electronics

MDPI

Article

A Triple Relation Network for Joint Entity and Relation Extraction

Zixiang Wang [1], Liqun Yang [2], Jian Yang [1], Tongliang Li [1], Longtao He [3] and Zhoujun Li [1,*]

[1] State Key Lab of Software Development Environment, Beihang University, Beijing 100191, China; wangzixiang@buaa.edu.cn (Z.W.); jiaya@buaa.edu.cn (J.Y.); tonyliangli@buaa.edu.cn (T.L.)
[2] School of Cyber Science and Technology, Beihang University, Beijing 100191, China; lqyang@buaa.edu.cn
[3] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China; hlt@cert.org.cn
* Correspondence: lizj@buaa.edu.cn

Abstract: Recent methods of extracting relational triples mainly focus on the overlapping problem and achieve considerable performance. Most previous approaches extract triples solely conditioned on context words, but ignore the potential relations among the extracted entities, which will cause incompleteness in succeeding Knowledge Graphs' (KGs) construction. Since relevant triples give a clue for establishing implicit connections among entities, we propose a Triple Relation Network (TRN) to jointly extract triples, especially handling extracting implicit triples. Specifically, we design an attention-based entity pair encoding module to identify all normal entity pairs directly. To construct implicit connections among these extracted entities in triples, we utilize our triple reasoning module to calculate relevance between two triples. Then, we select the top-K relevant triple pairs and transform them into implicit entity pairs to predict the corresponding implicit relations. We utilize a bipartite matching objective to match normal triples and implicit triples with the corresponding labels. Extensive experiments demonstrate the effectiveness of the proposed method on two public benchmarks, and our proposed model significantly outperforms previous strong baselines.

Keywords: entity and relation extraction; relational triple; triple relation network; implicit triples

Citation: Wang, Z.; Yang, L.; Yang, J.; Li, T.; He, L.; Li, Z. A Triple Relation Network for Joint Entity and Relation Extraction. *Electronics* **2022**, *11*, 1535. https://doi.org/10.3390/electronics11101535

Academic Editors: Andrea Prati, Luis Javier García Villalba, Vincent A. Cicirello and Gemma Piella

Received: 8 April 2022
Accepted: 7 May 2022
Published: 11 May 2022

1. Introduction

Relational triples' extraction aims to predict the semantic relation between the two entities in an unstructured text, which plays a vital role in succeeding Knowledge Graphs' (KGs) construction. The relational triples are normally presented in the form of (subject, relation, object). Consider this sentence: "Beijing is known as the capital of China." From the sentence, an ideal triple extraction would comprise Beijing, Capital_of, China, in which Capital_of is the relation of Beijing and China. Therefore, the relational triple in this example is expressed as (Beijing, Capital_of, China).

Early works in relational triple extraction [1,2] adopted a pipeline manner to extract entities, classified relations in two separate processes, and achieved promising performance. The pipeline methods suffer from the error propagation problem, lacking interaction between the two stages. To address the problem, subsequent works proposed joint learning of entities and relations, which are mainly categorized as feature-based [3–6] and neural-network-based [7,8]. To tackle extensive manual efforts in feature-based models, recent studies observed neural-based models to perform better in the task. However, neural-based models like [8] fail to extract overlapping triples that the entities share with other triples in a sentence, as Figure 1 illustrates. Many studies, such as sequence-tagging-based [9–14], sequence-to-sequence (Seq2Seq)-based [15–20], and table-filling-based [21–24] methods mainly handle the overlapping problem.

However, most existing methods only extract triples via context words that are explicitly expressed in a sentence [25]. In fact, there are many implicit relations between

entities. Figure 2 shows an example: the explicit relations **Work_for** and **Locate_in** can be easily recognized by recent models via context words **played for** and **in**, respectively, but hardly recognize the implicit relation **Live_in** due to the lack of the explicit expression in the sentence. That means many extracted entities that have certain connections with each other could be ignored. Consequently, such a scenario will cause the incompleteness of KGs' construction [26–28]. To handle these scenarios, the model needs to be equipped with a reasoning ability.

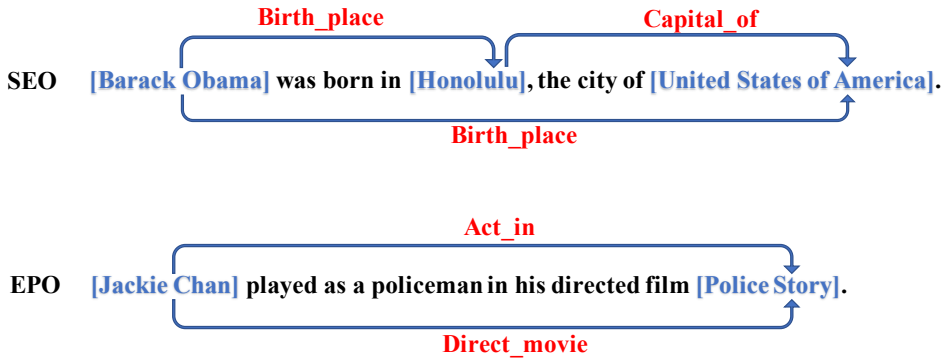

Figure 1. The two examples of overlapping types. Single Entity Overlap (SEO) and Entity Pair Overlap (EPO) overlapping patterns.

Figure 2. An example of an explicit relation and an implicit relation. The underlined words are context words. The red arrow line means the implicit relation between entity pairs without explicit context words in a sentence.

Given the deficiency of the reasoning ability in previous works, we aim to explore more potential information in the raw sentence. The relevant triples provide critical evidence in establishing implicit connections among entities. As shown in Figure 2, the entity pairs (Fan Zhiyi, Shanghai Shenhua) and (Shanghai Shenhua, Shanghai) are overlapped with **Shanghai Shenhua**, which gives a strong clue to connect with entities **Fan Zhiyi** and **Shanghai**. The two selected entities are combined into a new triple (Fan Zhiyi, Live_in, Shanghai). The implicit relation **Live_in** can be inferred from the triple pair structures (A, Work_for, B) and (B, Locate_in, C), which means the implicit connections can be learned from the structures. Therefore, we extract the critical evidence from the relevant relational triples.

In this paper, we propose a **T**riple **R**elation **N**etwork (TRN) to address the implicit relational triples' extraction problem. Given a sentence, TRN first utilizes the encoder to obtain the token representations. Most joint models first extract possible head entities and then match with all tail entities, leading to high complexity (N^2 entity pairs) and performance degradation introduced by unrelated entity pairs. To solve this problem, we design an attention-based entity pair encoding module to obtain the entity pair representations. We obtain the corresponding relation representations for normal entity pairs. A triple reasoning module is introduced to calculate the probability of each triple (concatenation of entity pairs and their relations) pair for reasoning a new triple to find the implicit entity pairs. The higher probability indicates that there is an implicit connection among entities in two triples. Our triple generation module is used to decode the entities in a span-based manner. To calculate the loss of implicit triples and normal triples with the corresponding training labels, we utilize the bipartite matching loss in the training stage. The experimental

results show that our proposed TRN outperforms the previous strong methods on two benchmarks and performs better in extracting implicit triples.

Our main contributions include the following:

- We propose a novel **T**riple **R**elation **N**etwork framework to extract the relational triples, especially implicit triples existing in raw sentences. This implies that the designed model possesses the power of reasoning to search for comprehensive information in the triples' extraction task.
- We instantiate the proposed TRN, including an attention-based entity pair encoding module to collect normal entity pairs directly and a triple reasoning module to find the implicit connections among all entities. Finally, a bipartite matching training objective is introduced to update the model parameters.
- The main experiments show that the proposed model outperforms the previous strong baselines on two benchmarks, and analytic experiments verify our motivation in potential relations among the extracted entities.

2. Related Work

Relational triples' extraction [29] is a task in which factual knowledge is mined from texts. It is a well-studied task in information extraction. It is also an important step for the construction of large-scale KGs such as DBpedia [30], Freebase [31], and Knowledge Vault [32]. The task is mainly divided into two manners.

Early studies, such as [1,2], addressed the task in a pipeline manner. They extracted both entities and relations, where entities are extracted first, and then, the relation between the extracted entities is predicted. These methods ignore the interactions between entities and relations and cause the error propagation problem.

To prevent error propagation in the pipeline manner, subsequent studies addressed the task in a joint entity and relation extraction manner, which jointly extracts entities and relations. Traditional joint models are feature-based methods [3–6], which heavily rely on feature engineering with manual efforts. To eliminate hand-crafted features, recent studies proposed the neural-network-based method, which achieves state-of-the-art performance. However, most joint neural models, such as [7], extract entity pairs and predict the corresponding relations between them through parameter sharing, like the pipeline manner, in which the decoding setting is separated and still leads to an error propagation to a certain extent. Unlike parameter sharing, [8] introduced a unified tagging scheme that transforms the task into an end-to-end sequence tagging problem. Such a joint decoding setting identifies the entities and relations in one shot instead of two separate processes, Named Entity Recognition (NER) and Relation Classification (RC), and naturally handle the error propagation problem.

Most of the previous joint models suffer from the overlapping problem in which triples are overlapped with one or two entities in a sentence. The problem directly affects sequence tagging models like [8], since they only recognize each entity token by one tag. To solve this problem, subsequent studies can be roughly categorized into three frameworks: sequence-tagging-based [9–14], sequence-to-sequence-based [15–20], and table-filling-based [21–24]. Sequence tagging models the entities and relations' extraction task as a tagging problem. To extract an entity (subject or object), the tagging sequences are used to recognize the start position and end position and sometimes determine the relations simultaneously based on the subjects extracted. Sequence-to-sequence models regard the triples as a sequence and view the entities and relations' extraction task as a generation task. Basically, the method uses an encoder–decoder architecture just like other generation tasks in NLP. Table filling converts the entities and relations' extraction task into filling an $n \times n$ table problem (n is the number of tokens in a sentence). The table elements usually indicate the start and end positions of two entities or the entity type of each entity.

Despite their success, most methods only capture the connection between two entities by explicit context words in a sentence. Unfortunately, it is hard for them to identify relational facts, which are not explicitly expressed in a sentence [25], leading to the under-

utilization of resources and the incompleteness of KGs' construction [26,27]. To capture the inner connections, [10] proposed a method named TME to use entity embedding translation to predict relations. However, TME needs a complicated tagging scheme (position, type, relation) and cannot handle symmetric relations.

To overcome the aforementioned problems, we designed a framework called TRN. Following the Seq2Seq-based methods, we modeled triples as a sequence and naturally dealt with the triple overlapping problem. We propose a triple reasoning module to construct implicit connections among these entity pairs. It can find the two relevant triples to infer a new triple and obtain these two implicit entities inside them.

3. Proposed Framework

In this section, we introduce the proposed model TRN. The main architecture of TRN is shown in Figure 3. Section 3.1 introduces our sentence encoder. Section 3.2 is our entity pair encoding module. Section 3.3 is our triple reasoning module used to find implicit entity pairs. Section 3.4 shows our triple generation module. Section 3.5 describes the details of the bipartite matching training objective.

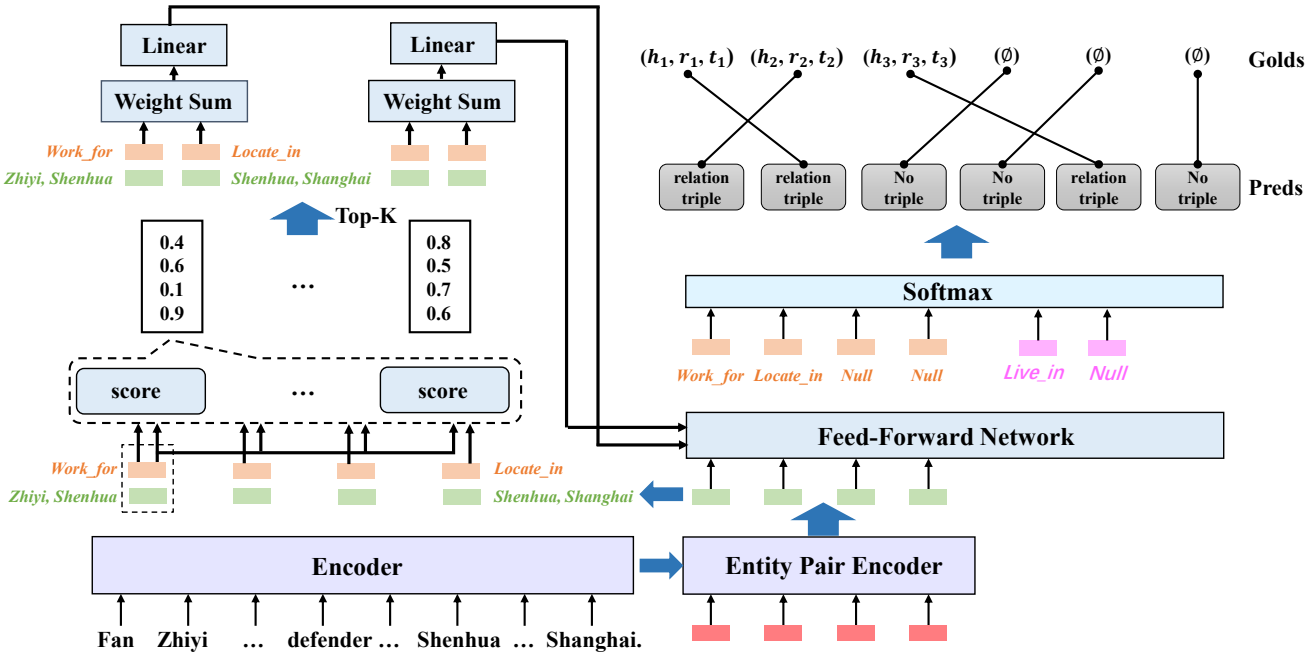

Figure 3. The framework of the proposed model. The red box is a raw entity pair representation. The green box is an encoded entity pair representation. The orange box is a relation representation. The pink box is a relation corresponding to an implicit entity pair. As illustrated in the left upper part, we set $k = 2$, which means only 2 triple pairs are selected in the final inference.

3.1. Sentence Encoder

Let $S = [w_1, w_2, \ldots, w_m]$ be an input sentence, where w_i $(1 \leq i \leq m)$ is the i-th token in the sentence and m is the sentence length. We utilized the BERT [33] model as the encoder. The output of the encoder is denoted as follows:

$$H = \{h_1, h_2, \ldots, h_m\} \tag{1}$$

where h_i $(1 \leq i \leq m) \in \mathbb{R}^d$ is the encoding representation of the i-th token in a sentence (including start symbol `[CLS]` and end symbol `[SEP]`) and d is the hidden size.

3.2. Entity Pair Encoding

After we obtained the token representations of the current sentence from the sentence encoder, we randomly initialized n-sized raw entity pairs as the initial ones, which

are explicitly recognized by context words. Similar to the decoder of [34], we used the self-attention and cross-attention parts as the entity pair encoding module. Through the self-attention layer, we can capture the dependency information regardless of the distance between entity pairs. Through the cross-attention layer, we can capture contextual information to establish an entity pair representation. Since there is no causal mask in the self-attention layer, we fused the information into raw entity pairs in parallel. Thus, we set n to a relatively large number to ensure we can extract enough entity pairs in a sentence. Consequently, the representations of n entity pairs $E = \left\{e_1, e_2, \ldots, e_n \mid e_i \in \mathbb{R}^d\right\}$ is computed as:

$$E = EntityPairEncoding(H; E_{raw}) \tag{2}$$

where E_{raw} denotes n raw entity pairs.

We utilized multi-head attention [35] as the entity pair encoding module. The details of self-attention and cross-attention are described as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V \tag{3}$$

where Q, K, V are three matrices that are used to transform embeddings to query, key, and value. $\sqrt{d}$ is the scaling factor used to control the dot product value. We set $Q = K = V = E_{raw}$ for the self-attention layer and $K = V = H$ for the cross-attention layer.

The multi-head attention is defined by:

$$head_i = Attention(QW_i^q, KW_i^k, VW_i^v) \tag{4}$$

where W_i^q, W_i^k, W_i^v are learnable matrices and $head_i$ is the i-th head representation.

Finally, we obtain the entity pair representations:

$$\mathbb{E} = Concat(head_1, ..., head_o)W^o \tag{5}$$

where o is the number of heads in the attention mechanism and W^o is the learnable matrix.

3.3. Triple Reasoning

The triple reasoning module aims to find the implicit entity pairs among relational triples. Due to the lack of context words to recognize the connection between two implicit entities, we can only rely on sufficient information provided by the extracted triples, e.g., in Figure 2, (Fan Zhiyi, Live_in, Shanghai) is reasoned by (Fan Zhiyi, Work_for, Shanghai Shenhua) and (Shanghai Shenhua, Locate_in, Shanghai). We also considered a single triple reasoning case, e.g., from the triple (Beijing, Capital_of, China), we can obtain implicit triple (Beijing, Locate_in, China) or (China, Contain, Beijing), which infers the new relation Locate_in or Contain from the two same entities. We treated the single triple reasoning also as the triple pair reasoning by simply utilizing the triple and its duplicate as the input.

Basically, we utilized the triples to reason the implicit entity pairs in three steps. First, for each entity pair representation, we collected the relation representation between them to form the triple representation. Then, we calculated the relevant scores with all other triples (including itself) to obtain the probabilities for reasoning a new triple. Last, we used the top-K strategy to select a fixed number of implicit entity pairs.

Formally, given the current entity pair e_i and corresponding relation type $r_i \in \mathbb{R}^d$, the probabilities of k implicit entity pairs are defined as:

$$P\left(E_{imp}\right) = \underset{i,j \in \{1,2,\ldots,n\}}{\arg\max} \prod^{k} p\left(e_{ij}^{imp} \mid e_i \| r_i, e_j \| r_j; \theta\right) \tag{6}$$

where E_{imp} is the entity pair set, k is the number of implicit triples that we set from our strategy described in the **top-K strategy**, and $e_{ij}^{imp} \in \mathbb{R}^d$ is the implicit entity pair in which i and j are the indices of two selected triples. The entity pair-embeddings- and

relation-embeddings-based feature vectors are concatenated with $\|$ to obtain the triples' representation.

Triple Collection: To infer the potential relationship between two entities, we need to capture the information from the triples that we predicted through explicit expressions. In this part, we calculate the relation representation r_i for each entity pair e_i:

$$r_i = FFN(e_i) \tag{7}$$

where FFN is a feed-forward network. Then, we obtain the triple representation $t_i = (e_i \| r_i) \in \mathbb{R}^{2d}$ as the input of the left upper part of Figure 3.

Scoring Pool: We computed the probability to decide whether an implicit entity pair could be constructed via a triple pair. Specifically, given the input set of triples $T = [t_1, t_2, \ldots, t_n]$, for any triple t_i coupled with t_j, the score ω^{ij} is computed as:

$$\omega^{ij} = Relu(W^s \cdot (t_i \| t_j) + b^s), (i, j = 1, 2, ..., n) \tag{8}$$

where $W^s \in \mathbb{R}^{4d \times 1}$ and b^s are learnable parameters. Figure 4 corresponds to Equation (8). Note that since not all triple pairs are useful for predicting the new one, the triple that is irrelevant with respect to the others is supposed to be set at lower scores. Therefore, we did not use the attention mechanism [36] to calculate the probabilities with others because all weight scores were restricted to sum to 1. The higher the weight score, the higher the probability of the triple pair inferring the implicit triple is. Finally, we formed a scoring pool that collects all the scores calculated in a total of n triples with each other.

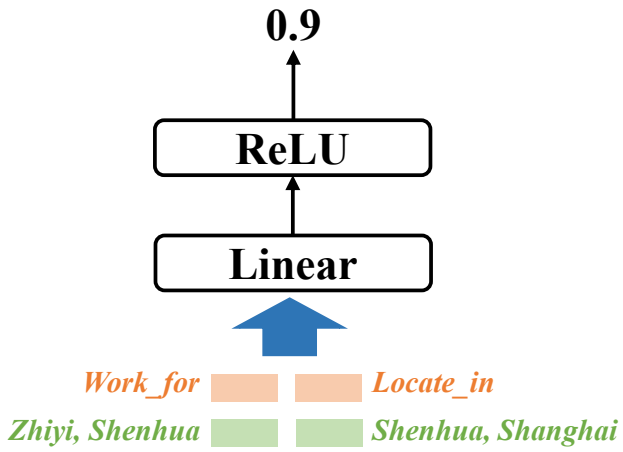

Figure 4. The details of the score operation in Figure 3. The orange box and the green box are concatenated to be a triple representation. Two triples are used to compute the probability to infer a new triple.

Top-K Strategy: We introduced our top-K strategy. We selected the top-k relevant triple pairs based on the scoring pool as candidates, which were used for inferring implicit triples. When $k = 1$, our strategy is downgraded to be a "max" approach in selecting the best triple pair to conduct reasoning. The output is a weighted sum of the features of implicit entity pair e_{ij}^{imp} inferred from triple pair t_i and t_j. Let $\mathcal{K}$ denote the indices of the best k triple pairs. The implicit entity pair is computed as:

$$e_{ij}^{imp} = W^{\psi} \cdot \omega^{ij}(t_i \| t_j), \forall (i, j) \in \mathcal{K} \tag{9}$$

where $W^{\psi} \in \mathbb{R}^{4d \times d}$. In this paper, we set $k = 5$ as the default.

3.4. Triple Generation

Through the triple reasoning module, we obtained k implicit entity pairs and added them into E. All entities are recognized from the start positions to the end positions in the

sentence with m tokens. We decided the positions of two entities that gave the max values of four pointer probabilities to form the best entity pair combination.

Given an entity pair e_i from E, the classification can be obtained by:

$$p^r = MLP_r(e_i) \tag{10}$$

$$H_{fuse} = Dup(e_i, m) + W^{x,y}H \tag{11}$$

$$p^{x,y} = MLP_{x,y}(H_{fuse}) \tag{12}$$

where p^r denotes the probability of the relation type, $p^{x,y}$ is the probabilities of starting or ending indices for an entity pair, and superscripts x, y are the combinations of $head/tail$ and $start/end$, i.e., $p^{head,start}$, $p^{head,end}$, $p^{tail,start}$, $p^{tail,end}$. MLP denotes multi-layer perceptron (same as the FFN of Equation (7)) with the softmax function. $W^{x,y}$ (superscript combinations are the same as $p^{x,y}$) are learnable parameters. $Dup(\cdot)$ will duplicate e_i m times into shape $\mathbb{R}^{m\times d}$. Therefore, each predicted triple in the set is denoted as $\hat{y}_i = (p_i^r, p_i^{head,start}, p_i^{head,end}, p_i^{tail,start}, p_i^{tail,end})$, and each golden triple in our training labels is preprocessed as the same structure $y_i = (r_i, head_i^{start}, head_i^{end}, tail_i^{start}, tail_i^{end})$.

3.5. Training Objective

Since our training labels are disordered, the main difficulty in training is that our predicted triples must be in line with the order of the golden triples. Unfortunately, there are no annotations of implicit triples in either dataset, so it is impossible to be certain that the predicted implicit triples exactly correspond to the position of the golden implicit triples. Besides, the model extremely focuses on predicting the order of multiple triples instead of learning how to generate accurate triples, which hurts the performance in the training stage. To cope with these permutation problems, we need to find an optimal matching strategy between the golden triple set and the predicted triple set before computing the training objective. We used a set prediction loss, which can produce an optimal bipartite matching by using the Hungarian algorithm [37]. To obtain an optimal matching, we searched for a permutation of n elements $\hat{\alpha} \in \Omega(n+k)$ with the lowest cost as the first step:

$$\hat{\alpha} = \underset{\alpha\in\Omega(n+k)}{\arg\min} \sum_{i}^{n+k} \mathcal{L}_{match}(y_i, \hat{y}_{\alpha(i)}) \tag{13}$$

where $\mathcal{L}_{match}(y_i, \hat{y}_{\alpha(i)})$ is a triple pair matching cost, y_i is a golden triple, and $y_{\alpha(i)}$ is a prediction with index $\alpha(i)$. $\Omega(n+k)$ is the space of all permutations of an $n+k$-sized set. $\mathcal{L}_{match}(y_i, \hat{y}_{\alpha(i)})$ is defined as:

$$\begin{aligned}\mathcal{L}_{\text{match}}\left(y_i, \hat{y}_{\alpha(i)}\right) = -1_{\{r_i\neq\varnothing\}}\Big[&p^r_{\alpha(i)}(r_i) \\ &+ p^{\text{head,start}}_{\alpha(i)}\left(\text{head}_i^{\text{start}}\right) \\ &+ p^{\text{head,end}}_{\alpha(i)}\left(\text{head}_i^{\text{end}}\right) \\ &+ p^{\text{tail,start}}_{\alpha(i)}\left(\text{tail}_i^{\text{start}}\right) \\ &+ p^{\text{tail,end}}_{\alpha(i)}\left(\text{tail}_i^{\text{end}}\right)\Big]\end{aligned} \tag{14}$$

Then, we obtain the optimal matching $\hat{\alpha}$ between the permutation of golden triples and predictions. We define the final loss $\mathcal{L}_{loss}$ as:

$$\begin{aligned}\mathcal{L}_{\text{loss}}(y, \hat{y}) = \sum_{i=1}^{n+k} \Big\{ & -\log p^{r}_{\hat{\alpha}(i)}(r_i) \\ & + 1_{\{r_i \neq \varnothing\}} \Big[-\log p^{\text{head,start}}_{\hat{\alpha}(i)} \left(\text{head}^{\text{start}}_{i} \right) \\ & - \log p^{\text{head,end}}_{\hat{\alpha}(i)} \left(\text{head}^{\text{end}}_{i} \right) \\ & - \log p^{\text{tail,start}}_{\hat{\alpha}(i)} \left(\text{tail}^{\text{start}}_{i} \right) \\ & - \log p^{\text{tail,end}}_{\hat{\alpha}(i)} \left(\text{tail}^{\text{end}}_{i} \right) \Big] \Big\}\end{aligned} \tag{15}$$

where $\hat{\alpha}(i)$ is the element of the optimal permutation.

4. Experiments and Discussion

4.1. Experimental Setup

Datasets In the experimental setup, we evaluated our proposed model on the two widely used public datasets: New York Times (NYT) [38] and WebNLG [39].

NYT collects sentences from the New York Times corpus, and it contains 24 predefined relation types. Since there are many versions of the NYT dataset, we adopted the preprocessed version used in [15].

WebNLG contains 171 (The number is miswritten by many works. We followed the correct number from [23].) predefined relation types. The dataset was originally created for the Natural Language Generation (NLG) task.

We split the sentences into three categories [15] according to different overlapping patterns of triples: Normal, SEO, and EPO. The statistics of the two datasets are shown in Table 1.

Table 1. Statistics of NYT and WebNLG.

Dataset	NYT		WebNLG	
	Train	Test	Train	Test
Normal	37,013	3266	1596	246
SEO	14,735	1297	3406	457
EPO	9782	978	227	26
ALL	56,195	5000	5019	703

Evaluation We utilized the standard micro-F1-scores to evaluate the proposed model on both NYT and WebNLG. The predicted triple is regarded as correct if the relation type and two entities are all correct. Note that the entity is correct if the last words of the entity are correct because both datasets only annotated the last word of every entity. Therefore, we only performed partial matching evaluations.

Implementation Details We implemented our TRN with the Pytorch framework. For a fair comparison with previous works [12,23], we used BERT_{base_cased} in our experiments as the encoder and set the max length of a sentence to 100. The learning rate of BERT was set to 10^{-5}, and the attention layers and triple reasoning module were all set to 2×10^{-5}. The number of entity pairs n was set to 10, and the number of implicit entity pairs k was set to 5. We used Byte Pair Encoding [40] to segment the input sentences. The hidden dimension d was set to 768. We set 3 layers in the entity pair encoding module. The dropout [41] of the attention layer was set to 0.1. We used AdamW [42] to update TRN to minimize the loss function defined in Section 3.5 and set the weight decay as 0.02. We trained the designed model for 200 epochs with the batch size as 32/16 on NYT/WebNLG.

4.2. Comparative Models

We compared TRN with several previous models:

- NovelTagging [8] proposes a novel tagging scheme to transform the task into a sequence labeling problem.
- CopyRE [15] proposes an end-to-end based on sequence-to-sequence learning with a copy mechanism, which jointly extracts triples as the sequence form.
- CopyRE$_{RL}$ [16] considers the order of the relational facts extracted from the sentence to be also important. They utilized reinforcement learning (RL) in the sequence-to-sequence model to learn the extraction order.
- TME [10] proposes a joint model that is capable of discovering multiple triples in a sentence via ranking with a translation mechanism. They used a tri-part tagging scheme (position, type, entity) to obtain entity features.
- ETL-Span [11] proposes a decomposition-based tagging scheme. They decomposed the joint extraction task into HE extraction and TER extraction, which are interrelated subtasks. Their model can not only capture the semantic interdependency between different steps, but also reduces noise from irrelevant entity pairs.
- CopyMTL [17] proposes a Multi-Task Learning (MTL) framework equipped with a copy mechanism to predict the multi-token entities and accurately differentiate head and tail entities.
- WDec [18] proposes two approaches to use the sequence-to-sequence architecture to jointly extract entities and relations.
- CasRel [12] proposes a novel cascade binary tagging framework. They treated relations as functions that map subjects to objects in a sentence to solve the overlapping problem.
- PMEI [43] designs a multitask learning architecture to extract relevant features that correlate with entity recognition and relation extraction from the input. They controlled the injection of early predictions to ensure that the extracted task-specific representations were good for classification.
- TPLinker [23] proposes a novel handshaking tagging scheme to perform the one-stage extraction. TPLinker is capable of discovering overlapping relations sharing one or both entities to address the exposure bias problem.
- RMAN [14] proposed a multi-feature fusion sentence representation and decoder sequence annotation to deal with overlapping triples.
- CGT$_{UniLM}$ [20] proposes a model, contrastive triple extraction with a generative transformer. They introduced a single shared transformer module for encoder–decoder-based generation.
- RIFRE [13] designs a tagging framework, which first extracts all the candidate head entities and then labels the tail entities with corresponding relations.

4.3. Performance

4.3.1. Main Results

From Table 2, we can observe the results of these baseline models and TRN: our proposed model TRN significantly outperforms all previous baseline models on NYT and WebNLG. Concretely, the proposed model achieves 0.6% and 0.5% absolute F1-score improvements over the previous method RIFRE [13] on NYT and WebNLG, respectively. TRN−TRm outperforms all the baseline models even without using the triple reasoning module. This demonstrates that our basic architecture is more effective in the relational triples' extraction task. In addition, for the model TRN−TRm without using the triple reasoning module, we can observe that there are 0.4% and 0.3% F1-score improvementscompared with TRN on NYT and WebNLG, respectively. This proves the triple reasoning module can truly extract additional relational triples.

Compared to TME [10], capturing inner connections, TRN performs better on NYT. This demonstrates that TRN has a strong capability of identifying implicit triples.

We can also observe from Table 2 that there is a gap between the performance on NYT and WebNLG for most baseline models. In detail, from Table 1, we can find that the ratio

of overlapping triples in WebNLG is larger than the ratio in NYT. The imbalanced data distribution leads to worse performance on WebNLG since most baselines are weak at dealing with overlapping triples. We also find that [23] obtains a fair performance on the two benchmarks, which benefits from their handshaking architecture.

Table 2. The performance of our proposed TRN and previous models on the NYT and WebNLG test sets. TRN−TRm indicates the model does not use the triples' reasoning module presented in Section 3.3. The best scores are in bold font.

Models	NYT			WebNLG		
	Precision	Recall	F1	Precision	Recall	F1
NovelTagging [8]	62.4	31.7	42.0	52.5	19.3	28.3
CopyRE [15]	61.0	56.6	58.7	37.7	36.4	37.1
CopyRE$_{RL}$ [16]	77.9	67.2	72.1	63.3	59.9	61.6
TME [10]	69.6	47.8	56.7	-	-	-
ETL-Span [11]	84.9	72.3	78.1	84.0	91.5	87.6
CopyMTL [17]	75.7	68.7	72.0	58.0	54.9	56.4
WDec [18]	94.5	76.2	84.4	-	-	-
CasRel [12]	89.7	89.5	89.6	93.4	90.1	91.8
PMEI [43]	90.5	89.8	90.1	91.0	**92.9**	92.0
TPLinker [23]	91.3	**92.5**	91.9	91.8	92.0	91.9
RMAN [14]	87.1	83.8	85.4	83.6	85.3	84.5
CGT [20]	**94.7**	84.2	89.1	92.9	75.6	83.4
RIFRE [13]	93.6	90.5	92.0	93.3	92.0	92.6
TRN (Our method)	93.0	92.3	**92.6**	**93.5**	92.7	**93.1**
TRN−TRm	92.4	92.1	92.2	93.4	92.4	92.8

4.3.2. Detailed Results on Different Types of Sentences

We compared the ability of the models in extracting relational facts from sentences with different numbers of triples and different overlapping patterns. The detailed results are shown in Table 3. The results indicate that the designed model can not only extract multiple relational triples, but also handles the overlapping problem, which is popular in this task.

Table 3. The F1-sores on sentences with different numbers of triples. N is the number of triples in a sentence. The best scores are in bold, while the second-best scores are underlined.

Models	NYT								WebNLG							
	Normal	SEO	EPO	N = 1	N = 2	N = 3	N = 4	N $\geq$ 5	Normal	SEO	EPO	N = 1	N = 2	N = 3	N = 4	N $\geq$ 5
CopyRE	66.0	48.6	55.0	67.1	58.6	52.0	53.6	30.0	59.2	33.0	36.6	59.2	42.5	31.7	24.2	30.0
CopyRE$_{RL}$	71.2	69.4	72.8	71.7	72.6	72.5	77.9	45.9	65.4	60.1	67.4	63.4	62.2	64.4	57.2	55.7
CasRel	87.3	91.4	92.0	88.2	90.3	91.9	94.2	83.7	89.4	92.2	94.7	89.3	90.8	94.2	92.4	90.9
TPLinker	90.1	93.4	94.0	90.0	92.9	93.1	96.1	90.0	87.9	92.5	**95.3**	88.0	90.1	94.6	93.3	91.6
RIFRE	**90.7**	93.2	93.5	**90.7**	92.8	93.4	94.8	89.6	90.1	93.1	94.7	**90.2**	**92.0**	**94.8**	93.0	92.0
TRN	90.5	**94.6**	**94.6**	90.6	**93.5**	**94.1**	**96.4**	**91.9**	**90.4**	**93.6**	90.3	90.0	91.7	94.5	**94.8**	**94.0**

From Table 3, we divided the sentences into eight sub-classes for each test set. We observed that the designed model achieves 10 best scores and 4 second-best scores on the different numbers of triples over the two benchmarks. Our proposed model achieves 1.9% and 2.0% improvements on NYT and WebNLG over the second-best scores when $N \geq 5$. The results indicate that the designed model is more effective in handling difficult scenarios.

From the overlapping patterns perspective, we divided the sentences into three sub-classes (Normal, SEO, EPO). Our proposed model TRN also achieves the best scores in most cases compared to the previous baselines. There are 1.2% and 0.5% F1-score improvements compared with the previous best scores in SEO sentences on NYT and WebNLG, respectively, and 0.6% in EPO sentences on NYT. The results indicate that the proposed model is

more suitable for handling complicated scenarios. We found that the implicit triples always overlapped with others in a sentence. Therefore, we can infer that the improvements of extracting overlapping triples demonstrate that the designed model can truly extract more implicit triples.

We found that the EPO performance of TRN on WebNLG was worse than TPLinker from Table 4. To explore the reasons, we conducted the detailed experiment only on EPO sentences. The gap of the recall indicates that the proposed model cannot reason more complete triples than TPLinker. We speculated that one reason is that TRN needs much more EPO data to fit and evaluate. We found that all previous work classified overlapping pattern A→B and B→A (A, B are two different entities; → is the relation between them) into SEO rather than the EPO defined by [15], which means implicit triples mostly come from SEO sentences. Thus, EPO sentences only contain A→B and A→B, for which we can only reason implicit triples in a multi-labeled pattern. The designed model mainly reasons the implicit triples via SEO sentences. Therefore, the single triple reasoning, i.e., A→B reasons multi-labeled A→B categorized as EPO very few times.

Table 4. The EPO performance of TRN and TPLinker on the WebNLG test set.

Model	Precision	Recall	F1
TPLinker	97.6	94.3	96.0
TRN	95.5	85.7	90.3

4.4. Ablation Study

Results on Different Model Settings: We conducted ablation studies to prove the inferential effectiveness on our subset (We collected triple combinations in a sentence, such as A→B, B→C, coupled with A→C or A→B and A→C, coupled with B→C or A→B, coupled with B→A combinations. A, B and C are three different entities.), which were manually selected from NYT. We selected the 100 overlapping pattern sentences, which are rich in implicit triples. Then, we conducted our experiments on the previous models and ours with different ablation settings:

- TRN−TRm removes the whole triple reasoning module.
- TRN−TRm-Seq removes the triple reasoning module, adds a causal mask to the entity pair encoding module to predict the triples order by order (does not distinguish the difference between normal triples and implicit triples), and uses NLL loss as the training objective.

We compared the performance of different settings on our subset. The results are illustrated in Table 5. Compared with these different models, we can observe that: (1) TRN brings a significant improvement over TRN−TRm on the subset. This indicates that our triple reasoning module can capture the information among triples and construct the implicit connections between entities. (2) TRN−TRm also brings improvements over TRN−TRm-Seq. We considered that it benefits from the training strategy. The bipartite training strategy successfully matches the normal triples and implicit triples with the ground truth at the training stage. (3) TRN−TRm-Seq is close to TPLinker and RIFRE on our subset. Note that the sequence-to-sequence architecture brings improvements over the generative model CGT. We considered that it benefits from our entity pair encoding module. (4) TRN−TRm also outperforms the previous models. Even without using the triple reasoning module, the proposed model obtains a competitive result with respect to the previous methods. This indicates that using our basic structure is effective.

Table 5. Performance of different models on our subset.

Model	F1 Score
CGT	79.1
TPLinker	84.9
RIFRE	85.3
TRN−TRm-Seq	82.2
TRN−TRm	87.6
TRN	91.4

Parameter Analysis: We trained different models and evaluated them on our subset to analyze how the k number of implicit triples affects model performance. From Figure 5, we can observe that when $k = 0$ and $n \leq 4$, the F1-score presents a tendency to increase. This indicates that the number of normal triples is still insufficient to extract from a raw sentence. There is no obvious increase when $n \geq 4$, which means our basic architecture cannot extract more triples. When $k = 5$, we can observe that the F1-scores increase slightly as the number of normal triples progressively increases. This means more implicit triples will be extracted as normal triples increase. It seems the F1-score will converge when $n \geq 5$.

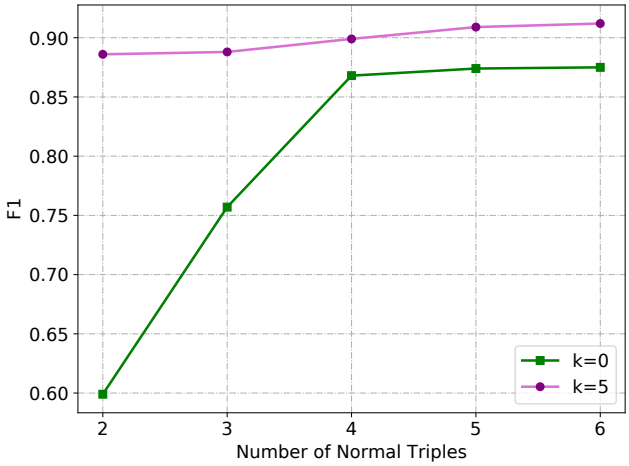

Figure 5. Results on different numbers of implicit triples. The green line indicates not using the triple reasoning module. The purple line indicates there are 5 implicit triples at most in the raw sentence.

4.5. Case Study

Figure 6 shows the three examples of TRN and TPLinker on our subset. The first example shows an implicit connection (Braga, country, Portugal) by the reversed relation pattern (Portugal, contains, Braga). In the second example, we infer (Edmund, lived, Louisiana) by the multi-hop relation patterns (Edmund, lived, Crowley) and (Louisiana, contains, Crowley). The third example shows (China, contains, Jinghong) in transitive relation patterns (China, contains, Yunnan) and (Yunnan, contains, Jinghong). We also observe that the proposed model can recognize new relations from the orange arrows, although the connections do not givegolden triples.

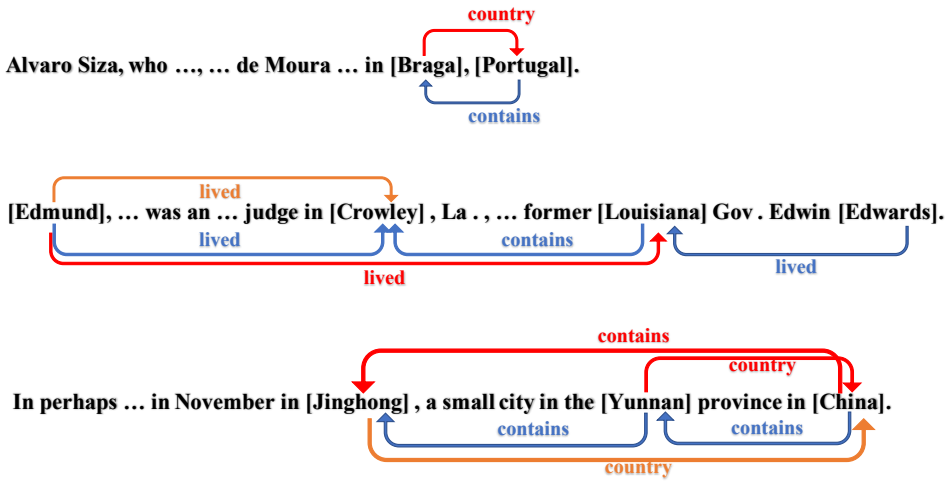

Figure 6. Three examples with implicit triples. Entities are in square brackets. Blue arrows indicate relational connections predicted by TPLinker, as well as ours. The orange arrows indicate that relational connections do not appear in golden triples. The red arrows indicate implicit connections only inferred by ours.

4.6. Error Analysis

To validate the effectiveness of TRN in handling implicit triples, we conducted a supplemental experiment to explore the factors that affect our proposed model. From Table 6, we observe that there is a trivial gap between ($E1$, $E2$) and ($E1$, R, $E2$) for NYT both on EPO and SEO sentences. This implies identifying entity pairs is the bottleneck for TRN. Furthermore, the recall score is relatively lower than the precision, which is the main factor affecting F1-score both on EPO and SEO. This reveals that TRN needs to find more entities that may have connections with each other even if we have already discovered more entity pairs. For WebNLG, we observe that identifying relations comprise the bottleneck on EPO sentences, while entity pairs are for SEO sentences. Note that there is a huge gap between the precision and recall of both ($E1$, $E2$) and R on EPO sentences. We discussed the reason why the designed model performed worse in Section 4.3. We also found that there is an obvious gap between ($E1$, $E2$) and ($E1$, R, $E2$) on SEO sentences and R and ($E1$, R, $E2$) on EPO sentences. This demonstrates that there is a mismatch between entity pairs and relations. Compared with NYT, we speculated the difference to be that the number of relations in WebNLG is larger than NYT. Therefore, identifying triples is harder for TRN on WebNLG than NYT. To further explore the factors affecting TRN, we classified several error sentences and conducted an analysis. From Table 7:

Wrong Arguments: As Sentence #1 shows, the model incorrectly recognizes an entity mention "Kandahar" as the tail, which is not annotated in the sentence. We argue that there is no evident clue to demonstrate the relation between "Pakistan" and "Kandahar" in context. The wrong prediction will be prohibited if the model is equipped with common sense or real-world knowledge (e.g., Kandahar is a city in Afghanistan; Pakistan is a country near Afghanistan).

No Relation: There is an evident clue in Sentence #2 to declare that "East China Normal University" is in "Shanghai", but the model incorrectly predicts "no_relation" in our case. We argue that the model normally focuses more on implicit triple reasoning if there are more overlapping pattern triples in a sentence. Thus, it neglects the extraction of normal triples. In particular, such "no_relation" wrong cases are rich in predictions. We can also find in Table 6 that the recall score is much lower than the precision score in any type of sentence.

Arguments Mismatch: We can observe from Sentence #3 that both entities "Immelt" and "York" are annotated in the sentence, but they are not related. We argue that the reason is the same as "no_relation". It seems balancing the extraction of implicit triples and normal triples is a problem for the proposed model.

No Label: The designed model in Sentence #4 generated the correct triple, but it did not exist in the ground truth label. Generally, this situation also occurs, and it is treated as a wrong case.

Table 6. Results on triple elements. The triple (E1, R, E2) is divided into (E1, E2) and R, where E1 represents the head entity, E2 represents the tail entity, and R represents the relation between E1, E2.

Dataset	Element	EPO			SEO			Overall		
		Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
NYT	(E1, E2)	96.3	93.3	94.8	97.1	92.5	94.7	93.7	93.0	93.4
	R	98.4	95.4	96.9	99.4	94.8	97.0	96.9	96.2	96.6
	(E1, R, E2)	96.1	93.1	94.6	96.9	92.4	94.6	93.0	92.3	92.6
WebNLG	(E1, E2)	97.7	87.8	92.5	96.4	94.9	95.7	95.8	94.9	95.3
	R	96.6	86.7	91.4	96.9	95.5	96.2	96.1	95.2	95.6
	(E1, R, E2)	95.5	85.7	90.3	94.3	92.9	93.6	93.5	92.7	93.1

Table 7. Error analysis on different cases.

	Instances
	Sentence #1: He said he had brought in one bomber called Imran from Pakistan's North - West Frontier Province, who blew himself up on the road near the Kandahar airport. Gold: (Pakistan, administrative_divisions, Province), (Province, country, Pakistan), (Pakistan, contains, Province) Pred: (Pakistan, administrative_divisions, Province), (Province, country, Pakistan), (Pakistan, contains, Province), (Pakistan, contains, Kandahar)
	Sentence #2: The vision for this sort of thing has existed in China for a very long time," said Wu Yongyi, deputy dean of the International College of Chinese Studies at East China Normal University in Shanghai, who has been involved in overseas language instruction missions since the 1980's. Gold: (Shanghai, country, China), (China, contains, Shanghai), (China, administrative_divisions, Shanghai), (Shanghai, contains, University) Pred: (Shanghai, country, China), (China, contains, Shanghai), (China, administrative_divisions, Shanghai), (Shanghai, no_relation, University)
	Sentence #3: Jeffrey R. Immelt, chairman and chief executive of General Electric, bounced a $2000 check to the failed New York gubernatorial campaign of William F. Weld, according to a campaign finance filing released Monday. Gold: (Weld, place_lived, York), (Immelt, company, Electric), (Immelt, major_shareholder_of, Electric), (Electric, major_shareholders, Immelt) Pred: (Immelt, place_lived, York), (Immelt, company, Electric), (Immelt, major_shareholder_of, Electric), (Electric, major_shareholders, Immelt)
	Sentence #4: It is also adding flights on existing routes to several cities, including Kiev and Odessa in Ukraine, and Dubrovnik and Split in Croatia. Gold: (Kiev, country, Ukraine), (Ukraine, capital, Kiev), (Ukraine, administrative_divisions, Kiev), (Croatia, contains, Dubrovnik), (Ukraine, contains, Odessa), (Ukraine, contains, Kiev) Pred: (Kiev, country, Ukraine), (Ukraine, capital, Kiev), (Ukraine, administrative_divisions, Kiev), (Croatia, contains, Dubrovnik), (Ukraine, contains, Odessa), (Croatia, contains, Split), (Ukraine, contains, Kiev)

5. Conclusions

In this paper, we proposed a novel framework called TRN to jointly extract triples, especially implicit triples. Compared with previous methods, our method is capable of reasoning the implicit triples even if context words are not explicitly expressed in the relation between two entities. Our model will extract more triples that are overlapped

with others from the sentence, and we utilized an attention-based entity pair encoding module to extract entity pairs directly. To establish implicit connections among these extracted entities in triples, we designed a triple reasoning module to select relevant triple pairs. We matched the training labels with the corresponding triples using the bipartite method to optimize the matching training objective. Experiment results on two benchmarks validated the effectiveness of our method over different scenarios. Our proposed model also outperformed the most advanced baseline, especially for implicit triples' extraction.

Author Contributions: Z.W.: conceptualization, methodology, formal analysis, software, writing—original draft. L.Y.: supervision, writing—review and editing. J.Y.: supervision, writing—review and editing. T.L.: supervision, writing—review and editing. L.H.: supervision, writing—review and editing. Z.L.: supervision, writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (Grant Nos. U1636211, 61672081, 61370126), the 2020 Tencent Wechat Rhino-Bird Focused Research Program, and the Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2021ZX-18).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: MDPI Research Data Policies at https://github.com/weizhepei/CasRel, 15 September 2020.

Acknowledgments: This work was supported in part by the National Natural Science Foundation of China (Grant Nos. U1636211, 61672081, 61370126), the 2020 Tencent Wechat Rhino-Bird Focused Research Program, and the Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2021ZX-18), the National Key R&D Program of China (No.2020YFB1006102), and the Key Areas R&D Program of Guangdong Province (No.2019B010137003).

Conflicts of Interest: The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Zelenko, D.; Aone, C.; Richardella, A. Kernel methods for relation extraction. *J. Mach. Learn. Res.* **2003**, *3*, 1083–1106.
2. Chan, Y.S.; Roth, D. Exploiting syntactico-semantic structures for relation extraction. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 551–560.
3. Yu, X.; Lam, W. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Beijing, China, 23–27 August 2010;* Coling 2010 Organizing Committee: Beijing, China, 2010; pp. 1399–1407.
4. Li, Q.; Ji, H. Incremental joint extraction of entity mentions and relations. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistic, Baltimore, MD, USA, 22–27 June 2014; Volume 1, pp. 402–412.
5. 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.
6. 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.
7. 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, Berlin, Germany, 7–12 August 2016; Volume 1, pp. 1105–1116.
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, ACL 2017, Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1, pp. 1227–1236.
9. Dai, D.; Xiao, X.; Lyu, Y.; Dou, S.; She, Q.; Wang, H. Joint extraction of entities and overlapping relations using position-attentive sequence labeling. *Proc. AAAI Conf. Artif. Intell.* **2019**, *33*, 6300–6308. [CrossRef]
10. Tan, Z.; Zhao, X.; Wang, W.; Xiao, W. Jointly Extracting Multiple Triplets with Multilayer Translation Constraints. In Proceedings of the The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, HI, USA, 27 January–1 February 2019; AAAI Press: Menlo Park, Calif., USA, 2019; pp. 7080–7087.

11. Yu, B.; Zhang, Z.; Shu, X.; Liu, T.; Wang, Y.; Wang, B.; Li, S. Joint Extraction of Entities and Relations Based on a Novel Decomposition Strategy. In Proceedings of the ECAI 2020—24th European Conference on Artificial Intelligence, including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020), Santiago de Compostela, Spain, 29 August–8 September 2020; Volume 325, pp. 2282–2289.
12. 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, pp. 1476–1488.
13. Zhao, K.; Xu, H.; Cheng, Y.; Li, X.; Gao, K. Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. *Knowl. Based Syst.* **2021**, *219*, 106888. [CrossRef]
14. Lai, T.; Cheng, L.; Wang, D.; Ye, H.; Zhang, W. RMAN: Relational multi-head attention neural network for joint extraction of entities and relations. *Appl. Intell.* **2022**, *52*, 3132–3142. [CrossRef]
15. Zeng, X.; Zeng, D.; He, S.; Liu, K.; Zhao, J. Extracting relational facts by an end-to-end neural model with copy mechanism. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; Volume 1, pp. 506–514.
16. Zeng, X.; He, S.; Zeng, D.; Liu, K.; Liu, S.; Zhao, J. Learning the extraction order of multiple relational facts in a sentence with reinforcement learning. 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. 367–377.
17. Zeng, D.; Zhang, H.; Liu, Q. Copymtl: Copy mechanism for joint extraction of entities and relations with multi-task learning. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 9507–9514. [CrossRef]
18. Nayak, T.; Ng, H.T. Effective modeling of encoder–decoder architecture for joint entity and relation extraction. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 8528–8535. [CrossRef]
19. Sui, D.; Chen, Y.; Liu, K.; Zhao, J.; Zeng, X.; Liu, S. Joint Entity and Relation Extraction with Set Prediction Networks. *arXiv* **2020**, arXiv:2011.01675.
20. Ye, H.; Zhang, N.; Deng, S.; Chen, M.; Tan, C.; Huang, F.; Chen, H. Contrastive Triple Extraction with Generative Transformer. In Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI, Virtually, 2–9 February 2021; pp. 14257–14265.
21. 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.
22. 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, 9–11 September 2017; pp. 1730–1740.
23. 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, 8–13 December 2020; pp. 1572–1582.
24. Yan, Z.; Zhang, C.; Fu, J.; Zhang, Q.; Wei, Z. A Partition Filter Network for Joint Entity and Relation Extraction. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event/Punta Cana, Dominican Republic, 7–11 November, 2021; pp. 185–197.
25. Zhu, H.; Lin, Y.; Liu, Z.; Fu, J.; Chua, T.S.; Sun, M. Graph Neural Networks with Generated Parameters for Relation Extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019, pp. 1331–1339.
26. Angeli, G.; Manning, C.D. Philosophers are mortal: Inferring the truth of unseen facts. In Proceedings of the Seventeenth Conference on Computational Natural Language Learning, Sofia, Bulgaria, 8–9 August 2013; pp. 133–142.
27. Jia, N.; Cheng, X.; Su, S. Improving knowledge graph embedding using locally and globally attentive relation paths. *Adv. Inf. Retr.* **2020**, *12035*, 17–32.
28. Liang, Z.; Yang, J.; Liu, H.; Huang, K.; Cui, L.; Qu, L.; Li, X. HRER: A New Bottom-Up Rule Learning for Knowledge Graph Completion. *Electronics* **2022**, *11*, 908. [CrossRef]
29. Peng, G.; Chen, X. Entity–Relation Extraction—A Novel and Lightweight Method Based on a Gate Linear Mechanism. *Electronics* **2020**, *9*, 1637. [CrossRef]
30. Auer, S.; Bizer, C.; Kobilarov, G.; Lehmann, J.; Cyganiak, R.; Ives, Z. Dbpedia: A nucleus for a web of open data. In *The Semantic Web*; Springer: Manhattan, New York City, USA, 2007; pp. 722–735.
31. Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; Taylor, J. Freebase: A collaboratively created graph database for structuring human knowledge. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, Vancouver, BC, Canada, 10–12 June 2008; pp. 1247–1250.
32. Dong, X.; Gabrilovich, E.; Heitz, G.; Horn, W.; Lao, N.; Murphy, K.; Strohmann, T.; Sun, S.; Zhang, W. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 601–610.
33. 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, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, pp. 4171–4186.

34. Gu, J.; Bradbury, J.; Xiong, C.; Li, V.O.K.; Socher, R. Non-Autoregressive Neural Machine Translation. In Proceedings of the 6th International Conference on Learning Representations, ICLR 2018, Conference Track Proceedings, Vancouver, BC, Canada, 30 April–3 May 2018.
35. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
36. Bahdanau, D.; Cho, K.; Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings, San Diego, CA, USA, 7–9 May 2015.
37. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
38. Riedel, S.; Yao, L.; McCallum, A. Modeling relations and their mentions without labeled text. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Würzburg, Germany, 16–20 September 2010; pp. 148–163.
39. Gardent, C.; Shimorina, A.; Narayan, S.; Perez-Beltrachini, L. Creating Training Corpora for NLG Micro-Planners. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1, pp. 179–188.
40. Sennrich, R.; Haddow, B.; Birch, A. Neural Machine Translation of Rare Words with Subword Units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Berlin, Germany, 7–12 August 2016; Volume 1.
41. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
42. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019.
43. Sun, K.; Zhang, R.; Mensah, S.; Mao, Y.; Liu, X. Progressive multitask learning with controlled information flow for joint entity and relation extraction. *Assoc. Adv. Artif. Intell.* **2021**, *35*, 13851–13859.