

# RTF: Region-based Table Filling Method for Relational Triple Extraction

Ning An, Lei Hei, Yong Jiang, Weiping Meng, Jingjing Hu  
Boran Huang, Feiliang Ren\*

School of Computer Science and Engineering  
Northeastern University, Shenyang, 110169, China  
renfeiliang@cse.neu.edu.cn

## Abstract

Relational triple extraction is crucial work for the automatic construction of knowledge graphs. Existing methods only construct shallow representations from a token or token pair-level. However, previous works ignore local spatial dependencies of relational triples, resulting in a weakness of entity pair boundary detection. To tackle this problem, we propose a novel **Region-based Table Filling** method (RTF). We devise a novel region-based tagging scheme and bi-directional decoding strategy, which regard each relational triple as a region on the relation-specific table, and identifies triples by determining two endpoints of each region. We also introduce convolution to construct region-level table representations from a spatial perspective which makes triples easier to be captured. In addition, we share partial tagging scores among different relations to improve learning efficiency of relation classifier. Experimental results show that our method achieves state-of-the-art with better generalization capability on three variants of two widely used benchmark datasets.

## 1 Introduction

Relational Triple Extraction (RTE) aims to extract relation triples with form (*subject, relation, object*) from massive unstructured text. This task is one of the most fundamental tasks in information extraction, thus can be widely applied to various downstream applications in the real world (Nayak et al., 2021).

Early works (Zelenko et al., 2002; Zhou et al., 2005; Chan and Roth, 2011) often decompose RTE task into two consecutive sub-tasks, entity extraction and relation classification of entity pairs. This pipeline methods suffer from error propagation problems due to inconsistencies during the training and inference stages (Li and Ji, 2014).

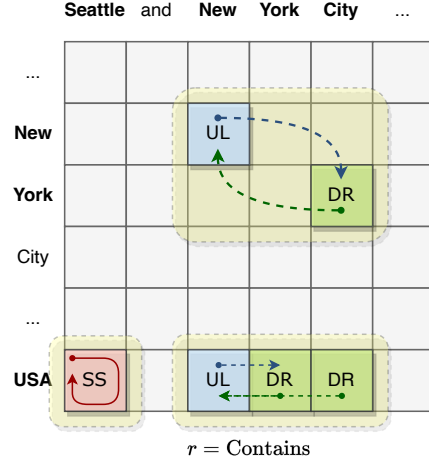


Figure 1: Example of our EPR tagging scheme and bi-directional decoding. This sentence contains Single Entity Overlapping (SEO) and Head Tail Overlapping (HTO) overlapping patterns.

Therefore, end-to-end joint extraction methods have received more widespread attention in recent years (Gupta et al., 2016; Miwa and Bansal, 2016; Zheng et al., 2017; Zhang et al., 2017). Joint extraction aims to extract both entity pairs and their relations simultaneously. The table filling-based approach is one of the best-performing methods for joint extraction. This approach converts RTE into sequence labeling by creating a relation-specific table for each relation and identifying subject and object pairs on the table separately. However, previous RTE methods only construct shallow features at the token or token pair-level, such as applying linear transformations to a single token or token pair. These approaches fail to capture the local spatial dependencies of the relational triples, thus neither allowing token pairs to interact explicitly over a two-dimensional structure, nor modeling interactions within and across triples. This may lead to weaker detection of entity pair boundaries.

As we have observed, each relational triple corresponds to a fixed rectangular region on a

\* Corresponding author.

relation-specific table. If we can identify the boundaries of entity pairs from a spatial perspective, we can extract triples more conveniently. For example, consider the input sentence “*Seattle and New York City in New York are popular in the USA.*” in Figure 1. Suppose the relation of the table is “*Contains*”, and there are four (*subject, object*) pairs: (*New York, New York City*), (*USA, Seattle*), (*USA, New York*), and (*USA, New York City*). Figure 1 represents a wealth of regional information that has been overlooked by other works (Wang et al., 2020; Shang et al., 2022). This figure clearly illustrates that each entity pair can be treated as a rectangular region, which can be determined by recognizing the upper-left and bottom-right endpoints. One special case is (*USA, Seattle*), where the rectangle degenerates to a single point. Furthermore, entity pair boundary divisions are more explicit from a region-level perspective. Such as SEO entity pairs (*USA, New York*) and (*USA, New York City*), when the tag of (*USA, New*) depends not only on the token pair itself but also on its surrounding region, it is easier to argue that *USA* plays a subject role in some relational triple. Overlapping triples are also more likely to be detected at once during this process.

Inspired by the above observations, we propose a simple but effective **Region-based Table-Filling (RTF)** method to handle neglecting spatially local dependencies when extracting relational triples. We devise a novel **Entity Pair as Region (EPR)** tagging scheme and bi-directional decoding strategy to identify each entity pair on the table. EPR tagging scheme recognizes entity pairs by identifying the upper-left and lower-right corners of the region, which can also handle various types of overlapping triples. Bi-directional decoding provides some fault tolerance to the model to a certain extent. To exploit the spatial local dependencies, we apply convolution to construct region-level table representations. This allows each token pair representation not only to depend on itself but also on the surrounding token pairs. When the convolutional kernel slides over the table, the entire triple or even across triples can be simultaneously interacted. Moreover, we shared a portion of tagging scores among different relations, i.e., the scores are divided into relation-dependent and relation-independent parts. This allows the relation classifier to learn the relation-specific residuals only, which reduces the burden of the relation classifier.

We comprehensively conducted experiments on two benchmark datasets, NYT (Riedel et al., 2010) and WebNLG (Gardent et al., 2017). Experimental results show that our approach achieves state-of-the-art performance with better generalization. We summarize our contributions as follows:

- We propose a novel region-based table filling relational triple extraction method, which devises a novel EPR tagging scheme and bi-directional decoding algorithm to extract triples by determining the upper-left and bottom-right endpoints of the region.
- We introduce convolution to construct region-level table representations and share tagging scores of different relations to fully utilize regional correlations for improving entity pair boundary recognition.
- Our approach achieves state-of-the-art on two benchmark on three variants of two widely used benchmark datasets with superior generalization capability.

## 2 Related Work

Early works (Zelenko et al., 2002; Zhou et al., 2005; Chan and Roth, 2011) decomposes the relational triple extraction into two separate sub-tasks, entity recognition and relation classification between entity pairs. In this paradigm, all the entities in a sentence are first extracted, and then the extracted entities are combined into entity pairs to determine the relations between them. This pipeline approach suffers from error propagation (Li and Ji, 2014) due to inconsistency in training and inference.

To eliminate inconsistencies during the training and inference phases, joint entity and relation extraction (Gupta et al., 2016; Miwa and Bansal, 2016; Zheng et al., 2017; Zhang et al., 2017) is more widely researched. We roughly divide the existing research methods into three categories according to the research route.

**Sequence Tagging Methods** Wei et al. (2020) proposes a cascade framework that first extracts all subjects in a sentence and then predicts the corresponding object under each relation. Zheng et al. (2021) predicts potential relations in a sentence and aligns extracted subject and object pairs in a one-dimensional sequence with a two-dimensional matrix. Li et al. (2021) used translating decoding to accomplish one-dimensional sequence tagging, similar to TransE (Bordes et al., 2013).

**Table Filling Methods** Recently, table filling-based methods have attracted widespread attention from researchers due to their superior performance. Wang et al. (2020) models entity pair extraction as a token pair linking problem on two-dimensional tables, eliminating gaps in training and inference. Yan et al. (2021) proposes a two-way interaction method for balanced information transmission from two-task interaction perspective between named entity recognition and relation extraction. Shang et al. (2022) extracts entity pairs using the relation-specific horn tagging scheme and resolves RTE task by fine-grained triple classification.

**Sequence to Sequence Methods** Some works use encoder-decoder framework to copy or generate relational triples. Zeng et al. (2018, 2019, 2020) applies the copy mechanism to copy each triple element from the sentence. Sui et al. (2020) regards RTE as a set prediction problem of relational triples, which avoids exposure bias in generative methods. Cabot and Navigli (2021) extracts each triple via pre-training and structured generation. Tan et al. (2022) focusing on constructing instance-level representations for relational triples.

However, either type of method ignores the spatial local dependencies of triples. Even though table filling-based RTE methods have been constructed with two-dimensional table representations, tagging is only performed with a token pair-level representation. Compared with existing works, our goal is to construct region-level table representations with regional correlations on two-dimensional table structures, to tackle the problem of underutilized local spatial dependencies of triples.

### 3 Methodology

#### 3.1 Task Definition

For a given sentence  $S = (x_1, x_2, \dots, x_N)$  containing  $N$  tokens, the target of the relational triple extraction task is to extract all the triples  $T = \{(e_i, r_t, e_j) | e \in \mathcal{E}, r \in \mathcal{R}\}$ , where  $e_i, e_j, r_t$  are subject, object and the relation between them respectively,  $\mathcal{E}$  is the set of entities formed by consecutive tokens, and  $r_t$  belongs to the predefined set of relations  $\mathcal{R} = \{r_1, r_2, \dots, r_m\}$ .

Identifying the entity pair  $(e_i, e_j)$  can be regarded as finding the start and end positions  $(x_i, x_j)$  of  $e_i$ , and identifying the start and end positions  $(x_p, x_q)$  of  $e_j$ . Since the boundaries of  $(e_i, e_j)$  are determined simultaneously on each relation-

specific table, entity pairs and relations can be extracted in parallel.

#### 3.2 EPR Tagging Scheme

We propose a novel Entity Pair as Region (EPR) tagging scheme, which can identify triples from the spatial perspective. Following previous table filling-based approaches (Wang et al., 2020; Shang et al., 2022), we maintain a relation-specific two-dimensional table for each relation  $r$ , and identify entity pairs  $(e_i, e_j)$  on each table.

Specifically, entity pairs  $(e_i, e_j)$  need to be determined by the start and end positions  $(x_i, x_j)$  of subject  $e_i$  and the start and end positions  $(x_p, x_q)$  of object  $e_j$ . Obviously, this is equivalent to finding the Upper-Left coordinate  $(x_i, x_p)$  and the Bottom-Right coordinate  $(x_j, x_q)$  of a rectangular region on a relation-specific table, which we denote by “UL” and “BR”, respectively. When the subject and object both have a single token, the entity pair coincides with the special case of a rectangular region collapsed into a Single Point, which we represent with “SP” individually. All remaining cells are tagged as “None”. As shown in Figure 1, entity pair (*New York, New York City*) is a rectangular region on the relation “Contains” table, and the upper-left corner of the region (*New, New*) and the bottom-right corner (*York, City*) should be filled with “UL” and “BR” respectively. The corresponding decoding strategy is described in Section 3.4.

Our tagging scheme can address various overlapping patterns in complex scenarios (Zeng et al., 2018), such as Single Entity Overlapping (SEO), Entity Pair Overlapping (EPO), Head Tail Overlapping (HTO), because they are all non-intersecting in regional perspective. Notably, although our tagging scheme can be similar to OneRel (Shang et al., 2022), it takes up 3 times more memory than EPR because OneRel ignores the case where a triple degenerates to a single point.

#### 3.3 RTF

##### 3.3.1 Table Representation

Given an input sentence  $S = (x_1, x_2, \dots, x_N)$ , we first use the pre-trained language model to obtain the contextual representation of each token  $x_i$  in the sentence:

$$\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_N = \text{PLM}(x_1, x_2, \dots, x_N) \quad (1)$$

where  $\mathbf{h}_i \in \mathbb{R}^d$  is token contextual representation,  $N$  is the sequence length of sentence. PLM

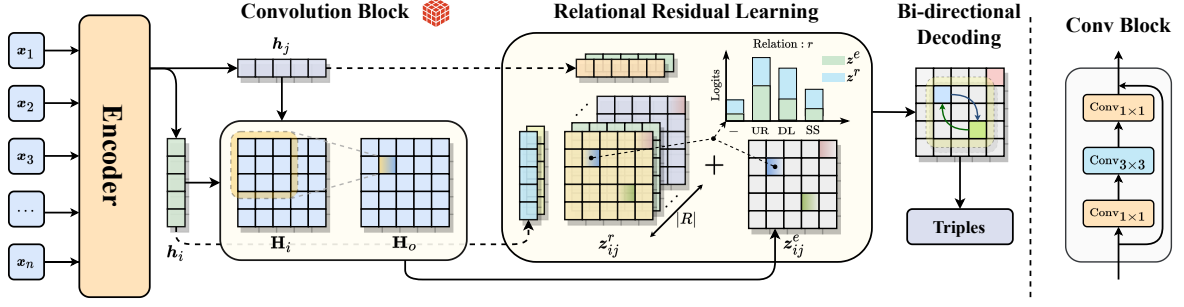


Figure 2: RTF overview. **Left:** First, we construct a token pair-level table representation. Then capture the local dependencies and construct a region level table representation by Convolution Block. After that, we obtain the tagging score of each cell from the sharing score of all relations  $z_{ij}^e$  and the relation-specific residual  $z_{ij}^r$ . Finally, we obtain triples by bi-directional decoding strategy. **Right:** Details of the Convolution Block. Similar to ResNet (He et al., 2016), Conv Block consists of  $1 \times 1$ ,  $3 \times 3$ ,  $1 \times 1$  convolution and residual connection.

can be arbitrary pre-trained bi-directional language models, i.e. BERT (Devlin et al., 2019).

Then we use a linear transformation to construct the shallow representation of each token pair  $(x_i, x_j)$  in the two-dimensional table:

$$h_{ij} = \phi(\mathbf{W}_t[h_i; h_j] + \mathbf{b}_t) \quad (2)$$

where  $\phi(\cdot)$  denotes ReLU non-linear activation function,  $[\cdot]$  is the concatenation operation,  $\mathbf{W}_t \in \mathbb{R}^{2d \times d}$ ,  $\mathbf{b}_t \in \mathbb{R}^d$  are trainable parameters.

### 3.3.2 Convolution Block

To construct enriched region-level table representations, we need a component to fully exploit the association of triples over regions, which can be useful for detecting entity pairs. We think that convolution is very appropriate to achieve this purpose. For instance, in Figure 1, when we take a  $3 \times 3$  convolution kernel to construct a region-level table representation, the central token pair can interact with 8 surrounding token pairs explicitly. Convolution can perceive the difference in semantic information inside (denoted by yellow rectangle) and outside (indicated by the gray cells outside the yellow rectangle) the entity pair boundary<sup>1</sup>. When the convolution kernel covers the bottom yellow region of the table, it not only allows the entire region of  $(USA, New York)$  to interact simultaneously, but also allows  $(USA, New York)$ ,  $(USA, New York City)$  to interact across triples on the table.

ResNet (He et al., 2016) has been proven to be an effective convolutional neural network backbone in computer vision, which is especially strong at capturing local dependencies from a spatial perspec-

tive. Therefore, we adopt the Convolution Block as our component for constructing region-level table representations. Token pair-level table representation  $\mathbf{H}_i$  is composed of  $h_{ij}$ . When  $\mathbf{H}_i$  is input into the Convolution Block, the region-level table representation  $\mathbf{H}_o$  will be obtained. So our Convolution Block is defined as follows:

$$\begin{aligned} \mathbf{H}_o &= \phi(\text{Conv}_{1 \times 1}(\mathbf{H}_i)) \\ \mathbf{H}_o &= \phi(\text{Conv}_{3 \times 3}(\mathbf{H}_o)) \\ \mathbf{H}_o &= \phi(\text{Conv}_{1 \times 1}(\mathbf{H}_o)) \\ \mathbf{H}_o &= \mathbf{H}_o + \mathbf{H}_i \end{aligned} \quad (3)$$

where  $\mathbf{H} \in \mathbb{R}^{N \times N \times d}$  is table representation. Furthermore, layer normalization (Ba et al., 2016) is applied after each convolution operation to accelerate the convergence of the trainable parameters in the convolution layer.

### 3.3.3 Relational Residual Learning

Regardless of the relation categories, the ability to determine entity pairs by finding regions on each table should be generalized across all relations, rather than restricted to particular relations. For instance, in Figure 1,  $(New, New)$  looks like it could be the head of some entity pair (i.e., the upper-left corner of the region), so it should be slightly more probable to fill with “UL” under any relation. In particular, a large number of negative cells on each table should be filled with “None” in either relation.

Inspired by Su et al. (2022), we separate the tagging score of each cell  $z_{ij}$  into the sum of two parts: relation-independent score  $z_{ij}^e$  and relation-dependent score  $z_{ij}^r$ . As shown in Figure 2,  $z_{ij}^e$  is considered as a basic score for all cells, which is obtained from the region-level table representation.  $z_{ij}^r$  is regarded as the relation-specific residual

<sup>1</sup>We illustrate this process in Appendix B.

based on  $z_{ij}^e$ , which learned by different relation classifiers. This means that each relation classifier only needs to focus on fitting the residual  $z_{ij} - z_{ij}^e$  for a particular relation, rather than on fitting  $z_{ij}^e$ , because  $z_{ij}^e$  is handled by one additional classifier which shared by all relations.

Therefore, if all classifiers can share and leverage this relation-independent information, not only can alleviate the severe class imbalance with the introduction of convolution (Buda et al., 2018), but also help alleviate the learning burden of relation-specific classifiers.

Specifically, under the relation  $r$ , the logits  $z_{ij}$  of token pair  $(x_i, x_j)$  in the table are divided into  $z_{ij}^e$  and  $z_{ij}^r$ , which are computed as follows:

$$\begin{aligned} z_{ij}^e &= \mathbf{W}_e \mathbf{h}_{ij} + \mathbf{b}_e \\ z_{ij}^r &= \mathbf{W}_r [\mathbf{h}_i; \mathbf{h}_j] + \mathbf{b}_r \end{aligned} \quad (4)$$

where  $\mathbf{W}_e, \mathbf{W}_r \in \mathbb{R}^{d \times |L|}$ ,  $\mathbf{b}_e, \mathbf{b}_r \in \mathbb{R}^{|L|}$  are trainable parameters.  $|L|$  is the number of tags, which is 4 under our EPR tagging scheme.

We simply add these two scores directly and use Softmax to obtain the probability distribution  $P_r$  of each tag in each cell:

$$P_r(\mathbf{y}_{ij}|S) = \text{Softmax}(z_{ij}^e + z_{ij}^r) \quad (5)$$

### 3.4 Bi-directional Decoding

Our decoding process is logically straightforward. Since two points determine a region (or a triple), only a heuristic nearest neighbor matching algorithm is required to match “UL” to the nearest “BR” on the bottom-right, and similarly, to match “BR” to the nearest “UL” on the upper-left. However, the model may give wrong predictions, so the decoding algorithm is required to have some fault tolerance mechanism. For example, the model may misjudge the number of tokens of an entity, which may result in “UL” or “BR” being identified as “SP”. In this case, when the model “UL”/“BR” fails to match “BR”/“UL”, we should try to match “SP”. Our pseudo-code of bi-directional decoding algorithm is presented in Algorithm 1.

Taking Figure 1 as an example, suppose token pairs  $(USA, Seattle)$ ,  $(USA, New)$ ,  $(USA, York)$ ,  $(USA, City)$  are filled with “SP”, “UL”, “BR”, “BR” respectively. First, we need to find all the single points on the table, then we obtain the entity pair  $(USA, Seattle)$ . Then, taking the “UL”  $(USA, New)$  as the anchor point, finding the nearest “BR”  $(USA, York)$  on the bottom-right, and ob-

### Algorithm 1 Bi-directional Decoding Strategy

**Input:** Given sentence  $S = (x_1, x_2, \dots, x_N)$ , predefined relation set  $\mathcal{R}$ , prediction labels  $\mathbf{Y}_r \in \mathbb{R}^{N \times N}$  for the 2D table of each relation  $r \in \mathcal{R}$ .

**Output:** Predicted relational triple set  $T$ .

```

1 for each  $r \in \mathcal{R}$  do
2   Find the token pairs predicted as “SP”, “UL”, “BR”
   from the relation-specific table  $\mathbf{Y}_r$ , denoted as  $\mathbf{Y}_r^{SP}$ ,
    $\mathbf{Y}_r^{UL}$  and  $\mathbf{Y}_r^{BR}$ , respectively.
3   for token pair  $(x_i, x_j) \in \mathbf{Y}_r^{SP}$  do
4     Extract entity pair  $(s, o)$  from sentence  $S$  under
      $(x_i, x_j)$ , then add triple  $(s, r, o)$  to  $T$ .
5   for token pair  $(x_i, x_j) \in \mathbf{Y}_r^{UL}$  do // UL  $\rightarrow$  BR
6     Compute the Euclidean distance between each token
     pair  $(x_i, x_j)$  and the token pair  $(x_m, x_n) \in$ 
      $\mathbf{Y}_r^{BR}$  satisfying the token pair located in the bottom-
     right, trying to match with the nearest “BR”.
7     if Nearest “BR”  $(x_p, x_q)$  not exists then
8       Compute the Euclidean distance between
       each token pair  $(x_i, x_j)$  and the token pair
        $(x_m, x_n) \in \mathbf{Y}_r^{SP}$  satisfying the token pair lo-
       cated in the bottom-right, trying to match with
       the nearest “SP”.
9     if Nearest location  $(x_p, x_q)$  exists then
10      Extract the entity pair  $(s, o)$  from the sentence  $S$ 
      according to subject position  $(x_i, x_p)$  and object
      position  $(x_j, x_q)$ , then add triple  $(s, r, o)$  to  $T$ .
11    for token pair  $(x_i, x_j) \in \mathbf{Y}_r^{BR}$  do // UL  $\leftarrow$  BR
12      Following the same manner as above for “UL” find-
      ing “BR”, finally add decoded triple  $(s, r, o)$  to  $T$ .
13 return  $T$ 
```

taining the entity pair  $(USA, New York)$ . Finally, using “BR”  $(USA, City)$  as the anchor point, find the nearest “UL”  $(USA, New)$  on the upper-left, and obtain the entity pair  $(USA, New York City)$ . Thus we decode the entity pairs  $(USA, Seattle)$ ,  $(USA, New York)$ ,  $(USA, New York City)$  for the relation “Contains”.

### 3.5 Loss Function

Our training object is to minimize cross entropy loss function:

$$\mathcal{L} = -\frac{1}{N^2 \times |\mathcal{R}|} \sum_{i=1}^N \sum_{j=1}^N \sum_{r=1}^{|\mathcal{R}|} \log P_r(\mathbf{y}_{ij} = \mathbf{y}'_{ij}|S) \quad (6)$$

where  $\mathbf{y}'_{ij}$  is golden label of relation-specific table under our tagging scheme,  $r$  is relation in predefined set of relations  $\mathcal{R}$ .

## 4 Experiments

### 4.1 Experiments Settings

**Datasets** According to previous work, we evaluate our proposed model by choosing two widely used datasets for relational triple extraction: NYT (Riedel et al., 2010) and WebNLG (Gardent

et al., 2017). Typically there are two different variants, partial matching (denote with \*) and exact matching. Partial matching means that only the last word of each subject and object is annotated, while exact matching strictly annotates the first and last word of each subject and object. However, Lee et al. (2022) points out that evaluating generalization ability of RTE methods with existing datasets is unrealistic. So we still compare our model with other state-of-the-art models in the rearranged dataset NYT-R and WebNLG-R (Lee et al., 2022). We will discuss this further in Section 4.4. Statistics of all datasets are shown in Table 1.

| Dataset  | Train | Test | Overlapping Pattern |      |      |     |
|----------|-------|------|---------------------|------|------|-----|
|          |       |      | Normal              | SEO  | EPO  | HTO |
| NYT      | 56196 | 5000 | 3071                | 1273 | 1168 | 117 |
| NYT*     |       |      | 3266                | 1297 | 978  | 45  |
| NYT-R    |       |      | 3965                | 709  | 408  | 110 |
| WebNLG   | 5019  | 703  | 239                 | 448  | 6    | 85  |
| WebNLG*  |       |      | 246                 | 457  | 26   | 83  |
| WebNLG-R |       |      | 389                 | 293  | 9    | 84  |

Table 1: Dataset Statistics. “-R” represents dataset with rearranged variants. Overlapping patterns statistics on test set only. HTO is also referred as SOO in some works Zheng et al. (2021). Detailed statistics on the rest of the dataset can be found in Appendix A.

**Evaluation Metrics** We use three standard evaluation metrics to evaluate how consistently the relational triples are predicted: Micro Precision (Prec.), Recall (Rec.), and F1 Score (F1). In partial matching, triple is considered correct if the predicted relation, tail token of the subject, and tail token of the object are all correct. In exact matching, triple is considered correct if the predicted relation and the entire boundaries of both the subject and object are correct.

**Implementation Details** For a fair comparison with the previous work (Zeng et al., 2018; Fu et al., 2019; Wang et al., 2020; Zheng et al., 2021; Shang et al., 2022), we use bert-base-cased<sup>2</sup> with almost 109M parameters as the default pre-trained language model, with hidden size  $d$  of 768. All parameters are fine-tuned on NYT/WebNLG with learning rates of  $4e-5/3e-5$ , and batchsize of 24/6. We use Adam (Kingma and Ba, 2015) with linear decay of learning rate as optimizer. Consistent with previous work, we set the maximum sentence length for training to 100. All our experiments were

conducted on one NVIDIA RTX A6000 GPU.

## 4.2 Main Results

We took twelve powerful state-of-the-art RTE methods as baselines to compare with our model: CopyRE (Zeng et al., 2018), GraphRel (Fu et al., 2019), RSAN (Yuan et al., 2020), MHSA (Liu et al., 2020), CasRel (Wei et al., 2020), TPLinker (Wang et al., 2020), SPN (Sui et al., 2020), PRGC (Zheng et al., 2021), TDEER (Li et al., 2021), EmRel (Xu et al., 2022), OneRel (Shang et al., 2022), QIDN (Tan et al., 2022). For fair comparison, all results are taken from the original paper.

The main experimental results in Table 2 demonstrate the effectiveness of our proposed method. In total, our method outperforms twelve baselines in almost all metrics, and achieves the new state-of-the-art. Specifically, RTF achieves the best performance among all table filling approaches. RTF improved +0.4%, +0.6% over OneRel on NYT and WebNLG, and +0.5%, +0.3% on NYT\* and WebNLG\*. These improvements may come from fully leveraging regional information.

RTF improves performance significantly for exact matching. In particular, RTF achieves a +0.6% improvement on WebNLG with 216 relations. This indicates RTF fully exploits the local dependencies of triples, instead of using a shallow representation of token pair-level. RTF can also gain from determining the entity pair boundaries which are learned from massive relations. The improvement of RTF in partial matching is lower than that in exact matching. Because NYT\* and WebNLG\* only need to identify the last token of the subject and object, which leads to a smaller gain of RTF from regional information. Therefore, the full potential of RTF cannot be fully utilized for partial matching. It is worth to note that RTF has a remarkable improvement in recall on all datasets. We will further explore this phenomenon in Section 4.5.

## 4.3 Analysis on Complex Scenarios

According to previous works (Wei et al., 2020; Wang et al., 2020; Sui et al., 2020; Zheng et al., 2021; Li et al., 2021; Shang et al., 2022), we also evaluated our model under different overlapping triples and different number of triples.

The results in Table 4 show that our model works effectively in complex scenarios. RTF is comparable to OneRel in handling different overlapping patterns in NYT\*, and outperforms other methods in most scenarios with different numbers of triples.

<sup>2</sup>More details at <https://huggingface.co/bert-base-cased>

| Model                        | NYT*        |             |             | NYT         |             |             | WebNLG*     |             |             | WebNLG      |             |             |
|------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|                              | Prec.       | Rec.        | F1          | Prec.       | Rec.        | F1          | Prec.       | Rec.        | F1          | Prec.       | Rec.        | F1          |
| CopyRE (Zeng et al., 2018)   | 61.0        | 56.6        | 58.7        | -           | -           | -           | 37.7        | 36.4        | 37.1        | -           | -           | -           |
| GraphRel (Fu et al., 2019)   | 63.9        | 60.0        | 61.9        | -           | -           | -           | 44.7        | 41.1        | 42.9        | -           | -           | -           |
| RSAN (Yuan et al., 2020)     | -           | -           | -           | 85.7        | 83.6        | 84.6        | -           | -           | -           | 80.5        | 83.8        | 82.1        |
| MHSA (Liu et al., 2020)      | 88.1        | 78.5        | 83.0        | -           | -           | -           | 89.5        | 86.0        | 87.7        | -           | -           | -           |
| CasRel (Wei et al., 2020)    | 89.7        | 89.5        | 89.6        | -           | -           | -           | 93.4        | 90.1        | 91.8        | -           | -           | -           |
| TPLinker (Wang et al., 2020) | 91.3        | 92.5        | 91.9        | 91.4        | 92.6        | 92.0        | 91.8        | 92.0        | 91.9        | 88.9        | 84.5        | 86.7        |
| SPN (Sui et al., 2020)       | <u>93.3</u> | 91.7        | 92.5        | 92.5        | 92.2        | 92.3        | 93.1        | 93.6        | 93.4        | -           | -           | -           |
| PRGC (Zheng et al., 2021)    | <u>93.3</u> | 91.9        | 92.6        | <u>93.5</u> | 91.9        | 92.7        | 94.0        | 92.1        | 93.0        | 89.9        | 87.2        | 88.5        |
| TDEER (Li et al., 2021)      | 93.0        | 92.1        | 92.5        | -           | -           | -           | 93.8        | 92.4        | 93.1        | -           | -           | -           |
| EmRel (Xu et al., 2022)      | 91.7        | 92.5        | 92.1        | 92.6        | 92.7        | 92.6        | 92.7        | 93.0        | 92.9        | 90.2        | 87.4        | 88.7        |
| OneRel (Shang et al., 2022)  | 92.8        | 92.9        | 92.8        | 93.2        | 92.6        | 92.9        | <u>94.1</u> | 94.4        | 94.3        | 91.8        | 90.3        | 91.0        |
| QIDN (Tan et al., 2022)      | <u>93.3</u> | 92.5        | 92.9        | 93.4        | 92.6        | 93.0        | <u>94.1</u> | 93.7        | 93.9        | -           | -           | -           |
| RTF                          | 93.2        | <b>93.4</b> | <b>93.3</b> | <u>93.5</u> | <b>93.2</b> | <b>93.3</b> | 94.0        | <b>95.3</b> | <b>94.6</b> | <b>92.0</b> | <b>91.1</b> | <b>91.6</b> |

Table 2: Main results. The underline means they are both the maximum.

RTF outperforms OneRel by +1.0% and +1.1% for T=1 and T=3, respectively. OneRel better than RTF for T≥5. We attribute to OneRel increasing the number of valid tags by doubling the sentence length. We disregard this because it would increase memory consumption significantly. In Normal subset, RTF achieves +0.8% and +0.9% improvement over OneRel for NYT\* and WebNLG\*, respectively, significantly outperforming other methods.

In addition, we observe that RTF performs better on WebNLG\* than on NYT\*, especially in EPO where it achieves +0.7% improvement over OneRel. We argue that WebNLG\* has more types of relations, thus there is more entity pair boundary information to share, and more regional dependencies to exploit.

#### 4.4 Analysis on Generalization Capability

Although the current RTE approach has been successful on widely used datasets, however, this performance may be unrealistic. Therefore, we evaluate generalization capability of our method on NYT-R and WebNLG-R (Lee et al., 2022). Same as previous settings (Lee et al., 2022), we trained RTF and OneRel with 300 and 500 epochs on NYT-R and WebNLG-R respectively.

Experimental results in Table 3 show that although OneRel achieves state-of-the-art on widely used datasets, it still cannot outperform TPLinker on NYT-R. This means OneRel has a strong overfitting ability on widely used datasets and prefers memorization to generalization. In contrast, RTF

| Model               | NYT-R       |             |             | WebNLG-R    |             |             |
|---------------------|-------------|-------------|-------------|-------------|-------------|-------------|
|                     | Prec.       | Rec.        | F1          | Prec.       | Rec.        | F1          |
| CasRel              | 65.9        | 60.1        | 62.9        | 73.6        | 64.2        | 68.6        |
| TPLinker            | <b>69.0</b> | 60.8        | 64.7        | 75.1        | 63.9        | 69.1        |
| PRGC                | 63.5        | 61.6        | 62.6        | 61.6        | 62.0        | 61.8        |
| OneRel <sup>†</sup> | 67.3        | 60.8        | 63.9        | 75.2        | 65.7        | 70.1        |
| RTF                 | 68.8        | <b>62.3</b> | <b>65.4</b> | <b>76.5</b> | <b>66.5</b> | <b>71.2</b> |

Table 3: Generalization capability evaluation. † means not reported in the original paper, results are obtained by running the provided source code.

shows consistent improvement on NYT-R and WebNLG-R and displays stronger generalization ability than TPLinker and OneRel. This indicates that our method can effectively generalize to unseen triples. Such capability may be derived from relational residual learning and a more robust decoding strategy. In addition, we can observe from the results of Table 3 that there is a significant difference in the performance of the models between the widely used datasets and the rearranged datasets. Therefore, we also advocate evaluating the generalization capability of RTE methods.

#### 4.5 Analysis on Regional Correlation

In Table 2, we observe that almost all improvements in F1 of RTF derive from recall. RTF improves the recall on NYT\*, NYT, WebNLG\*, WebNLG by +0.5%, +0.6%, +0.9%, +0.8% over

| Model    | NYT*        |             |             |                   |             |             |             |             |             | WebNLG*     |             |             |             |             |             |             |             |             |
|----------|-------------|-------------|-------------|-------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|          | Normal      | EPO         | SEO         | HTO               | T=1         | T=2         | T=3         | T=4         | T≥5         | Normal      | EPO         | SEO         | HTO         | T=1         | T=2         | T=3         | T=4         | T≥5         |
| CasRel   | 87.3        | 92.0        | 91.4        | 77.0 <sup>†</sup> | 88.2        | 90.3        | 91.9        | 94.2        | 83.7        | 89.4        | 94.7        | 92.2        | 90.4        | 89.3        | 90.8        | 94.2        | 92.4        | 90.9        |
| TPLinker | 90.1        | 94.0        | 93.4        | 90.1 <sup>†</sup> | 90.0        | 92.8        | 93.1        | 96.1        | 90.0        | 87.9        | 95.3        | 92.5        | 86.0        | 88.0        | 90.1        | 94.6        | 93.3        | 91.6        |
| SPN      | 90.8        | 94.1        | 94.0        | -                 | 90.9        | 93.4        | 94.2        | 95.5        | 90.6        | -           | -           | -           | -           | 89.5        | 91.3        | 96.4        | 94.7        | 93.8        |
| PRGC     | 91.0        | 94.5        | 94.0        | 81.8              | 91.1        | 93.0        | 93.5        | 95.5        | 93.0        | 90.4        | 95.9        | 93.6        | 94.6        | 89.9        | 91.6        | 95.0        | 94.8        | 92.8        |
| TDEER    | 90.8        | 94.5        | 94.1        | -                 | 90.8        | 92.8        | 94.1        | 95.9        | 92.8        | 90.7        | 95.4        | 93.5        | -           | 90.5        | 93.2        | 94.6        | 93.8        | 92.3        |
| OneRel   | 90.6        | <b>95.1</b> | 94.8        | <b>90.8</b>       | 90.5        | 93.4        | 93.9        | <b>96.5</b> | <b>94.2</b> | 91.9        | 95.4        | 94.7        | <b>94.9</b> | 91.4        | <b>93.0</b> | 95.9        | 95.7        | 94.5        |
| RTF      | <b>91.4</b> | 94.9        | <b>94.9</b> | 88.3              | <b>91.5</b> | <b>93.7</b> | <b>95.0</b> | 96.3        | 93.1        | <b>92.8</b> | <b>96.1</b> | <b>95.0</b> | 94.8        | <b>92.4</b> | 92.9        | <b>96.4</b> | <b>95.8</b> | <b>95.0</b> |

Table 4: F1-score on sentences with different overlapping patterns and different triple numbers. T is the number of triples contained in the sentence. † means the results reported by Zheng et al. (2021).

OneRel, respectively.

We assume RTF improves triple extraction by detecting more entity pairs with regional correlation. Figure 3 results support this hypothesis. RTF recalls more entity pairs ( $h, t$ ) than OneRel on NYT\*, and thus achieves better recall of the triples ( $h, r, t$ ). This implies that introducing regional correlation of triples can detect more entity pairs, which is highly beneficial for RTE. It appears that the improvement in recall of RTF on WebNLG\* is not substantial. We attribute this to the presence of a large number of overlapping triple patterns in RTE, especially multi-relation dataset. Overlapping leads to the same entity or entity pair existing in multiple triples. Although RTF only detects slightly more entity pairs, it significantly improves the recall of triples on WebNLG\*.

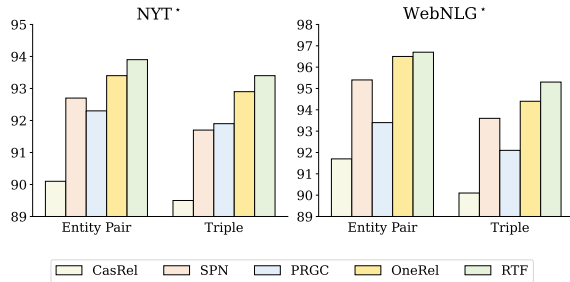


Figure 3: Recall for two different tasks, entity pair recognition and triple extraction.

To further investigate the effectiveness of the two modules of our method that utilize region correlations, we conduct ablation experiments next. From Table 5, we can observe that when either Convolution Block (CB) or Relational Residual (RR) is removed, there is a different degree of performance degradation in our models on both NYT and WebNLG. Removing both simultaneously will result in more noticeable decreases. This illus-

| Model      | NYT         |             |             | WebNLG      |             |             |
|------------|-------------|-------------|-------------|-------------|-------------|-------------|
|            | Prec.       | Rec.        | F1          | Prec.       | Rec.        | F1          |
| RTF        | <b>93.5</b> | <b>93.2</b> | <b>93.3</b> | <b>92.0</b> | <b>91.1</b> | <b>91.6</b> |
| w/o CB     | 93.0        | 92.7        | 92.9        | 91.6        | 89.9        | 90.7        |
| w/o RR     | 93.1        | 92.8        | 93.0        | 90.6        | 90.2        | 90.4        |
| w/o CB, RR | 92.9        | 92.5        | 92.7        | 90.9        | 89.5        | 90.2        |

Table 5: Ablation Study. “CB” means Convolution Block, “RR” means Relational Residual Learning.

trates the effectiveness of both. In particular, removing convolution blocks or relation residuals in WebNLG degrades performance dramatically. Since there are numerous relations in WebNLG, this hurts shared entity pair boundary information across relations. In addition, we observe a significant performance degradation when removing convolution block and the relational residual concurrently (-0.6% on NYT and -1.4% on WebNLG), which suggests that they are complementary to each other and utilize local dependencies more adequately when combined.

## 5 Conclusion

In this study, we propose a novel region-based table filling method to fully exploit the local spatial correlations of relational triples. We devise a novel region-based tagging scheme and decoding algorithm to identify triples by determining two endpoints of each region. We use convolution to capture regional correlations of triples on a table, and reduce learning stress for relation classifiers by sharing the tagging score between different relations. Experimental results demonstrate that RTF achieves state-of-the-art performance on two benchmark datasets with better generalization ability.

## Limitations

Despite the superior performance of table filling-based methods, they have some disadvantages from the perspective of resource usage, such as larger GPU memory consumption and requiring longer training time.

For example, due to the limitation of two-dimensional table structure, when the sentence length increases, the memory consumption increases exponentially and the speed becomes slower due to the expansion of the table area. Alternatively, when the number of relations increases, more cells need to be filled as the number of relations increases. The mentioned problem is not serious in one-dimensional sequence-based labeling approaches.

Therefore, the problem becomes difficult when extending from sentence-level relational triple extraction to document-level relational triple extraction.

## References

- Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. [Layer normalization](#). *CoRR*, abs/1607.06450.
- Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. [Translating embeddings for modeling multi-relational data](#). In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795.
- Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. 2018. [A systematic study of the class imbalance problem in convolutional neural networks](#). *Neural Networks*, 106:249–259.
- Pere-Lluís Huguet Cabot and Roberto Navigli. 2021. [REBEL: relation extraction by end-to-end language generation](#). In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2370–2381. Association for Computational Linguistics.
- Yee Seng Chan and Dan Roth. 2011. [Exploiting syntactico-semantic structures for relation extraction](#). In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 551–560. The Association for Computer Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.
- Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. [Graphrel: Modeling text as relational graphs for joint entity and relation extraction](#). In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1409–1418. Association for Computational Linguistics.
- Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for nlg micro-planning. In *55th annual meeting of the Association for Computational Linguistics (ACL)*.
- Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. [Table filling multi-task recurrent neural network for joint entity and relation extraction](#). In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2537–2547. ACL.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. [Deep residual learning for image recognition](#). In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.
- Diederik P. Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.
- Juhyuk Lee, Min-Joong Lee, June Yong Yang, and Eunho Yang. 2022. [Does it really generalize well on unseen data? systematic evaluation of relational triple extraction methods](#). In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 3849–3858. Association for Computational Linguistics.
- Qi Li and Heng Ji. 2014. [Incremental joint extraction of entity mentions and relations](#). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 402–412. The Association for Computer Linguistics.
- Xianming Li, Xiaotian Luo, Chenghao Dong, Daichuan Yang, Beidi Luan, and Zhen He. 2021. [TDEER: an efficient translating decoding schema for joint extraction of entities and relations](#). 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, pages 8055–8064. Association for Computational Linguistics.
- Jie Liu, Shaowei Chen, Bingquan Wang, Jiaxin Zhang, Na Li, and Tong Xu. 2020. [Attention as relation: Learning supervised multi-head self-attention for relation extraction](#). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3787–3793. International Joint Conferences on Artificial Intelligence Organization. Main track.
- Makoto Miwa and Mohit Bansal. 2016. [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, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.
- Tapas Nayak, Navonil Majumder, Pawan Goyal, and Soujanya Poria. 2021. [Deep neural approaches to relation triplets extraction: a comprehensive survey](#). *Cogn. Comput.*, 13(5):1215–1232.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. [Modeling relations and their mentions without labeled text](#). In *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III*, volume 6323 of *Lecture Notes in Computer Science*, pages 148–163. Springer.
- Yuming Shang, Heyan Huang, and Xianling Mao. 2022. [Onerel: Joint entity and relation extraction with one module in one step](#). In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11285–11293. AAAI Press.
- Jianlin Su, Ahmed Murtadha, Shengfeng Pan, Jing Hou, Jun Sun, Wanwei Huang, Bo Wen, and Yunfeng Liu. 2022. [Global pointer: Novel efficient span-based approach for named entity recognition](#). *CoRR*, abs/2208.03054.
- Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, Xiangrong Zeng, and Shengping Liu. 2020. [Joint entity and relation extraction with set prediction networks](#). *CoRR*, abs/2011.01675.
- Zeqi Tan, Yongliang Shen, Xuming Hu, Wenqi Zhang, Xiaoxia Cheng, Weiming Lu, and Yueting Zhuang. 2022. [Query-based instance discrimination network for relational triple extraction](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 7677–7690. Association for Computational Linguistics.
- 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](#). In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1572–1582. International Committee on Computational Linguistics.
- Zhepei Wei, Jianlin Su, Yue Wang, Yuan Tian, and Yi Chang. 2020. [A novel cascade binary tagging framework for relational triple extraction](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1476–1488. Association for Computational Linguistics.
- Benfeng Xu, Quan Wang, Yajuan Lyu, Yabing Shi, Yong Zhu, Jie Gao, and Zhendong Mao. 2022. [Emrel: Joint representation of entities and embedded relations for multi-triple extraction](#). In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 659–665. Association for Computational Linguistics.
- Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. 2021. [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*, pages 185–197. Association for Computational Linguistics.
- Yue Yuan, Xiaofei Zhou, Shirui Pan, Qiannan Zhu, Zeliang Song, and Li Guo. 2020. [A relation-specific attention network for joint entity and relation extraction](#). In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4054–4060. ijcai.org.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. [Kernel methods for relation extraction](#). In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, EMNLP 2002, Philadelphia, PA, USA, July 6-7, 2002*, pages 71–78.
- Daojian Zeng, Haoran Zhang, and Qianying Liu. 2020. [Copymtl: Copy mechanism for joint extraction of entities and relations with multi-task learning](#). In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9507–9514. AAAI Press.
- 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 *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 367–377. Association for Computational Linguistics.

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 *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 506–514. Association for Computational Linguistics.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2017. [End-to-end neural relation extraction with global optimization](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1730–1740. Association for Computational Linguistics.

Hengyi Zheng, Rui Wen, Xi Chen, Yifan Yang, Yunyan Zhang, Ziheng Zhang, Ningyu Zhang, Bin Qin, Xu Ming, and Yefeng Zheng. 2021. [PRGC: potential relation and global correspondence based joint relational triple extraction](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6225–6235. Association for Computational Linguistics.

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 *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1227–1236. Association for Computational Linguistics.

Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. [Exploring various knowledge in relation extraction](#). In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 427–434. The Association for Computer Linguistics.

## A Detailed dataset statistics

The remaining detailed statistics for all variants of NYT and WebNLG are shown in Table 6 and Table 7. NYT has 24 relations, while WebNLG has far more relations than NYT. Avg.Len represents the average length of entity tokens, and Avg.Area represents the average area of the triple on the table. Note that the area of all triples are almost always around 9, just covered by one  $\text{Conv3} \times 3$ .

| Dataset  | Relations | Avg. Len | Avg. Area |
|----------|-----------|----------|-----------|
| NYT      | 24        | 1.83     | 3.25      |
| NYT*     | 24        | 1.33     | 1.71      |
| NYT-R    | 24        | 2.40     | 5.11      |
| WebNLG   | 216       | 3.30     | 10.84     |
| WebNLG*  | 171       | 1.56     | 2.43      |
| WebNLG-R | 216       | 3.46     | 12.66     |

Table 6: Statistics on the number of relations, the average length of entities, and the average area of triads. Avg. is an abbreviation for Average.

| Dataset  | T=1  | T=2  | T=3 | T=4 | T≥5 |
|----------|------|------|-----|-----|-----|
| NYT      | 3089 | 1137 | 300 | 317 | 157 |
| NYT*     | 3244 | 1045 | 312 | 291 | 108 |
| NYT-R    | 3958 | 733  | 192 | 50  | 67  |
| WebNLG   | 256  | 175  | 138 | 93  | 41  |
| WebNLG*  | 266  | 171  | 131 | 90  | 45  |
| WebNLG-R | 411  | 143  | 91  | 59  | 10  |

Table 7: Statistics for the number of different triples in each sentence in the test set.

## B Interpretation of Convolutional Interaction

Figure 4 explains the interaction process when convolution is combined with EPR tagging scheme.

As the convolution kernel sweeps to the upper-left of the “UL”, the gray cells are located outside the entity pair boundary and the blue area is located inside. It is easier to detect the location of “UL” if the semantics inside the boundary and outside the boundary are different. When the entity pair collapses to a single point, the region information is no longer explicitly in effect, but degenerates to a token level or token pair-level representation.

## C Visualization

Figure 5 shows how RTF obtains each tagging score. For the input sentence “There are 60 floors at 300 North LaSalle in Illinois.”, there are two

SEO relational triples (*300 North LaSalle*, location, *Illinois*), (*300 North LaSalle*, floorCount, *60*).

First, we can observe the significant difference in the working patterns of  $z_{ij}^e$  and  $z_{ij}^r$ . No matter for which valid tag (UL, BR, SP), the highlighted part of  $z_{ij}^e$  is always concentrated in a two-dimensional region. In contrast, the highlighted areas of  $z_{ij}^r$  are always concentrated on rows and columns. This indicates that the Convolution Block utilizes region-level features.

Moreover, observe that the relation-independent score  $z_{ij}^e$  always accurately predicts the entity pair boundaries of all triples. For example, in tag “UL”,  $z_{ij}^e$  accurately detects that the token pair (*300*, *60*), (*300*, *Illinois*) must be the upper-left corner of the entity pair, regardless of whichever relationship they belong to. This means that  $z_{ij}^e$  is a more general score shared by all kinds of relations, thus reducing the learning burden of the relation classifier.

Finally, different relational classifiers learn different scores  $z_{ij}^r$  for the same tag, just complementary to  $z_{ij}^e$ . For example, in tag “BR”, the classifier with the relation “location” gives high scores for both (*300*, *Illinois*) and (*alle*, *Illinois*), but since (*alle*, *Illinois*) has a higher score in  $z_{ij}^e$ , so (*300*, *Illinois*) is not really the bottom-right corner of the entity pair. This means that  $z_{ij}^e$ ,  $z_{ij}^r$  are working by combination.

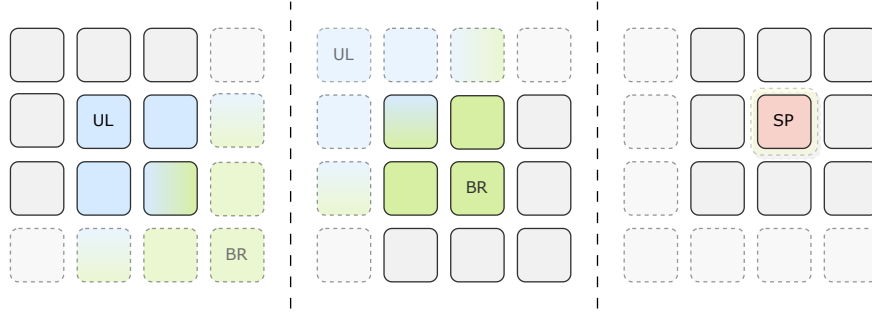


Figure 4: Interaction when combining convolution with EPR tagging scheme. From left to right is the scene when the convolution interacts with “UL”, “BR” and “SP” respectively.

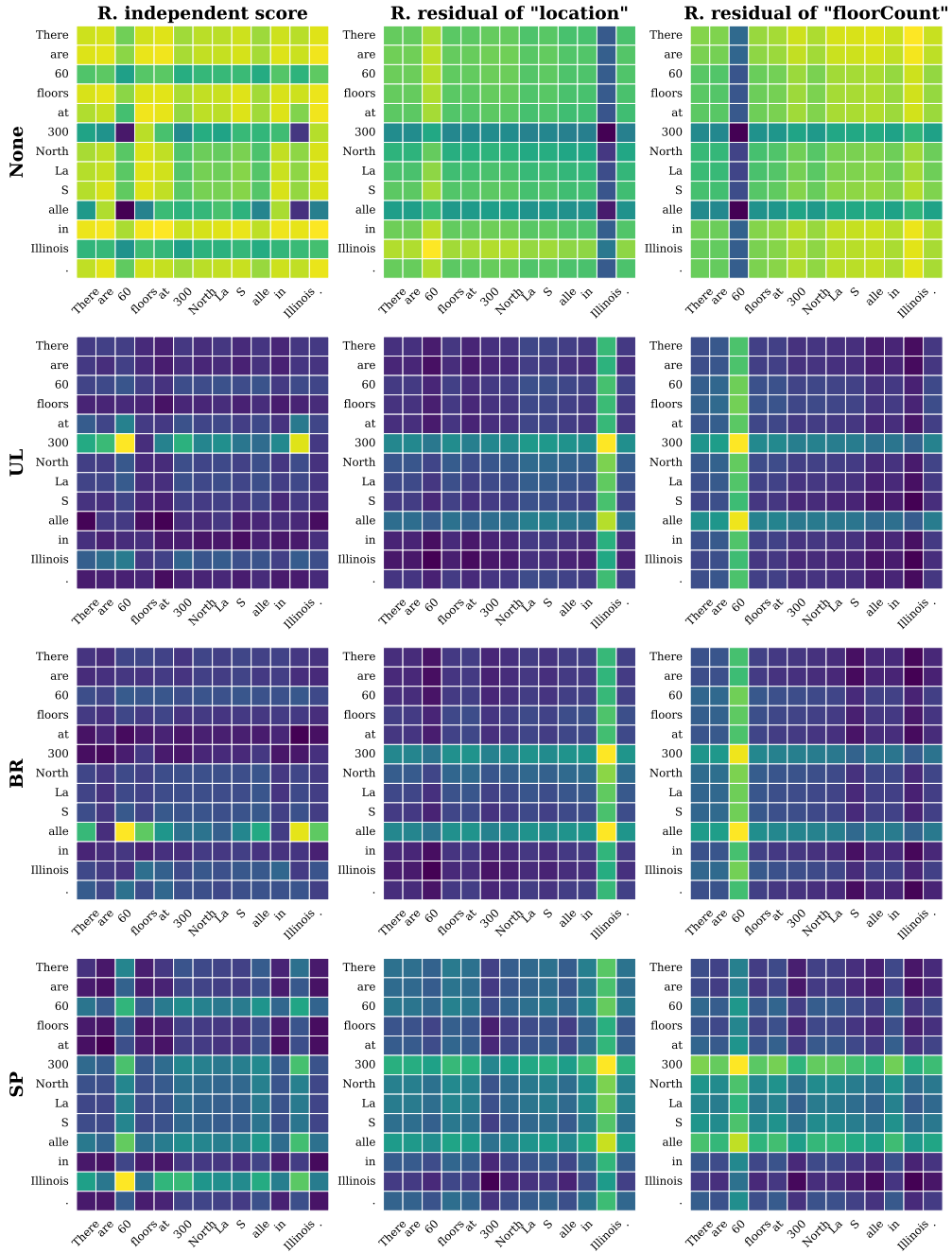


Figure 5: Visualization of Tagging Score. Brighter colors mean larger scores. Each column from left to right is the relation independent score  $z_{ij}^e$ , the relation residual  $z_{ij}^r$  for “location” and “floorCount”, respectively.