden, D.; Wennberg, U.; Luan, Y.; Hajishirzi, H. Entity, Relation, and Event Extraction with Contextualized Span Representations. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 5788–5793.
47. Li, X.; Yin, F.; Sun, Z.; Li, X.; Yuan, A.; Chai, D.; Zhou, M.; Li, J. Entity-Relation Extraction as Multi-Turn Question Answering. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1340–1350.
48. Wu, Y.; Schuster, M.; Chen, Z.; Le, Q.V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv* **2016**, arXiv:1609.08144.
49. Liu, X.; Duh, K.; Liu, L.; Gao, J. Very deep transformers for neural machine translation. *arXiv* **2020**, arXiv:2008.07772.
50. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer normalization. *arXiv* **2016**, arXiv:1607.06450.
51. 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; pp. 1–8.

52. Luan, Y.; He, L.; Ostendorf, M.; Hajishirzi, H. Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 3219–3232.
53. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; Brew, J. HuggingFace’s Transformers: State-of-the-art Natural Language Processing. *arXiv* **2019**, arXiv:1910.03771.
54. Beltagy, I.; Lo, K.; Cohan, A. SciBERT: A pretrained language model for scientific text. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 3606–3611.
55. Manning, C.D.; Surdeanu, M.; Bauer, J.; Finkel, J.R.; Bethard, S.; McClosky, D. The Stanford CoreNLP natural language processing toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, MD, USA, 22–27 June 2014; pp. 55–60.
56. Yao, Y.; Ye, D.; Li, P.; Han, X.; Lin, Y.; Liu, Z.; Liu, Z.; Huang, L.; Zhou, J.; Sun, M. DocRED: A Large-Scale Document-Level Relation Extraction Dataset. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 764–777.
57. Nan, G.; Guo, Z.; Sekulic, I.; Lu, W. Reasoning with Latent Structure Refinement for Document-Level Relation Extraction. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 1546–1557.
58. Zeng, S.; Xu, R.; Chang, B.; Li, L. Double Graph Based Reasoning for Document-level Relation Extraction. *arXiv* **2020**, arXiv:2009.13752.