

Jointly Extracting Explicit and Implicit Relational Triples with Reasoning Pattern Enhanced Binary Pointer Network

Yubo Chen[†], Yunqi Zhang[†], Changran Hu[‡], Yongfeng Huang[†]

[†]Department of Electronic Engineering & BNRist, Tsinghua University, Beijing, China

[‡]University of California, Berkeley, USA

ybch14@gmail.com zhang-yq15@outlook.com

changran_hu@berkeley.edu yfhuang@tsinghua.edu.cn

Abstract

Relational triple extraction is a crucial task for knowledge graph construction. Existing methods mainly focused on explicit relational triples that are directly expressed, but usually suffer from ignoring implicit triples that lack explicit expressions. This will lead to serious incompleteness of the constructed knowledge graphs. Fortunately, other triples in the sentence provide supplementary information for discovering entity pairs that may have implicit relations. Also, the relation types between the implicitly connected entity pairs can be identified with relational reasoning patterns in the real world. In this paper, we propose a unified framework to jointly extract explicit and implicit relational triples. To explore entity pairs that may be implicitly connected by relations, we propose a binary pointer network to extract overlapping relational triples relevant to each word sequentially and retain the information of previously extracted triples in an external memory. To infer the relation types of implicit relational triples, we propose to introduce real-world relational reasoning patterns in our model and capture these patterns with a relation network. We conduct experiments on several benchmark datasets, and the results prove the validity of our method.

1 Introduction

Relational triple extraction is defined as automatically recognizing semantic relations with triple structures (*subject, relation, object*) among multiple entities in a sentence. It is a critical task for constructing Knowledge Graphs (KGs) from unlabeled corpus (Dong et al., 2014).

Early work of relational triple extraction applied pipeline methods (Zelenko et al., 2003; Chan and Roth, 2011), which ran entity recognition and relation classification separately. However, such pipeline approaches suffered from error propagation. To address this issue, recent work proposed to jointly extract entity and relations from the text

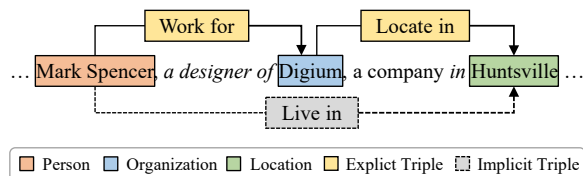


Figure 1: An example of explicit and implicit relational triples. Italic phrases are key relational expressions corresponding to the explicit relational triples.

with feature-based methods (Yu and Lam, 2010; Li and Ji, 2014; Ren et al., 2017). Afterward, neural network-based models were proposed to eliminate hand-crafted features (Gupta et al., 2016; Zheng et al., 2017). More recently, several methods were proposed to extract *overlapping* triples, such as tagging-based (Dai et al., 2019; Wei et al., 2020), graph-based (Wang et al., 2018; Fu et al., 2019), copy-based (Zeng et al., 2018, 2019, 2020) and token pair linking models (Wang et al., 2020).

Existing models achieved considerable success on extracting *explicit* triples which have direct relational expressions in the sentence. However, there are many *implicit* relational triples that are not explicitly expressed. For example, in Figure 1, the explicit triples are strongly indicated by the key relational phrases, but the implicit relation “*Live in*” is not expressed explicitly. Unfortunately, existing methods usually ignored implicit triples (Zhu et al., 2019), which will cause serious incompleteness of the constructed KGs and performance degradation of downstream tasks (Angeli and Manning, 2013; Jia et al., 2020; Jun et al., 2020).

Our work is motivated by several observations. First, other relational triples within a sentence provide supplementary information for discovering entity pairs that may have implicit relational connections. For example, in Figure 1, the explicit triples establish a relational connection between “*Mark Spencer*” and “*Huntsville*” through the intermediate entity “*Digium*”. Second, the relation

types of implicit relation triples can be derived through real-world reasoning patterns. For example, in Figure 1, the reasoning pattern “one *lives* where the company he *works* for is *located*” helps identify the type of the implicit triple as “*Live in*”.

In this paper, we propose a unified framework for the joint extraction of explicit and implicit relational triples. We propose a Binary Pointer Network (BPtrNet), which is based on the pointer network (Vinyals et al., 2015), to extract overlapping relational triples relevant to each word sequentially. To discover implicitly connected entity pairs, we preserve the information of previously extracted triples in an external memory and use it to enhance the extraction of later time steps. To infer the relation types between the implicitly connected entity pairs, we propose to augment our model with real-world relational reasoning patterns and capture the relational inference logic with a Relation Network (RN) (Santoro et al., 2017). The RN obtains a pattern-enhanced representation from the memory for each word pair. Then the Reasoning pattern enhanced BPtrNet (R-BPtrNet) uses the word pair representation to compute a binary score for each candidate triple. Finally, triples with positive scores are output as the extraction result.

The main contributions of this paper are:

- We propose a unified framework to jointly extract explicit and implicit relational triples.
- To discover entity pairs that are implicitly connected by relations, we propose a BPtrNet model to extract overlapping relational triples sequentially and utilize an internal memory to retain the extracted triples.
- To enhance the relation type inference of implicitly connected entity pairs, we propose to introduce relational reasoning patterns, captured with a RN, to augment our model.
- We conduct experiments on several benchmark datasets and the experimental results demonstrate the validity of our method.

2 Related Work

Early work of relational triple extraction addressed this task in a pipelined manner (Zelenko et al., 2003; Zhou et al., 2005; Chan and Roth, 2011; Gormley et al., 2015). They first ran named entity recognition to identify all entities and then classified relations between all entity pairs. However, these pipelined methods usually suffered from error propagation problem and failed to capture the

interactions between entities and relations.

To overcome these drawbacks, recent research focused on jointly extracting entities and relations, including feature-based models (Yu and Lam, 2010; Li and Ji, 2014; Ren et al., 2017) and neural network-based models (Gupta et al., 2016; Miwa and Bansal, 2016; Zheng et al., 2017). For example, Ren et al. (2017) proposed to jointly embed entities, relations, text features and type labels into two low-dimensional spaces. Miwa and Bansal (2016) proposed a joint model containing two long-short term memories (LSTMs) (Gers et al., 2000) with shared parameters. Zheng et al. (2017) proposed to extract relational triples directly by transforming this task into a sequence tagging problem, whose tags contain the information of entities and the relations they hold. However, they only assigned one label for each word, which means that this method failed to extract overlapping triples. Subsequent work proposed several mechanisms to solve this problem: (1) labeling tagging sequences for words (Dai et al., 2019) or entities (Yu et al., 2019; Wei et al., 2020); (2) transforming the sentence into a graph structure (Wang et al., 2018; Fu et al., 2019); (3) generating triple element sequences with copy mechanism (Zeng et al., 2018, 2019, 2020; Nayak and Ng, 2020); (4) linking token pairs with a handshake tagging scheme (Wang et al., 2020). However, these methods usually ignored implicit relational triples that are not directly expressed in the sentence (Zhu et al., 2019), thus will lead to the incompleteness of the resulting KGs and negatively affect the performance of downstream tasks (Angeli and Manning, 2013; Jia et al., 2020).

Our work is motivated by two observations. First, other triples in the sentence provide supplementary evidence for discovering entity pairs with implicit relational connections. Second, the relation types of the implicit connections need to be identified through real-world reasoning patterns.

In this paper, we propose a unified framework for the joint extraction of explicit and implicit relational triples. We propose a binary pointer network to sequentially extract overlapping relational triples and externally keep the information of predicted triples for exploring implicitly connected entity pairs. We also propose to introduce real-world reasoning patterns in our model to help derive the relation type of implicit triples with a relation network. Experimental results on several benchmark datasets demonstrate the effectiveness of our method.

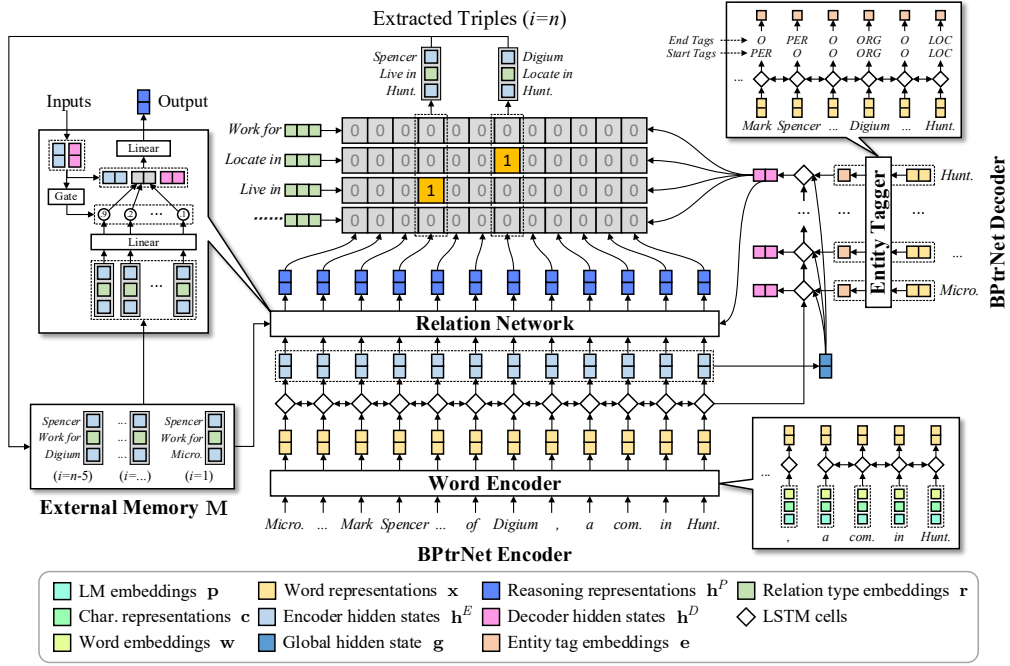


Figure 2: The overall framework of our approach.

3 Our Approach

The overall framework of our approach is shown in Figure 2. We introduce the Binary Pointer Network (BPTrNet) and the Relation Network (RN) in Section 3.1 and 3.2 and the details of training and inference in Section 3.3, respectively.

3.1 Binary Pointer Network

Existing methods usually failed to extract implicit relational triples due to the lack of explicit expressions (Zhu et al., 2019). Fortunately, we observe that other triples in the sentence can help discover entity pairs that may have implicit relational connections. For instance, in the sentence “George is Judy’s father and David’s grandfather”, the relation between *Judy* and *David* is not explicitly expressed. In this case, if we first extract the explicit triple (*Judy*, *father*, *George*) and keep its information in our model, we can easily establish an implicit connection between *Judy* and *David* through *George* because *George* is explicitly connected with *David* by the relational keyword “grandfather”. Inspired by this observation, our model extracts relational triples relevant to each word sequentially and keeps all previous triples of this sentence to enhance the extraction at future time steps. This word-by-word extraction process can be regarded as transforming a text sequence into a sequence of extracting actions, which leads us to a

sequence-to-sequence (seq2seq) model.

Therefore, we propose a Binary Pointer Network (BPTrNet), based on a seq2seq pointer network (Vinyals et al., 2015), to jointly extract explicit and implicit relational triples. Our model first encodes the words of a sentence into vector representations (Section 3.1.1). Then, we use a binary decoder to sequentially transform the vectors into (overlapping) relational triples (Section 3.1.2). We also introduce an external memory to retain previously extracted triples for enhancing future decoding steps (Section 3.1.3).

3.1.1 Encoder

Given a sentence $[w_1, \dots, w_n]$, we first capture morphological patterns of entities with a convolutional neural network (CNN) (LeCun et al., 1989) and compute the character representation c_i of the word w_i ($i = 1, \dots, n$): $c_i = \text{CNN}(w_i; \theta) \in \mathbb{R}^{d_c}$. Then we introduce the context-sensitive representations $\mathbf{p}_{1:n}$ captured with a pre-trained Language Model (LM) to bring rich semantics and prior knowledge from the large-scale unlabeled corpus. We feed c_i , \mathbf{p}_i and the word embedding \mathbf{w}_i into a bidirectional LSTM (BiLSTM) to compute the contextualized word representations $\mathbf{x}_{1:n}$ and encode the sentence with another BiLSTM:

$$\begin{aligned} \mathbf{x}_i &= \text{BiLSTM}^{in}([\mathbf{w}_i; \mathbf{p}_i; \mathbf{c}_i]) \in \mathbb{R}^{d_{in}}, \\ \mathbf{h}_i^E &= \text{BiLSTM}^{Enc}(\mathbf{x}_i) \in \mathbb{R}^{2d_E}. \end{aligned} \quad (1)$$

3.1.2 Binary Decoder

First, to capture the interactions between entities and relations, we recognize the entities with a span-based entity tagger (Yu et al., 2019; Wei et al., 2020) and transform the tags into vectors as part of the decoder’s input (Figure 2). Specifically, we assign each token a start and end tag to indicate whether the current token corresponds to a start or end position of an entity of a certain type:

$$\begin{aligned} \mathbf{h}_i^T &= \text{BiLSTM}^{\text{Tag}}(\mathbf{x}_i) \in \mathbb{R}^{d_T} \\ p(\mathbf{y}_i^{s/e}) &= \text{softmax}(\mathbf{W}^{s/e} \mathbf{h}_i^T + \mathbf{b}^{s/e}) \\ \text{tag}_i^{s/e} &= \arg \max_k p(\mathbf{y}_i^{s/e} = k) \end{aligned} \quad (2)$$

where $(\mathbf{W}^s, \mathbf{b}^s)$ and $(\mathbf{W}^e, \mathbf{b}^e)$ are parameters of the start and end tag classifiers, respectively. Then we obtain the entity tag embedding $\mathbf{e}_i \in \mathbb{R}^{d_e}$ by averaging the look-up embeddings of the start and end tags. We also capture a global contextual embedding \mathbf{g} by max pooling over $\mathbf{h}_{1:n}^E$. Then we adopt a LSTM as the decoder ($\mathbf{h}_0^D = \mathbf{h}_n^E$):

$$\mathbf{h}_i^D = \text{LSTM}^{\text{Dec}}([\mathbf{x}_i; \mathbf{e}_i; \mathbf{g}], \mathbf{h}_{i-1}^D) \in \mathbb{R}^{2d_E}. \quad (3)$$

Next, we introduce how to extract relational triples at the i -th time step. We consider the current word as the object entity, select words as subjects that form triples with the object from all the words of the sentence, and predict the relation types between the subjects and the object. For example, in Figure 2, when the current object is “*Huntsville*”, the model selects “*Digium*” as the subject and classifies the relation as “*Locate in*”. Thus (“*Digium*”, “*Locate in*”, “*Huntsville*”) is extracted as a relational triple. Multi-token entities are represented with their last words and recovered by finding the nearest start tags of the same type from their last positions. However, the original softmax pointer in (Vinyals et al., 2015) only allows an object to point to **one** subject, thus fails to extract multiple triples with overlapping objects. To address this issue, we propose a binary pointer, which independently computes a binary score for each subject to form a relational triple with the current object under each relation type. Our method naturally solves the overlapping triple problem by producing multiple positive scores at one step (Figure 2). We formulate the score of the triple (w_j, r, w_i) as:

$$s_{ji}^{(r)} = \sigma(\mathbf{r}^\top \rho(\mathbf{W}^{ptr} [\mathbf{h}_j^E; \mathbf{h}_i^D] + \mathbf{b}^{ptr})), \quad (4)$$

and extract this candidate triple as a relational triple if $s_{ji}^{(r)}$ is higher than some threshold, such as 0.5

in our model ($i, j = 1, \dots, n$). σ and ρ are the sigmoid and tanh functions, respectively. $\mathbf{r} \in \mathbb{R}^{d_R}$ is the type embedding of the relation r . \mathbf{W}^{ptr} and \mathbf{b}^{ptr} are parameters of the binary pointer.

3.1.3 External Memory

We introduce an external memory \mathbf{M} to keep the previously extracted triples of the sentence. We first initialize \mathbf{M} as an empty set. After the decoder’s extraction process at the i -th time step, we represent the extracted triple $t = (w_{s_t}, r_t, w_i)$ as:

$$\mathbf{h}_t^M = [\mathbf{h}_{s_t}^E; \mathbf{r}_t; \mathbf{h}_i^E] \in \mathbb{R}^{d_M}. \quad (5)$$

Then we update the memory with the representations of the output triples t_1, \dots, t_{N_i} :

$$\mathbf{M} \leftarrow [\mathbf{M}; \mathbf{h}_{t_1}^M; \dots; \mathbf{h}_{t_{N_i}}^M] \in \mathbb{R}^{N \times d_M} \quad (6)$$

where N_i is the number of the currently extracted triples and $N = \sum_{k=1}^i N_k$. Note that we set and update the external memory for each sentence independently, and the memory stores only the triple representations of one single sentence. Thus triples of other sentences will not be introduced into the sentence currently being extracted. Finally, the triples in the memory are utilized to obtain the reasoning pattern-enhanced representations for future time steps, as described in Section 3.2.

3.2 Relation Network for capturing patterns of relational reasoning

Relation types of implicit relational triples are difficult to infer due to the lack of explicit evidence, thus need to be derived with real-world relational reasoning patterns. For example, in the sentence “*George is Judy’s father and David’s grandfather*”, the relation type between “*Judy*” and “*David*” can be inferred as “*father*” using the pattern “*father’s father is called grandfather*”.

Based on this fact, we propose to enhance our model by introducing real-world relational reasoning patterns. We capture the patterns with a Relation Network (RN) (Santoro et al., 2017), a neural network module specially designed for relational reasoning. A RN is essentially a composite function over a relational triple set \mathcal{T} : $\text{RN}(\mathcal{T}) = f_\phi(\{\mathbf{g}_\theta(t)\}_{t \in \mathcal{T}})$, where f_ϕ is an aggregation function and \mathbf{g}_θ projects a triple into a fixed-size embedding. We set the memory \mathbf{M} as the input relational triple set \mathcal{T} and utilize the RN to learn a pattern-enhanced representation \mathbf{h}_{ji}^P for the word pair (w_j, w_i) at the i -th time step. First, the \mathbf{g}_θ reads

the triple representations from \mathbf{M} and projects them with a fully-connected layer:

$$\mathbf{g}_\theta(t) = \rho(\mathbf{W}^\theta \mathbf{h}_t^M + \mathbf{b}^\theta) \in \mathbb{R}^{d_P}. \quad (7)$$

Then f_ϕ selects useful triples with a gating network¹: $u_t^\phi = \sigma(\mathbf{g}_\theta(t) \mathbf{U}^\phi [\mathbf{h}_j^E; \mathbf{h}_i^D]) \in \mathbb{R}$, and aggregates the selected triples with the word pair to compute \mathbf{h}_{ji}^P using another fully-connected layer:

$$\begin{aligned} \mathbf{h}_{ji}^P &= f_\phi(\{\mathbf{g}_\theta(t)\}_{t \in \mathbf{M}}) \\ &= \rho\left(\mathbf{W}^\phi \left[\mathbf{h}_j^E; \mathbf{h}_i^D; \sum_t u_t^\phi \mathbf{g}_\theta(t)\right] + \mathbf{b}^\phi\right). \end{aligned} \quad (8)$$

Finally, we modify Equation 4 as $s_{ji}^{(r)} = \sigma(\mathbf{r}^\top \mathbf{h}_{ji}^P)$ to compute the binary scores of candidate triples. We denote our Reasoning pattern enhanced BPtrNet model as R-BPtrNet. Note that we use quite simple formulas for f_ϕ and \mathbf{g}_θ because our contribution focuses on the effectiveness of introducing relational reasoning patterns for this task rather than the model structure. Exploration for more complex structures will be left for future work.

3.3 Training and Inference

We calculate the triple loss of a sentence as a binary cross entropy over valid candidate triples \mathcal{T}_v , whose subject and object are different entities (or the end words of different entities):

$$\mathcal{L}_t = -\frac{1}{|\mathcal{T}_v|} \left(\sum_{t \in \mathcal{T}_v} y_t \log s_t + (1 - y_t) \log(1 - s_t) \right) \quad (9)$$

where s_t is the score of the candidate triple t , $y_t = 1$ for gold triples and 0 for others. We also train the entity tagger with a cross-entropy loss:

$$\mathcal{L}_e = -\frac{1}{n} \sum_{i=1}^n \sum_{* \in \{s,e\}} \log p(y_i^* = \hat{y}_i^*) \quad (10)$$

where $\hat{y}_i^{s/e}$ are the gold start and end tags of the i -th word, respectively. Finally, we train the R-BPtrNet with the joint loss $\mathcal{L} = \mathcal{L}_t + \mathcal{L}_e$.

To prevent error propagation, we use the gold entity tags to filter out valid candidate triples and compute the tag embeddings $\mathbf{e}_{1:n}$ during training. We also update the memory \mathbf{M} with the gold relational triples. During inference, we extract triples from scratch and use the predicted entity tags and relational triples instead of the gold ones.

¹We don't use the more common attention mechanism (Bahdanau et al., 2015) to select triples because the attention weights are restricted to sum to 1. If all triples in the memory are useless, they will still be assigned a large weight due to the restriction, which will confuse the model.

4 Experiments

4.1 Datasets and Evaluation Metrics

We evaluate our method on two benchmark datasets. **NYT** (Riedel et al., 2010) consists of sentences from the New York Times corpus and contains 24 relation types. **WebNLG** (Gardent et al., 2017) was created for natural language generation task. It contains 171 relation types² and was adopted for relational triple extraction by (Zeng et al., 2018). We split the sentences into three categories: *Normal*, *SingleEntityOverlap (SPO)* and *EntityPairOverlap (EPO)* following Zeng et al. (2018). The statistics of the two datasets are shown in Table 1. Following previous work (Zeng et al., 2018; Wei et al., 2020; Wang et al., 2020), an extracted relational triple is regarded as correct only if the relation and the heads of both subject and object are all correct. We report the standard micro precision, recall, and F_1 -scores on both datasets.

Dataset	NYT		WebNLG	
	Train	Test	Train	Test
<i>Normal</i>	37013	3266	1596	246
<i>SEO</i>	9782	1297	227	457
<i>EPO</i>	14735	978	3406	26
ALL	56195	5000	5019	703

Table 1: Statistics of evaluation datasets.

4.2 Experimental Settings

We determine the hyper-parameters on the validation sets. We use the pre-trained GloVe (Pennington et al., 2014) embeddings as \mathbf{w} . We adopt a one-layer CNN with $d_c = 30$ channels to learn \mathbf{c} from 30-dimensional randomly-initialized character embeddings. We choose the state-of-the-art RoBERTa_{LARGE} (Liu et al., 2019) model³ as the pre-trained LM. For a fair comparison with previous methods, we also conduct experiments and report the scores with BERT_{BASE} (Devlin et al., 2019). We set d_{in} (Equation 1) as 300. The hidden dimensions of the encoder d_E and the entity tagger d_T are both 200. The dimensions of entity tag embeddings d_e and relation type embeddings d_R are set as 50 and 200, respectively. The projection dimension d_P of the RN is set as 500.

²As mentioned in (Wang et al., 2020), this number is miswritten as 246 in (Wei et al., 2020) and (Yu et al., 2019). Here we quote the correct number from (Wang et al., 2020).

³<https://github.com/huggingface/transformers>

Method	NYT			WebNLG		
	Prec.	Rec.	F_1	Prec.	Rec.	F_1
NovelTagging (Zheng et al., 2017)	62.4	31.7	42.0	52.5	19.3	28.3
CopyRE (Zeng et al., 2018)	72.8	69.4	71.1	60.9	61.1	61.0
CopyRE _{RL} (Zeng et al., 2019)	77.9	67.2	72.1	63.3	59.9	61.6
GraphRel (Fu et al., 2019)	63.9	60.0	61.9	44.7	41.1	42.9
ETL-Span (Yu et al., 2019)	84.9	72.3	78.1	84.0	91.5	87.6
CopyMTL (Zeng et al., 2020)	75.7	68.7	72.0	58.0	54.9	56.4
WDec (Nayak and Ng, 2020)	94.5	76.2	84.4	-	-	-
CGT _{UniLM} (Ye et al., 2020)	94.7	84.2	89.1	92.9	75.6	83.4
CASREL _{LSTM} (Wei et al., 2020)	84.2	83.0	83.6	86.9	80.6	83.7
CASREL _{BERT} (Wei et al., 2020)	89.7	89.5	89.6	93.4	90.1	91.7
TPLinker _{LSTM} (Wang et al., 2020)	83.8	83.4	83.6	90.8	90.3	90.5
TPLinker _{BERT} (Wang et al., 2020)	91.3	92.5	91.9	91.8	92.0	91.9
R-BPtrNet	90.9	91.3	91.1	90.7	94.6	92.6
R-BPtrNet _{BERT}	92.7	<u>92.5</u>	<u>92.6</u>	<u>93.7</u>	92.8	<u>93.3</u>
R-BPtrNet _{RoBERTa}	<u>94.0</u>	92.9	93.5	94.3	<u>93.3</u>	93.8

Table 2: Performance of our method and previous state-of-the-art models on the NYT and WebNLG test sets. The best scores are in bold and the second-best scores are underlined. R-BPtrNet is our model without pre-trained LMs. R-BPtrNet_{BERT} and R-BPtrNet_{RoBERTa} are models using BERT_{BASE} and RoBERTa_{LARGE}, respectively.

We add 10% dropout (Srivastava et al., 2014) on the input of all LSTMs for regularization. Following previous work (Zeng et al., 2018; Wei et al., 2020; Wang et al., 2020), we set the max length of input sentences to 100. We use the Adam optimizer (Kingma and Ba, 2014) to fine-tune the LM and train other parameters with the learning rates of 10^{-5} and 10^{-3} , respectively. We train our model for 30/90 epochs with the batch size as 32/8 on NYT/WebNLG. At the beginning of the last 10 epochs, we load the parameters with the best validation performance and divide the learning rates by ten. Finally, we choose the best model on the validation set and output results on the test set.

4.3 Performance Evaluation

We present our results on the NYT and WebNLG test sets in Table 2 and compare them with several previous state-of-the-art models:

- **NovelTagging** (Zheng et al., 2017) transformed this task into a sequence tagging problem but neglected the overlapping triples.
- **CopyRE** (Zeng et al., 2018) proposed a seq2seq model based on the copy mechanism to generate triple element as sequences.
- **CopyRE_{RL}** (Zeng et al., 2019) proposed to learn the extraction order of CopyRE with

Reinforcement Learning (RL).

- **GraphRel** (Fu et al., 2019) proposed a graph convolutional network for this task.
- **ETL-Span** (Yu et al., 2019) proposed a decomposition-based tagging scheme.
- **CopyMTL** (Zeng et al., 2020) proposed a Multi-Task Learning (MTL) framework based on CopyRE to address multi-token entities.
- **WDec** (Nayak and Ng, 2020) proposed an encoder-decoder architecture for this task.
- **CGT_{UniLM}** (Ye et al., 2020) proposed a generative transformer module with a triple contrastive training object.
- **CASREL** (Wei et al., 2020) proposed a cascade binary tagging framework.
- **TPLinker** (Wang et al., 2020) proposed a one-stage token pair linking model with a novel handshaking tagging scheme.

From Table 2 we have the following observations: (1) The R-BPtrNet significantly outperforms all previous non-LM methods. It demonstrates the superiority of our seq2seq-based framework to jointly extract explicit and implicit relational triples and improve the performance for this task. Additionally, the R-BPtrNet produces competitive performance to the BERT-based baseline models

Method	NYT								WebNLG							
	Nor.	SEO	EPO	N=1	N=2	N=3	N=4	N \geq 5	Nor.	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
GraphRel	69.6	51.2	58.2	71.0	61.5	57.4	55.1	41.1	65.8	38.3	40.6	66.0	48.3	37.0	32.1	32.1
ETL-Span [†]	88.5	87.6	60.3	85.5	82.1	74.7	75.6	76.9	87.3	91.5	80.5	82.1	86.5	91.4	89.5	91.1
CASREL _{BERT}	87.3	91.4	92.0	88.2	90.3	91.9	94.2	83.7	89.4	92.2	94.7	<u>89.3</u>	90.8	94.2	92.4	90.9
TPLinker _{BERT}	90.1	93.4	94.0	<u>90.0</u>	92.9	93.1	96.1	90.0	87.9	92.5	95.3	88.0	90.1	94.6	93.3	91.6
R-BPtrNet _{BERT}	<u>90.4</u>	<u>94.4</u>	<u>95.2</u>	89.5	<u>93.1</u>	<u>93.5</u>	<u>96.7</u>	<u>91.3</u>	89.5	<u>93.9</u>	<u>96.1</u>	88.5	<u>91.4</u>	<u>96.2</u>	<u>94.9</u>	<u>94.2</u>
R-BPtrNet _{RoBERTa}	91.2	95.3	96.1	90.5	93.6	94.2	97.7	92.1	89.9	94.4	97.4	89.3	91.7	96.5	95.8	94.8

Table 3: F_1 scores on sentences with different overlapping patterns and different triple numbers. The best scores are in bold and the second-best scores are underlined. [†] marks scores reproduced by (Wang et al., 2020).

without using BERT. It shows that the improvements of our model come not primarily from the pre-trained LM representations, but from the introduction of relational reasoning patterns to this task. (2) R-BPtrNet_{BERT} outperforms BERT-based baseline models. It indicates that our method can effectively extract implicit relational triples with the assistance of the triple-retaining external memory and the pattern-capturing RN. (3) R-BPtrNet_{RoBERTa} further outperforms R-BPtrNet_{BERT} and other baseline methods. It indicates that the more powerful LM brings more prior knowledge and real-world relational facts, enhancing the model’s ability to learn real-world relational reasoning patterns.

4.4 Performance on Different Sentence Types

To demonstrate the ability of our model in handling the multiple triples and overlapping triples of a sentence, we split the test sets of NYT and WebNLG datasets according to the overlapping

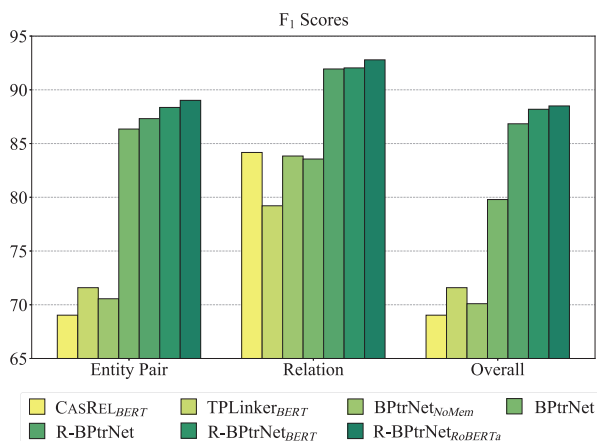


Figure 3: An ablation study on a manually selected subset with rich implicit relational triples.

patterns and the number of triples. We conduct further experiments on these subsets and report the results in Table 3, from which we can observe that: (1) The R-BPtrNet_{RoBERTa} and R-BPtrNet_{BERT} both significantly outperform previous models on the SPO and EPO subsets of NYT and WebNLG datasets. It proves the validity of our method to address the overlapping triple problem. Moreover, we find that implicit relational triples usually overlap with others. Therefore, the improvements on the overlapping subsets also validate the effectiveness of our method for extracting implicit relational triples. (2) R-BPtrNet_{RoBERTa} and R-BPtrNet_{BERT} both bring improvements to sentences with multiple triples compared to baseline models. It indicates that our method can effectively extract multiple relational triples from a sentence. Furthermore, we observe more significant improvements when the number of triples grows. We hypothesize that this is because implicit relational triples are more likely to occur in sentences with more triples. Our model extracts the implicit relational triples more correctly and improves the performance.

4.5 Ablation Study on Implicit Triples

We run an ablation study to investigate the contribution of each component in our model to the implicit relational triples. We manually select 134 sentences with rich implicit triples from the NYT test set⁴. We conduct experiments on the subset

⁴We first select sentences that contain at least two overlapping relational triples. For example, if a sentence contains entity A , B and C , and if $A \rightarrow B$ and $B \rightarrow C$ exists or $A \rightarrow B$ and $A \rightarrow C$ exists, this sentence is selected. Note that $A \rightarrow B$ and $B \rightarrow A$ are counted as one triple during this selecting procedure. Then, we manually check all selected sentences and keep the ones with implicit relational triples.

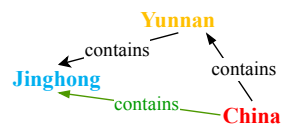

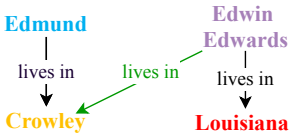
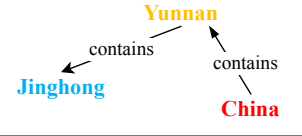

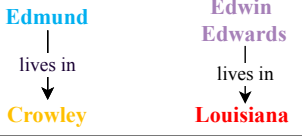
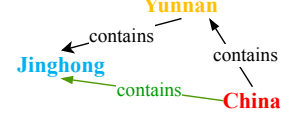
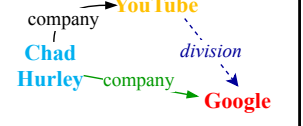
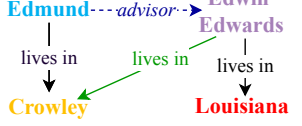
Sentence	... organizing an expedition starting in November in Jinghong , a small city in the Yunnan province in China .	We 're providing a new outlet for them for distribution," said Chad Hurley , chief executive and ... of YouTube , a division of Google Edmund , was an influential municipal judge in Crowley , who was ... as well as a close adviser to former Louisiana Gov. Edwin Edwards ...
Ground Truth			
TPLinker			
Ours			

Figure 4: Examples of sentences with implicit relational triples, and the predictions from the TPLinker_{BERT} and R-BPtrNet_{BERT} model. Bold and colored texts are entities. We distinguish different entities with different colors. Explicit and implicit relational triples are represented by black and green solid arrows, respectively. Blue dashed arrows indicate explicit relational connections between entities, but they do not appear as relational triples because their relation types don't belong to the pre-defined relations of the dataset.

using the following ablation options:

- **R-BPtrNet_{RoBERTa}** and **R-BPtrNet_{BERT}** are the full models using RoBERTa_{LARGE} and BERT_{BASE} as LMs, respectively.
- **R-BPtrNet** removes the **pre-trained LM representations** from the full model.
- **BPtrNet** removes **the RN** from the R-BPtrNet. Under this setting, we feed a gated summation of the memory into the decoder's input of the next time step.
- **BPtrNet_{NoMem}** removes **the external memory** from the BPtrNet, which means that the previously extracted triples are not retained.

We compare the performance of these options with the previous BERT-based models. We also analyze the performance on predicting only the entity pairs and the relations, respectively. We illustrate the results in Figure 3, from which we can observe that: (1) BPtrNet_{NoMem} produces comparable results to the baseline models. We speculate that it benefits from the seq2seq structure and the previous triples are embedded into the decoder's hidden states. (2) BPtrNet brings huge improvements over the BPtrNet_{NoMem} to the entity pair and the triple F_1 scores. It indicates that the external memory effectively helps discover entity pairs that have implicit relational connections by retaining previously

extracted triples. (3) R-BPtrNet brings significant improvements over the BPtrNet to the relation and the triple F_1 scores. It indicates that the RN effectively captures the relational reasoning patterns and enhances the relation type inference of implicit relations. (4) The pre-trained LMs only bring minor improvements. It proves that the effectiveness of our model comes primarily from the external memory and the introduction of relational reasoning patterns rather than the pre-trained LMs.

4.6 Case Study

Figure 4 shows the comparison of the best previous model TPLinker_{BERT} and our R-BPtrNet_{BERT} model on three example sentences from the implicit subset in Section 4.5. The first example contains the transitive pattern of the relation "contains". The second example contains a multi-hop relation path pattern between "Chad Hurley" and "Google" through the intermediate entity "Youtube". The third example contains a composite pattern between the siblings "Crowley" and "Edwin Edwards" with a common ancestor "Edmund". We can observe that the TPLinker_{BERT} model fails to extract the implicit relational triples. The R-BPtrNet_{BERT} successfully captures various reasoning patterns in the real world and effectively extracts all the implicit relational triples in the examples.

5 Conclusion

In this paper, we propose a unified framework to extract explicit and implicit relational triples jointly. To discover entity pairs that may have implicit relational connections, we propose a binary pointer network to extract relational triples relevant to each word sequentially and introduce an external memory to retain the extracted triples. To derive the relation types of the implicitly connected entity pairs, we propose to introduce real-world relational reasoning patterns to this task and capture the reasoning patterns with a relation network. We conduct experiments on two benchmark datasets, and the results prove the effectiveness of our method.

Acknowledgement

We would like to thank the anonymous reviewers for their constructive comments on this paper. This work was supported by the National Natural Science Foundation of China under Grant numbers U1936208, U1936216, and 61862002.

References

- Gabor Angeli and Christopher D Manning. 2013. Philosophers are mortal: Inferring the truth of unseen facts. In *CoNLL*, pages 133–142.
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.
- Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *ACL-HLT*, pages 551–560.
- Dai Dai, Xinyan Xiao, Yajuan Lyu, Shan Dou, Qiaoqiao She, and Haifeng Wang. 2019. Joint extraction of entities and overlapping relations using position-attentive sequence labeling. In *AAAI*, volume 33, pages 6300–6308.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.
- Xin Dong, Evgeniy Gabilovich, Jeremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmman, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *KDD*, pages 601–610.
- Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *ACL*, pages 1409–1418.
- Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for nlg micro-planning. In *ACL*.
- Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471.
- Matthew R Gormley, Mo Yu, and Mark Dredze. 2015. Improved relation extraction with feature-rich compositional embedding models. In *EMNLP*, pages 1774–1784.
- Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *COLING*, pages 2537–2547.
- Ningning Jia, Xiang Cheng, and Sen Su. 2020. Improving knowledge graph embedding using locally and globally attentive relation paths. In *European Conference on Information Retrieval*, pages 17–32. Springer.
- Kuang Jun, Cao Yixin, Zheng Jianbing, He Xiangnan, Gao Ming, and Zhou Aoying. 2020. Improving neural relation extraction with implicit mutual relations. In *ICDE*, pages 1021–1032.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.
- Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *ACL*, pages 402–412.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*, pages 1105–1116.
- Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In *AAAI*, volume 34, pages 8528–8535.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

- Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *WWW*, pages 1015–1024.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.
- Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. 2017. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, pages 4967–4976.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.
- Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700.
- Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018. Joint extraction of entities and relations based on a novel graph scheme. In *IJCAI*, pages 4461–4467.
- Yucheng Wang, Bowen Yu, Yueyang Zhang, Tingwen Liu, Hongsong Zhu, and Limin Sun. 2020. Tplinker: Single-stage joint extraction of entities and relations through token pair linking. *arXiv preprint arXiv:2010.13415*.
- Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. A novel cascade binary tagging framework for relational triple extraction. In *ACL*, pages 1476–1488.
- Hongbin Ye, Ningyu Zhang, Shumin Deng, Mosh Chen, Chuanqi Tan, Fei Huang, and Huajun Chen. 2020. Contrastive triple extraction with generative transformer. *arXiv preprint arXiv:2009.06207*.
- Bowen Yu, Zhenyu Zhang, and Jianlin Su. 2019. Joint extraction of entities and relations based on a novel decomposition strategy. *arXiv preprint arXiv:1909.04273*.
- Xiaofeng Yu and Wai Lam. 2010. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *COLING 2010: Posters*, pages 1399–1407.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research*, 3(Feb):1083–1106.
- Daojian Zeng, Haoran Zhang, and Qianying Liu. 2020. Copymtl: Copy mechanism for joint extraction of entities and relations with multi-task learning. In *AAAI*, pages 9507–9514.
- Xiangrong Zeng, Shizhu He, Daojian Zeng, Kang Liu, Shengping Liu, and Jun Zhao. 2019. Learning the extraction order of multiple relational facts in a sentence with reinforcement learning. In *EMNLP-IJCNLP*, pages 367–377.
- Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *ACL*, pages 506–514.
- Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *ACL*, pages 1227–1236.
- GuoDong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *ACL*, pages 427–434.
- Hao Zhu, Yankai Lin, Zhiyuan Liu, Jie Fu, Tat-Seng Chua, and Maosong Sun. 2019. Graph neural networks with generated parameters for relation extraction. In *ACL*, pages 1331–1339.