Knowledge-Based Systems 219 (2021) 106888

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction

Kang Zhao [a,b], Hua Xu [a,*], Yue Cheng [b], Xiaoteng Li [a,b], Kai Gao [b,*]

[a] State Key Laboratory of Intelligent Technology and Systems, Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

[b] School of Information Science and Engineering, Hebei University of Science and Technology, Shijiazhuang 050018, China

ARTICLE INFO

Article history:
Received 8 November 2020
Received in revised form 12 December 2020
Accepted 20 February 2021
Available online 4 March 2021

Keywords:
Relation extraction
Heterogeneous graph neural networks
Representation learning
Information extraction

ABSTRACT

Joint entity and relation extraction is an essential task in information extraction, which aims to extract all relational triples from unstructured text. However, few existing works consider possible relations information between entities before extracting them, which may lead to the fact that most of the extracted entities cannot constitute valid triples. In this paper, we propose a representation iterative fusion based on heterogeneous graph neural networks for relation extraction (RIFRE). We model relations and words as nodes on the graph and fuse the two types of semantic nodes by the message passing mechanism iteratively to obtain nodes representation that is more suitable for relation extraction tasks. The model performs relation extraction after nodes representation is updated. We evaluate RIFRE on two public relation extraction datasets: NYT and WebNLG. The results show that RIFRE can effectively extract triples and achieve state-of-the-art performance.[1] Moreover, RIFRE is also suitable for the relation classification task, and significantly outperforms the previous methods on SemEval 2010 Task 8 datasets.

1. Introduction

Joint entity and relation extraction is a central task of information extraction, which can automatically construct knowledge from unstructured text. The task is to identify all possible relational triples (subject, predicate, object) in the text, and need to handle the scenario where a sentence contains multiple overlapping relational triples. As shown in Fig. 1, triples share one or two entities in a sentence.

The early research on relational triple extraction is mainly base on the pipeline method [1,2]. It is mainly divided into two subtasks. First, identify the entities in the sentence, and then perform relation classification (RC) for each entity pair. Although these methods are flexible, they are prone to error propagation. Therefore, a joint training method for entities and relations was proposed [3–5], and make significant progress. However, most of the existing methods cannot deal with the scenario correctly, where the sentence contains multiple overlapping relational triples.

Recently, [4] first used the Seq2Seq model with a copy mechanism to alleviate the problem of overlapping triples. On this basis, the influence of extraction order was further studied [6], which was improved by reinforcement learning. [7] proposes an attention-based joint model, which model is devising a supervised multi-head self-attention mechanism as the relation detection module to learn the token-level correlation for each relation type separately. [8] proposes a relation-specific attention network, which utilizes a relation-aware attention mechanism to construct specific sentence representation for each relation and then extract its corresponding entities. [9] first used the BERT-based pre-training model, which model relations as functions that map subjects to objects and using the cascade tagging framework to improve the performance significantly.

For the relation extraction task, we know that the relation between entities is usually triggered by the context of the sentence rather than the target entities. For example, in Fig. 1, '*the capital is*' in the sentence will directly express relation *capital*. Therefore, if the relations information is introduced as a priori knowledge to reduce the extraction of semantically irrelevant entities, which alleviate the redundant extraction of triples. Consequently, we achieve mutual enhance between word representation and relation representation through a method called representation iterative fusion. As shown in Fig. 2, the purpose of iterative fusion is to make the word representation and relation representation contain information related to its, enhance its respective representation capabilities, and make representations more conducive to the relation extraction task.

* Corresponding authors.
E-mail addresses: zhaok7878@gmail.com (K. Zhao), xuhua@tsinghua.edu.cn (H. Xu), chengyue9797@gmail.com (Y. Cheng), lixiaoteng1999@gmail.com (X. Li), gaokai@hebust.edu.cn (K. Gao).

[1] The code will be available at https://github.com/zhao9797/RIFRE.

https://doi.org/10.1016/j.knosys.2021.106888
0950-7051/

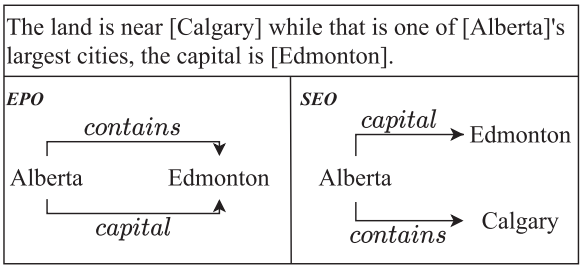


Fig. 1. Examples of EntityPairOverlap (EPO) and SingleEntityOverlap (SEO) overlapping triplets.

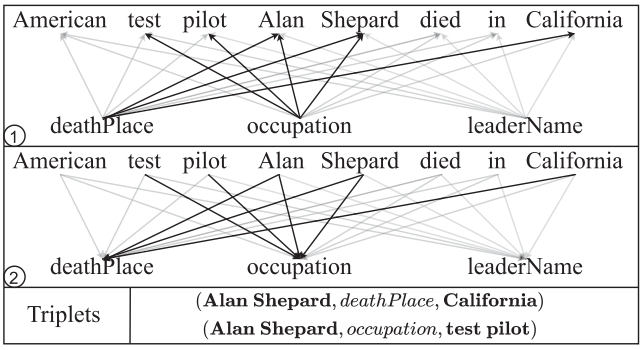


Fig. 2. Example of representation iterative fusion. First, the relational information passed to the word representation, and then the updated word information passed to the relation representation. The darker the lines, the more likely its to form triples between words and relations.

In this paper, we propose a representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. As shown in Fig. 3, we model relations and words as nodes on the graph and update the nodes through a message passing mechanism. The model performs relation extraction after nodes are updated. First, we use the subject tagger to detect all possible subjects on the word nodes. Then, we combine each word node with the candidate subject and relation, and the object tagger is used to tag the object on the new word nodes. The constructed heterogeneous graph manifests the following advantages: (a) Regarding the relations as nodes, each word node integrates specific relation and subject information to tag the object after tagged the subject, which makes it easy to handle overlapping triples. (b) Different nodes can make full use of each other through multiple message transfer processes. Before extracting entities, each word fuse the semantic information of the relations nodes that may be associated with it. After that, taggers can easy to extract the entities that form valid relations.

The contributions of this paper can be summarized as follows.

- We propose a representation iterative fusion strategy, which can effectively learn a suitable representation for relation extraction tasks.
- To our knowledge, we are the first to construct a heterogeneous graph neural network for relation extraction task, which contains relation nodes and word nodes.
- Our proposed method can effectively extract relational triples and is easily extended to relation classification tasks.
- Experiments show that our method achieves state-of-the-art results on three public datasets.

The rest of this paper is arranged as follows. In Section 2, we review previous work in relation extraction, heterogeneous graph network. In Section 3, we describe the task definition. In Section 4, we introduce the proposed method in detail. In Section 5, we describe the experimental steps and the experimental results. Finally, we demonstrate our conclusions and prospects for future work in Section 6.

2. Related works

2.1. Relation extraction

Joint entity and relation extraction are critical topics in NLP, and researchers have explored many methods to extract triples. The traditional method [1,2] generally uses the pipeline method to handle this task, which is divided into two subtasks, firstly identifying entities, and then classifying the relations between entities. These methods inevitably bring about the problem of error propagation. In order to alleviate the problem of error propagation, [10,11] shared parameters to achieve joint learning of entities and relations. [3] proposes a novel tagging scheme to jointly extract entities and relations, which transform the extraction problem into a tagging task. [12] proposed a relation extraction model based on an improved graph convolutional network, which first extracts entity spans, and then use a relation-aware attention mechanism to obtain the relation between entities.

However, most of the past methods cannot correctly handle the relational triplet scenes containing multiple overlapping entities in the sentence. Therefore, some recent methods are dedicated to solving this problem. [4] proposed a sequence-to-sequence model with a copy mechanism to solve the problem of triple overlap. [5] proposed a graph convolutional network method and also studied the problem of triple overlap. [13] proposes an end-to-end model with a double pointer module to improve the performance of relation extraction, which can jointly extract whole entities and relations. [9] proposed a novel cascade tagging framework and model the relations as functions, which significantly improved the problem.

2.2. Heterogeneous graph network for NLP

Graph neural network [14,15] are initially designed for the entire graph using one type of isomorphic graph. However, graphs in practical applications are usually heterogeneous graphs with multiple types of nodes. Recent works have made preliminary explorations. [16] introduced a heterogeneous document-entity graph for multi-hop reading comprehension, which contains different granularity levels of information. [17] built a heterogeneous graph with entity nodes and sentence nodes for the few-shot relation classification task. [18] proposed a neural network for abstract extraction based on heterogeneous graphs, which contains semantic nodes with different granularity levels except sentences.

For relation extraction tasks, there is a lack of research on heterogeneous graph neural networks. To realize the iterative representation fusion strategy, we propose to construct heterogeneous graphs by treating relations as nodes on the graph and apply heterogeneous graph neural networks to obtain better representation for relation extraction tasks.

3. Task definition

In the joint entity and relation extraction task, the goal is to identify all possible triples (subject, relation, object) in a sentence. Following the previous work [9], we directly design a training objective right at the triple level.

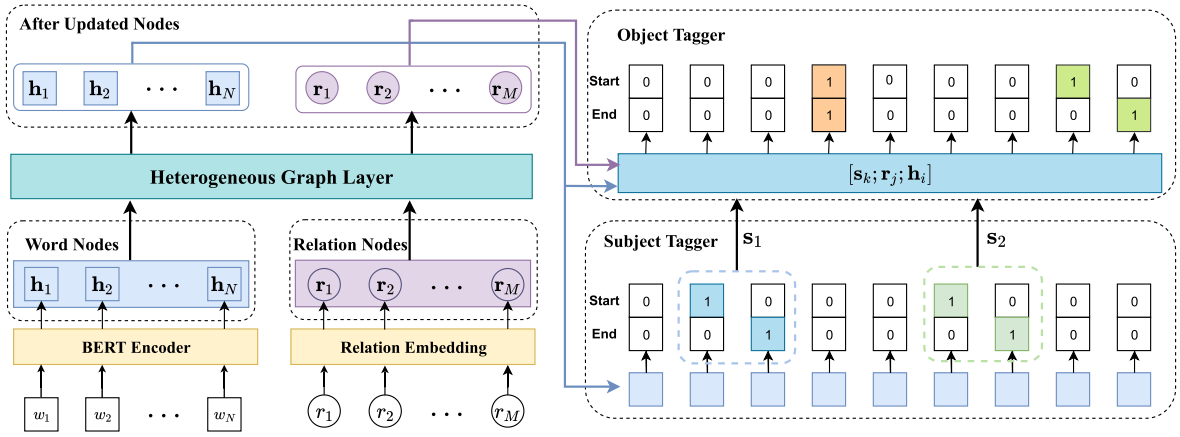


Fig. 3. Architectures of our proposed RIFRE for joint entity and relation extraction task.

Given an annotated sentence x and a set of triples $T = \{(s, r, o)\}$ in s, our goal is to maximize the data likelihood of all sentences in the training set, which is defined as follows:

$$\begin{aligned} &\prod_{(s,r,o)\in T} p\left((s, r, o) \mid x\right) \\ =&\prod_{s\in T} p\left(s \mid x\right) \prod_{(r,o)\in T|s} p\left((r, o) \mid x, s\right) \\ =&\prod_{s\in T} p\left(s \mid x\right) \prod_{r\in T|s} p\left(o \mid x, s, r\right) \prod_{r\in R\setminus T|s} p\left(o_{\varnothing} \mid x, s, r\right) \end{aligned} \tag{1}$$

where $s \in T$ represents the subject in the triple T. $T \mid s$ is the set of triples with s as the subject in T. $(r, o) \in T \mid s$ is a (r, o) pair in $T \mid s$. R is the set of all relations in the training set, $R\backslash T \mid s$ represents all relations except s as the subject in T. $o_{\varnothing}$ is "null" object, which indicate that all other relations except those contained in the triplets $T \mid s$ will have no corresponding objects in the sentence s.

4. Methodology

In this section, we will detail the overall framework of the Representation Iterative Fusion Based on Heterogeneous Graph Neural Networks for Joint Entity and Relation Extraction(RIFRE). The framework of our proposed RIFRE shown in Fig. 3, which consists mainly of three parts:

- **Vector Representations of Nodes** Given a sentence and a predefined relation type, we construct the input of the graph model by encoding the words in the sentence into vectors and embedding each relation as a vector.
- **Heterogeneous Graph Layer** We propose a heterogeneous graph neural network to iterative fuse the representation of word nodes and relation nodes.
- **Relation Extraction** After obtaining the representation of the word node and the relation node, we perform specific relation extraction steps.

4.1. Vector representations of nodes

We construct two types of semantic nodes for the proposed heterogeneous graph: word nodes and relation nodes.

4.1.1. Word node

Given a sentence x in the training set, we use the pre-trained model BERT [19] to encode the context information. We take all token embeddings in the last hidden layer of BERT as word nodes:

$$[\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N] = \mathbf{B}\left([w_1, w_2, \ldots, w_N]\right) \tag{2}$$

where w_i is the one-hot vectors of sub-word[2] in sentence, N is the sequence length, $\mathbf{B}$ is the pre-trained language model BERT, and $\mathbf{h}_i \in \mathbb{R}^{d_h}$ is the hidden vector of w_i after context encoding. We use $\mathbf{h}_i$ as the initial word node.

4.1.2. Relation node

We embed each predefined relation label as a high-dimensional vector and pass through a linear mapping layer after embedding:

$$[\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_M] = \mathbf{W}_r\mathbf{E}\left([r_1, r_2, \ldots, r_M]\right) + \mathbf{b}_r \tag{3}$$

where $M = |R|$ is the number of predefined relations, r_i is the one-hot vectors of relation indices in the predefined relations, $\mathbf{r}_i \in \mathbb{R}^{d_h}$ is the vector after embedding and mapping, $\mathbf{E}$ is the relation embedding matrix, $\mathbf{W}_r$ and $\mathbf{b}_r$ is the trainable parameters. We use $\mathbf{r}_i$ as the initial relation node.

4.2. Heterogeneous graph layer

Given two types of semantic nodes representation: $\{\mathbf{u}_i\}_{i=1}^{N}$ and $\{\mathbf{v}_j\}_{j=1}^{M}$, we regard all nodes of one type as neighbors of each other type of node. Then, we update the node representations by the message passing mechanism, similar to the graph attention network [14]:

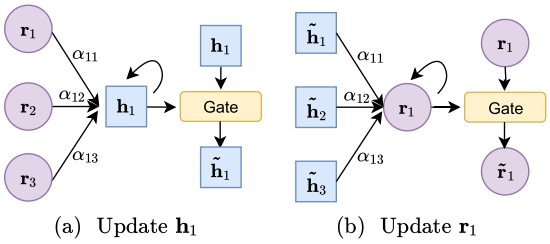
$$\begin{aligned} a_{ij} &= \mathbf{W}_a\left[\mathbf{W}_q\mathbf{u}_i; \mathbf{W}_k\mathbf{v}_j\right] \\ \alpha_{ij} &= \frac{\exp\left(a_{ij}\right)}{\sum_{l\in\mathcal{N}_i}\exp\left(a_{il}\right)} \\ \mathbf{u}_i' &= \mathbf{u}_i + \sum_{j\in\mathcal{N}_i}\alpha_{ij}\mathbf{W}_v\mathbf{v}_j \end{aligned} \tag{4}$$

where $[\,\cdot\,;\,\cdot\,]$ means concatenating representation of two vectors, $\mathbf{W}_a, \mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$ are trainable weights and α_{ij} is the attention weight between $\mathbf{u}_i \in \mathbb{R}^{d_h}$ and $\mathbf{v}_j \in \mathbb{R}^{d_h}$.

We use a gate mechanism instead of the activation function, which can maintain the scale of each dimension and maintain the non-linear capability, which is as follows,

$$\begin{aligned} g_i &= \text{sigmoid}\left(\mathbf{W}_g\left[\mathbf{u}_i; \mathbf{u}_i'\right]\right) \\ \tilde{\mathbf{u}}_i &= g_i \odot \mathbf{u}_i' + (1 - g_i) \odot \mathbf{u}_i \end{aligned} \tag{5}$$

[2] We split the input sentence use the WordPiece [20] method, each word in the input sentence will be tokenized to finegrained tokens, i.e., sub-words.



(a) Update $\mathbf{h}_1$ (b) Update $\mathbf{r}_1$

Fig. 4. The detailed update process of word nodes and relation nodes.

Table 1
Statistics of datasets. Note that a sentence can belong to both the EPO class and SEO class. SemEval refers to the SemEval 2010 Task 8 dataset. SemEval 2010 Task 8 does not establish a default split for development, we use a random slice of the training set with 6500 examples.

Category	NYT		WebNLG		SemEval	
	Train	Test	Train	Test	Train	Test
Normal	37013	3266	1596	246	6500	2717
EPO	9782	978	227	26	–	–
SEO	14735	1297	3406	457	–	–
ALL	56195	5000	5019	703	6500	2717

where $\mathbf{W}_g$ is trainable weights, g_i is a scalar value, $\tilde{\mathbf{u}}_i$ is the final output, and $\odot$ is element-wise production.

To simplify, we define the above formula as follows,

$$\tilde{\mathbf{u}}_i = \mathbf{GNN}\left(\mathbf{u}_i, \{\mathbf{v}_j\}_{j\in\mathcal{N}_i}\right) \tag{6}$$

where $\{\mathbf{v}_j\}_{j\in\mathcal{N}_i}$ is the set of all neighbors of node $\mathbf{u}_i$, $\tilde{\mathbf{u}}_i \in \mathbb{R}^{d_h}$ is the new node representation of the output, and **GNN** denotes the process of updating node $\mathbf{u}_i$ using Eqs. (4) and (5).

4.2.1. Nodes updating with iterative fusion

To perform semantic fusion between word nodes and relation nodes, we define an iterative message passing process. First, we use all the relation nodes as neighbors of each word nodes to update the word nodes through **GNN**:

$$\tilde{\mathbf{h}}_i^1 = \mathbf{GNN}\left(\mathbf{h}_i^0, \{\mathbf{r}_j^0\}_{j\in\mathcal{N}_i}\right) \tag{7}$$

where $\tilde{\mathbf{h}}_i^1 \in \mathbb{R}^{d_h}$ is the updated representation of the initial word node $\mathbf{h}_i^0 \in \mathbb{R}^{d_h}$. After that, we add a residual connection to avoid gradient vanishing during training. The final output of the updated node is as follows,

$$\mathbf{h}_i^1 = \tilde{\mathbf{h}}_i^1 + \mathbf{h}_i^0 \tag{8}$$

When we obtain the representation of the new word node, we update the relation node according to the new representation. Each layer contains the word node update and relation node update. The update process of the lth layer can be represented as:

$$\begin{aligned}
\tilde{\mathbf{h}}_i^{l+1} &= \mathbf{GNN}\left(\mathbf{h}_i^l, \{\mathbf{r}_j^l\}_{j\in\mathcal{N}_i}\right)\\
\mathbf{h}_i^{l+1} &= \tilde{\mathbf{h}}_i^{l+1} + \mathbf{h}_i^l\\
\tilde{\mathbf{r}}_j^{l+1} &= \mathbf{GNN}\left(\mathbf{r}_j^l, \{\mathbf{h}_i^{l+1}\}_{i\in\mathcal{N}_j}\right)\\
\mathbf{r}_j^{l+1} &= \tilde{\mathbf{r}}_j^{l+1} + \mathbf{r}_j^l
\end{aligned} \tag{9}$$

As Fig. 4 shows, word nodes aggregate all the relational information and utilize the updated word node representation to update the relational nodes so that the representation of nodes more suitable for the specific tasks through iterative updating.

4.3. Relation extraction

We use it for the joint entity and relation extraction task after obtaining the final representations of word nodes and relation nodes. Here, we also briefly introduce how to apply our method to the relation classification task.

4.3.1. Joint entity and relation extraction task

Similar to the previous work [9,21]. As shown on the right side of Fig. 3, first, we use the **Subject Tagger** to detect all possible subjects in the word nodes. Specifically, two binary classifiers detect the begin and end positions of the subject in the word node. Formally, the start and end position tags of subjects are predicted as follows,

$$\begin{aligned}
p_i^{sta_s} &= \text{sigmoid}\left(\mathbf{W}_{sta_s}\,\text{Tanh}\left(\mathbf{h}_i^o\right) + \mathbf{b}_{sta_s}\right)\\
p_i^{end_s} &= \text{sigmoid}\left(\mathbf{W}_{end_s}\,\text{Tanh}\left(\mathbf{h}_i^o\right) + \mathbf{b}_{end_s}\right)
\end{aligned} \tag{10}$$

where $\mathbf{h}_i^o \in \mathbb{R}^{d_h}$ is the output of the last layer of heterogeneous graph network layer, Tanh is the activation function, $\mathbf{W}_{sta\_s}$, $\mathbf{b}_{sta\_s}$, $\mathbf{W}_{end\_s}$ and $\mathbf{b}_{end\_s}$ is trainable weights, $p_i^{sta\_s}$ and $p_i^{end\_s}$ represent the probability of identifying the ith word node as the start and end position of a subject respectively. The subject tagger optimizes the likelihood function below to determine the span of subject s of in sentence x:

$$\begin{aligned}
&p_{\theta_s}(s \mid x)\\
&= \prod_{t\in\{sta_s, end_s\}} \prod_{i=1}^{N} \left(p_i^t\right)^{\mathrm{I}\{y_i^t=1\}} \left(1-p_i^t\right)^{\mathrm{I}\{y_i^t=0\}}
\end{aligned} \tag{11}$$

where θ_s is the parameter of the subject tagger. $\mathrm{I}\{z\} = 1$ if z is true and 0 otherwise. $y_i^{sta\_s}$ and $y_i^{end\_s}$ is the binary tag of subject start and end position for the ith word in x, respectively.

To extract the triples completely, the input of **Object Tagger** is different from the input node representation of the subject tagger. We further combine each word node, candidate subject, and each relation node to prepare for the next step of detecting the object, which as follows,

$$\mathbf{h}'_{ijk} = \text{Tanh}\left(\mathbf{W}_h\left[\mathbf{s}_k; \mathbf{r}_j^o; \mathbf{h}_i^o\right] + \mathbf{b}_h\right) \tag{12}$$

where $\mathbf{h}'_{ijk} \in \mathbb{R}^{d_h}$ is the word node representation that combines relation and subject, $\mathbf{s}_k$ is the representation vector of the kth candidate subject, r_j^o is the relation node output in the last layer of the graph neural network, $\mathbf{W}_h$ and $\mathbf{b}_h$ is trainable weights. Analogously, we predict the start and end labels of the object as follows,

$$\begin{aligned}
p_i^{sta_o} &= \text{sigmoid}\left(\mathbf{W}_{sta_o}\mathbf{h}'_{ijk} + \mathbf{b}_{sta_o}\right)\\
p_i^{end_o} &= \text{sigmoid}\left(\mathbf{W}_{end_o}\mathbf{h}'_{ijk} + \mathbf{b}_{end_o}\right)
\end{aligned} \tag{13}$$

where $\mathbf{W}_{sta\_o}$, $\mathbf{b}_{sta\_o}$, $\mathbf{W}_{end\_o}$ and $\mathbf{b}_{end\_o}$ are trainable weights, $p_i^{sta\_o}$ and $p_i^{end\_o}$ represent the probability of identifying the ith word node as the start and end position of a object respectively. The object tagger optimizes the likelihood function below to determine the object's span o given a sentence x, a subject s, and a relation r:

$$\begin{aligned}
&p_{\theta_o}(o \mid x, s, r)\\
&= \prod_{t\in\{sta_o, end_o\}} \prod_{i=1}^{N} \left(p_i^t\right)^{\mathrm{I}\{y_i^t=1\}} \left(1-p_i^t\right)^{\mathrm{I}\{y_i^t=0\}}
\end{aligned} \tag{14}$$

where θ_o is the parameter of the object tagger. $y_i^{sta\_o}$ and $y_i^{end\_o}$ is the binary tag of object start and end position for the ith word in x, respectively. For a "null" object $o_\varnothing$, the tags $y_i^{sta\_o_\varnothing} = y_i^{end\_o_\varnothing} = 0$ for all i.

Table 2
Results of different methods on NYT and WebNLG datasets. Our re-implementation is marked by *. The method marked with ∅ means unused heterogeneous graph layers.

Method	NYT			WebNLG		
	Prec.	Rec.	F1	Prec.	Rec.	F1
NovelTagging [3]	62.4	31.7	42.0	52.5	19.3	28.3
$\text{CopyR}_{OneDecoder}$ [4]	59.4	53.1	56.0	32.2	28.9	30.5
$\text{CopyR}_{MultiDecoder}$ [4]	61.0	56.6	58.7	37.7	36.4	37.1
GraphRel_{1p} [5]	62.9	57.3	60.0	42.3	39.2	40.7
GraphRel_{2p} [5]	63.9	60.0	61.9	44.7	41.1	42.9
SPointer [13]	72.8	69.0	70.9	38.7	37.5	38.1
CopyR_{RL} [6]	77.9	67.2	72.1	63.3	59.9	61.6
Relation-Aware [12]	83.2	64.7	72.8	66.4	62.7	64.5
CasRel [9]	89.7	89.5	89.6	**93.4**	90.1	91.8
CasRel*	88.3	90.2	89.2	90.9	91.8	91.3
$\text{RIFRE}_{\varnothing}$	90.0	89.1	89.5	91.8	91.5	91.6
RIFRE	**93.6**	**90.5**	**92.0**	93.3	**92.0**	**92.6**

Table 3
Results of different methods on SemEval 2010 Task 8 datasets. Our re-implementation is marked by *.

Method	F1
Att-Pooling-CNN [22]	88.0
KnowBert-W+W [23]	89.1
BERT_{EM}+MTB [24]	89.5
R-BERT [25]	89.3
R-BERT*	88.9
RIFRE	**91.3**

According to Eq. (1), we can obtain the following objective function:

$$\begin{aligned}\mathcal{L} &= \log \prod_{(s,r,o)\in T} p\left((s,r,o) \mid x\right) \\ &= \sum_{s\in T_j} \log p_{\theta_s}\left(s \mid x\right) + \sum_{r\in T_j|s} \log p_{\theta_o}\left(o \mid x,s,r\right) + \\ &\quad \sum_{r\in R\setminus T_j|s} \log p_{\theta_o}\left(o_{\varnothing} \mid x,s,r\right)\end{aligned} \tag{15}$$

To train the model, we use stochastic gradient descent (SGD) to maximize the loss function $\mathcal{L}$.

4.3.2. Relation classification task

For the relation classification task, given a sentence x with the annotated entities pairs (s, o), we aim to identify the relations between s and o. We use the updated word nodes and relation nodes to classify relations. First, the average vector between the start and end positions of each entity is used as the entity representation, and then it concatenates the representation of the relation node for classification, which as follows,

$$\begin{aligned} p_j^r &= \text{sigmoid}\left(\mathbf{MLP}([\mathbf{r}_j^o; \mathbf{s}; \mathbf{o}])\right) \\ p_{\theta_r}\left(r \mid x,s,o\right) &= \prod_{j=1}^{M} \left(p_j^r\right)^{\mathbb{1}\left\{y_j^r=1\right\}} \left(1-p_j^r\right)^{\mathbb{1}\left\{y_j^r=0\right\}} \end{aligned} \tag{16}$$

where $\mathbf{s}$ and $\mathbf{o}$ are the representation vectors of the entities pairs (s, o), **MLP** is the multilayer perceptron and y_j^r is the binary tag of relation for the jth relation in R, θ_r is the parameter of the relation selector. Model training by maximize the log-probability $J = \log p_{\theta_r}(r \mid x, s, o)$ of the true class r_j by SGD. We will take the maximum value of all relations predicted probability values as the predicted label in the prediction phase.

5. Experiments

In this section, we will demonstrate the effectiveness of our proposed method on two public relation extraction datasets. To illustrate the potential of representation iterative fusion strategy, we also conduct supplementary experiments on the relation classification dataset.

5.1. Dataset and evaluation metrics

We evaluate our model on two publicly available datasets NYT [26] and WebNLG [27]. We use the datasets released by [4], in which the NYT dataset is composed of 24 relations, including 56195 train sentences, 5000 validation sentences, and 5000 test sentences. WebNLG includes 5019 sentence training, 500 validation sentences, and 703 test sentences. According to the overlap type of triples, we divide sentences into three categories: Normal, EntityPairOverlap (EPO), and SingleEntityOverlap (SEO). We will discuss this result in a detailed experiment. We also evaluate our method's performance for relation classification task on SemEval 2010 Task 8 [28] datasets. SemEval 10 Task 8 is a public dataset, which contains 10,717 instances with 9 relations and a special "other" class. The statistical information of the datasets is shown in Table 1.

Following the previous work [5,9], we report the standard micro Precision (Prec.), Recall (Rec.), and F1-score as in line with baselines. If and only if the relation and the heads of the two corresponding entities are correct, the predicted triplet is considered correct. We evaluate our model for the relation classification task by using the SemEval-2010 Task 8 official scorer script. It computes the macro averaged F1-scores for the nine actual relations (excluding Other) and considers directionality.

5.2. Training details and parameters setting

For relational triple extraction, we build our model on top of the pre-trained BERT model (base-cased, with 12-layer transformer) implemented in PyTorch [29] and adopt its default hyperparameter settings. We use the SGD optimizer to optimize our model, the training batch size is 6, and the learning rate is 0.1. To prevent the model from over-fitting, we stop the training process when the performance on the validation set does not gain any improvement for at least 9 consecutive epochs. We set the maximum number of words in the input sentence to 100. We set a threshold as 0.5 to determine the start and end tags of the word in the training stage. For the relation classification task, we build our model on top of the pre-trained BERT model (base-uncased, with 12-layer transformer). We employ the same setup as for relational triple extraction, except for the number of relations. All the hyper-parameters are determined on the validation set.

5.3. Comparison with previous work

For triple extraction task, we compare the RIFRE with several strong state-of-the-art models, namely, **NovelTagging** [3], **CopyRE** [4], **GraphRel** [5], **SPointer** [13], $\textbf{CopyR}_{\text{RL}}$ [6] **Relation-Aware** [12] and **CasRel** [9]. The reported results for the above baselines are directly copied from the original published literature. Note that the result for NovelTagging [3] on the WebNLG dataset is obtained from [4]. For CasRel* is our reproduced model. Note that CasRel builds model on the top of the pre-trained $\text{BERT}_{\text{BASE}}$ model. The $\text{RIFRE}_{\varnothing}$ denote using only the initial nodes for relation extraction.

For the RC task, we compare our method with **Att-Pooling-CNN** [22], **KnowBert-W+W** [23], $\textbf{BERT}_{\textbf{EM}}$**+MTB** [24], **R-BERT** [25]. The reported results for the above baselines are directly copied

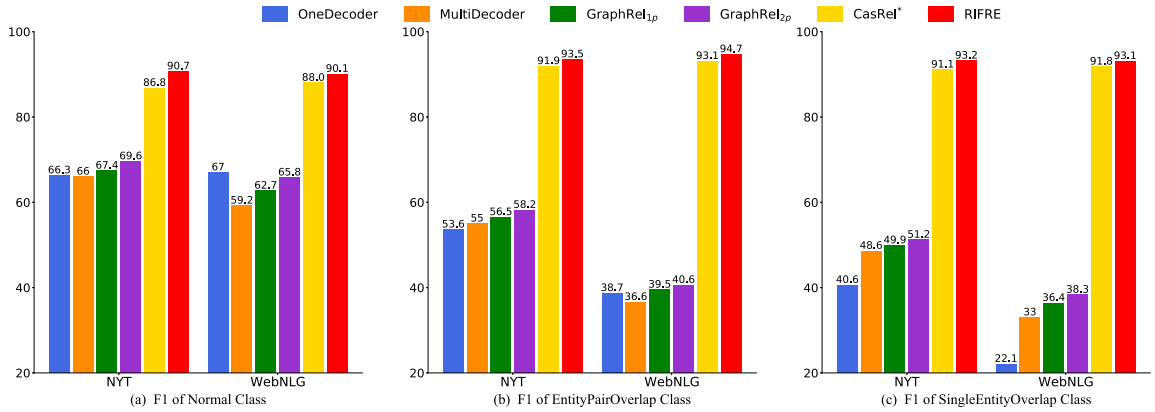


Fig. 5. F1-score (%) of extracting triples from sentences with different overlapping type.

Table 4
F1-score (%) of extracting relational triples from sentences with different number (denoted as N) of triples.

Method	NYT					WebNLG				
	$N=1$	$N=2$	$N=3$	$N=4$	$N\geq 5$	$N=1$	$N=2$	$N=3$	$N=4$	$N\geq 5$
$\text{CopyR}_{OneDecoder}$	66.6	52.6	49.7	48.7	20.3	65.2	33.0	22.2	14.2	13.2
$\text{CopyR}_{MultiDecoder}$	67.1	58.6	52.0	53.6	30.0	59.2	42.5	31.7	24.2	30.0
GraphRel_{1p}	69.1	59.5	54.4	53.9	37.5	63.8	46.3	34.7	30.8	29.4
GraphRel_{2p}	71.0	61.5	57.4	55.1	41.1	66.0	48.3	37.0	32.1	32.1
CasRel*	87.7	90.1	91.9	93.5	83.4	88.0	90.5	93.2	92.6	90.7
RIFRE	**90.7**	**92.8**	**93.4**	**94.8**	**89.6**	**90.2**	**92.0**	**94.8**	**93.0**	**92.0**

from the original published literature. Note that BERT_{EM}+MTB and R-BERT is builds model on the top of the pre-trained $\text{BERT}_{\text{LARGE}}$ model.

Table 2 shows the results of different methods for the joint entity and relation extraction on NYT and WebNLG datasets. This result indicates that our method outperforms all other methods and obtains state-of-the-art performance on two datasets. The comparison between RIFRE and $\text{RIFRE}_{\varnothing}$ indicates that the updated node representation can greatly enhance the ability of the model to extract triples. For the NYT dataset, our model achieves a 2.8% improvement in F1-score over the best method and consistently improved in Precision and Recall. For the WebNLG dataset, our model 1.3% increased in F1-score over the previous best model. As shown in Table 1, because the WebNLG dataset holds a higher proportion of overlapping triples than the NYT dataset, it is more difficult on the WebNLG dataset. However, our method can still improve the recall rate and maintain a high precision rate, proving that our method can effectively deal with the overlapping problem.

Table 3 shows the results of different methods for relation classification on the SemEval 2010 Task 8 dataset. We can see that RIFRE significantly exceeds all the baseline methods on SemEval datasets. The macro-F1 value is 91.3%, which is much better than the previous best solution on this dataset. These results show that the representation iterative fusion strategy can be well applied to relation classification tasks.

5.4. Detailed results on different types of sentences

To further analyze the RIFRE capability to extract overlapping triples, we conduct two expansion experiments on different types of sentences and compare the performance with previous work.

First, we divide the test sentences in NYT and WebNLG into three categories based on different overlapping types: Normal, EntityPairOverlap, and SingleEntityOverlap. Then we test the performance of RIFRE in each category. The result is shown in Fig. 5. We can see that the RIFRE outperforms all other methods under different overlapping types. We can also observe that the performance of RIFRE and CasRel* far exceeds the first four methods because these methods base on pre-trained language models. This experiment shows that our method can effectively extract overlapping triples.

Table 5
Ablation studies result on WebNLG test set. '-' means we remove or change the module from the original RIFRE.

Method	Prec.	Rec.	F1
RIFRE	93.3	92.0	92.6
-gate mechanism	92.8	91.3	92.0
-residual connection	92.7	91.9	92.3
-word node update	92.3	91.4	91.8
-relation node update	92.2	91.6	91.9

Table 6
Result of different numbers of heterogeneous graph network layers. The experiments are performed on the validation set of NYT and WebNLG. Time is the total time of 5 epochs.

Number	NYT		WebNLG	
	F1	Time	F1	Time
$l=0$	89.4	1.71 h	78.2	0.24 h
$l=1$	90.3	1.94 h	83.7	0.31 h
$l=2$	90.5	2.16 h	85.4	0.37 h
$l=3$	90.1	2.46 h	86.8	0.42 h

We also validate the RIFRE capability to extract multiple triples from a sentence. We divide test sentences into five classes, which indicate its number of triples is 1, 2, 3, 4, and $\geq$ 5. Table 4 shows the results, and we can see that RIFRE achieves the best performance under all classes settings. Besides, The RIFRE gains the most significant improvement over the best state-of-the-art method under the most challenging settings($N \geq 5$)), which shows that our method is easy to handle complex scenarios.

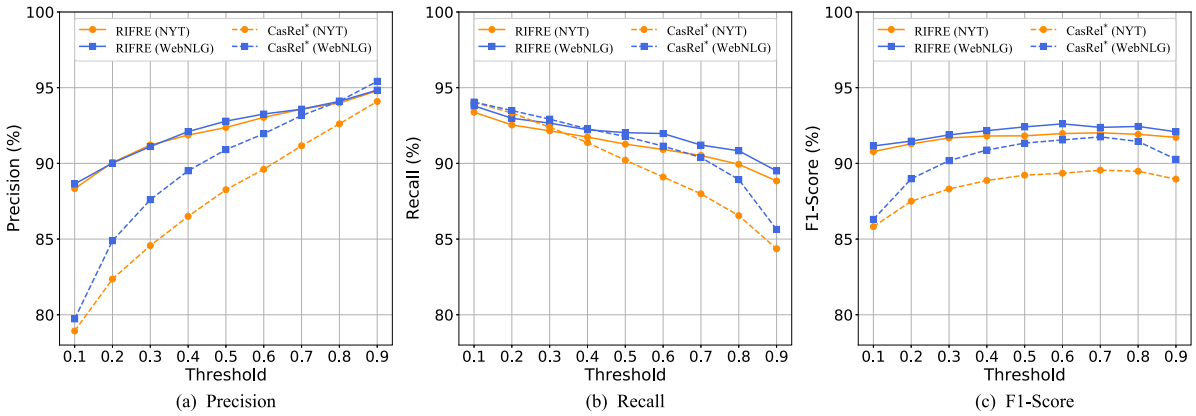


Fig. 6. Results on different tag thresholds.

Table 7
Result on relational triple elements.

Element	NYT			WebNLG		
	Prec	Rec	F1	Prec	Rec	F1
E1	95.9	93.2	94.5	97.8	94.7	96.2
E2	95.8	93.6	94.7	97.2	94.5	95.8
R	97.4	93.7	95.5	96.1	93.4	94.7
(E1, R)	95.5	91.9	93.6	94.5	92.4	93.5
(R, E2)	95.2	91.7	93.4	95.1	92.8	93.9
(E1, E2)	93.3	90.9	92.1	95.1	93.7	94.4
(E1, R, E2)	93.6	90.5	92.0	93.3	92.0	92.6

5.5. Analysis and discussion

5.5.1. Ablation study

We examine the contributions of different modules components in the Heterogeneous Graph Layer, using the best performing model on the WebNLG dataset. First, we replace the gate mechanism with the tanh activation function. We also study the impact of residual connection, word node update, and relation node update. Table 5 shows the results. We observe that using the gate mechanism and residual connection can improve the performance of the model. Results show that these two components help RIFRE learn better node representations, and the gate mechanism seems to be playing a more significant role. After removing the word node update, the value of F1 drops to 91.8%, which shows the importance of fusing relation representation and word representation. Besides, the residual connection and relation node update for the model also show their effectiveness.

5.5.2. Number of heterogeneous graph layers

To confirm the number of layers of the graph neural network, we studied the results of using different numbers of heterogeneous graph network layers on the validation set of NYT and WebNLG. All the models have trained 5 epochs. Table 6 presents the results of different layers, and we can observe that RIFRE has comparable results for $l = 2$ and $l = 3$. To balance the time cost and model performance, we set $l = 2$ for all tasks.

5.5.3. Analysis of tag threshold

We also explore the impact of different tag thresholds, where the same threshold is set for both the subject tagger and the object tagger. Fig. 6 reports the results of our method and the best state-of-the-art method CasRel [9] under different tag threshold settings. As the threshold increases, the precision of all the models gradually decreases, while the recall gradually increases. The F1-score first rises and then falls, but the F1-score of our method is relatively stable and outperforms the CasRel methods. The overall results show that the threshold can affect the number of entities extracted. As the threshold increases, the number of extracted triples also increases but the precision will decrease. The RIFRE achieved the best performances on WebNLG and NYT datasets with thresholds of 0.6 and 0.7. The performance of our model on the NYT dataset is relatively low because the size of the WebNLG dataset and the NYT dataset is nearly 10 times different. In general, our model has high confidence for relational triple extract. Even with a threshold of 0.1, the F1-score of our method is more than 90%.

5.5.4. Error analysis

To explore the factors that affect the extracted relational triples of the RIFRE model, we analyze the predictive performance of different elements in the triple(E1, R, E2) where E1 represents the subject entity, E2 represents the object entity, and R represents the relation between them. An element like (E1, E2) is regarded as correct only if the subject and the object in the predicted triple (E1, R, E2) and correct, regardless of the correctness of the predicted relation. Similarly, we say an instance of E2 is correct as long as the extracted triple is correct, so is E1 and R.

Table 7 shows the results on different relational triple elements. For the NYT dataset, the performance of E1 and E2 are consistent with that on (E1, R) and (R, E2), demonstrate the effectiveness of our proposed method in identifying both subject and object entity mentions. We can also find a tiny gap between the F1-score on (E1, R, E2) and (E1, E2), while there is a significant gap between (E1, R, E2) and (E1, R)/(R, E2). This result shows that when most entity pairs are correctly identified, the relation will be recognized correctly. It will be more accessible to identifying the relations than to identifying entities for our model. For the WebNLG dataset, the performance there is an apparent gap between (E1, R, E2) and (E1, E2), while (E1, R, E2) and (E1, R)/(R, E2) with a small gap. This is contrary to the NYT dataset results, indicating that the misidentifying model relation can lead to model performance degradation more than misidentifying the entities. We attribute such difference to the different number of relations contained in the two datasets (i.e., 24 in NYT and 246 inWebNLG), which makes the identification of relation much harder in WebNLG.

5.5.5. Relation nodes visualization

We use the 17 relations nodes vector learned on the NYT dataset for visualization analysis. Specifically, we take the after trained model's initial relation nodes' vector representation to do hierarchical clustering to analyze the association between the

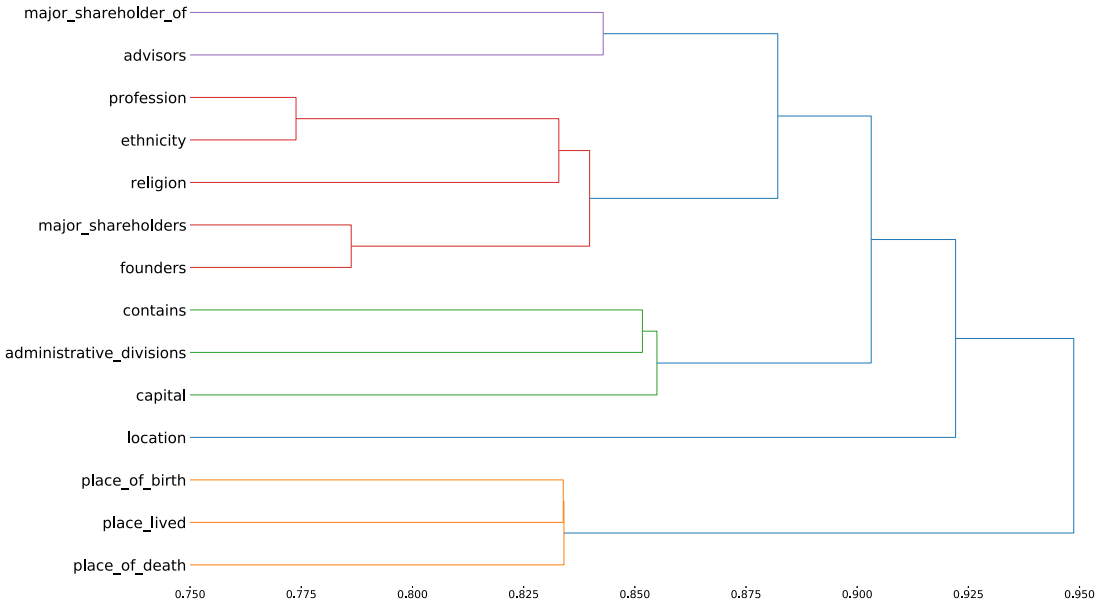


Fig. 7. Hierarchical clustering of relation nodes. The ordinate label is the name of each relation node, and the abscissa scale refers to the cosine distance.

Table 8
Case study for RIFRE$_{\varnothing}$ and RIFRE. Here different entities of the triples are shown in different colors.

Sentence	RIFRE$_{\varnothing}$	RIFRE
An article on June 18 about Amsterdam's celebration of the 400th anniversary of Rembrandt's birth, and an accompanying map, misspelled the names of churches that have ties to him.	(Rembrandt, place_of_death, Amsterdam) (Rembrandt, place_lived, Amsterdam)	(Rembrandt, place_of_death, Amsterdam) (Rembrandt, place_lived, Amsterdam)
Footnotes The Philadelphia Orchestra will perform for the first time in Singapore and Kyoto, Japan, when its music director, Christoph Eschenbach, below, leads it on a 15-concert tour of Asia from May 19 through June 7.	(Asia, contains, Singapore) (Asia, contains, Japan)	(Asia, contains, Singapore) (Asia, contains, Japan) (Japan, contains, Kyoto)
Another team, on the business side, worked on a particularly important deal to us, said Google's co-founder and co-president, Sergey Brin.	(Sergey Brin, company, Google) (Google, major_shareholders, Sergey Brin) (Google, founders, Sergey Brin)	(Sergey Brin, major_shareholder_of, Google) (Sergey Brin, company, Google) (Google, major_shareholders, Sergey Brin) (Google, founders, Sergey Brin)

different learned relation nodes. Fig. 7 shows the result of the hierarchical clustering of relation nodes. We can observe that *place_of _death*, *place_lived*, and *place_of _birth* can be regarded as one category under a threshold. Similarly, *contains*, *capital* and *administrative_divisions* both represent a broader '*contains*' relation. Intuitively, these three relations can be represented as a more coarse-grained relation. This shows that our method can learn the connections between relations. In practice, there are complex relationships with different levels between relations. Current relation extraction models cannot capture this relationship well. Our proposed method can not only extract triples well but also learn different granular information between relations implicitly.

5.5.6. Case study

Table 8 shows the case study of our proposed RIFRE. We compare the RIFRE with RIFRE$_{\varnothing}$ that does not use the graph neural network (i.e. the number of layers is 0). The first sentence is a simple example, and both RIFRE and RIFRE$_{\varnothing}$ can extract it accurately. The second case is an SEO type. RIFRE$_{\varnothing}$ failed to extract the triples with *Japan* as the subject, and RIFRE can more easily deal with this situation after fusing the relational semantic information. The third case is an EPO type, *Sergey Brin* and *Google* contains a variety of semantic relations, and RIFRE can extract all the triples.

6. Conclusion and future work

This paper proposes a heterogeneous graph neural network for joint entity and relation extraction task and demonstrates the effectiveness of the proposed method on substantial experiments. We treat relation and word as nodes on the graph, and then aggregate information from different nodes through a message passing mechanism. Our method achieves the best results on NYT and WEBNLG datasets, and detailed experiments also show that our method can handle complex scenarios. Furthermore, our method is also significantly better than previous methods on the SemEval 2010 Task 8 dataset, indicating that our method is generalized. In the future, we will explore different graph network models to encode node representation better and generalize the representation iteration fusion strategy to more tasks.

CRediT authorship contribution statement

Kang Zhao: Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review & editing. **Hua Xu:** Supervision, Project administration, Funding acquisition. **Yue Cheng:** Writing - review & editing. **Xiaoteng Li:** Writing - review & editing. **Kai Gao:** Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This paper is funded by National Key R&D Program Projects of China (Grant No: 2018YFC1707605).

References

[1] G. Zhou, J. Su, J. Zhang, M. Zhang, Exploring various knowledge in relation extraction, in: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (Acl'05), 2005, pp. 427–434.
[2] Y.S. Chan, D. Roth, Exploiting syntactico-semantic structures for relation extraction, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, 2011, pp. 551–560.
[3] S. Zheng, F. Wang, H. Bao, Y. Hao, P. Zhou, B. Xu, Joint extraction of entities and relations based on a novel tagging scheme, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 1227–1236.
[4] X. Zeng, D. Zeng, S. He, K. Liu, J. Zhao, Extracting relational facts by an end-to-end neural model with copy mechanism, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, pp. 506–514.
[5] T.-J. Fu, P.-H. Li, W.-Y. Ma, GraphRel: Modeling text as relational graphs for joint entity and relation extraction, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 1409–1418.
[6] X. Zeng, S. He, D. Zeng, K. Liu, S. Liu, J. Zhao, 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, pp. 367–377.
[7] J. Liu, S. Chen, B. Wang, J. Zhang, N. Li, T. Xu, Attention as relation: Learning supervised multi-head self-attention for relation extraction, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, ijcai.org, 2020, pp. 3787–3793.
[8] Y. Yuan, X. Zhou, S. Pan, Q. Zhu, Z. Song, L. Guo, A relation-specific attention network for joint entity and relation extraction, in: C. Bessiere (Ed.), Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, ijcai.org, 2020, pp. 4054–4060.
[9] Z. Wei, J. Su, Y. Wang, Y. Tian, Y. Chang, A novel cascade binary tagging framework for relational triple extraction, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 1476–1488.
[10] M. Miwa, Y. Sasaki, Modeling joint entity and relation extraction with table representation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1858–1869.
[11] A. Katiyar, C. Cardie, Going out on a limb: Joint extraction of entity mentions and relations without dependency trees, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 917–928.
[12] Y. Hong, Y. Liu, S. Yang, K. Zhang, A. Wen, J. Hu, Improving graph convolutional networks based on relation-aware attention for end-to-end relation extraction, IEEE Access 8 (2020) 51315–51323.
[13] C. Bai, L. Pan, S. Luo, Z. Wu, Joint extraction of entities and relations by a novel end-to-end model with a double-pointer module, Neurocomputing 377 (2020) 325–333.
[14] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, 2017, arXiv preprint arXiv:1710.10903.
[15] V.G. Satorras, J.B. Estrach, Few-shot learning with graph neural networks, in: International Conference on Learning Representations, 2018.
[16] M. Tu, G. Wang, J. Huang, Y. Tang, X. He, B. Zhou, Multi-hop reading comprehension across multiple documents by reasoning over heterogeneous graphs, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 2704–2713.
[17] Y. Xie, H. Xu, J. Li, C. Yang, K. Gao, Heterogeneous graph neural networks for noisy few-shot relation classification, Knowl.-Based Syst. (2020) 105548.
[18] D. Wang, P. Liu, Y. Zheng, X. Qiu, X. Huang, Heterogeneous graph neural networks for extractive document summarization, 2020, arXiv preprint arXiv:2004.12393.
[19] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.
[20] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: Bridging the gap between human and machine translation, 2016, arXiv preprint arXiv:1609.08144.
[21] B. Yu, Z. Zhang, J. Su, Joint extraction of entities and relations based on a novel decomposition strategy, 2019, arXiv preprint arXiv:1909.04273.
[22] L. Wang, Z. Cao, G. De Melo, Z. Liu, Relation classification via multi-level attention cnns, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016, pp. 1298–1307.
[23] M.E. Peters, M. Neumann, R.L. Logan IV, R. Schwartz, V. Joshi, S. Singh, N.A. Smith, Knowledge enhanced contextual word representations, 2019, arXiv preprint arXiv:1909.04164.
[24] L.B. Soares, N. FitzGerald, J. Ling, T. Kwiatkowski, Matching the blanks: Distributional similarity for relation learning, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 2895–2905.
[25] S. Wu, Y. He, Enriching pre-trained language model with entity information for relation classification, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2361–2364.
[26] S. Riedel, L. Yao, A. McCallum, Modeling relations and their mentions without labeled text, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2010, pp. 148–163.
[27] C. Gardent, A. Shimorina, S. Narayan, L. Perez-Beltrachini, Creating training corpora for NLG micro-planners, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 179–188.
[28] I. Hendrickx, S.N. Kim, Z. Kozareva, P. Nakov, D.Ó. Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, S. Szpakowicz, SemEval-2010 Task 8: Multi-way classification of semantic relations between pairs of nominals, in: Proceedings of the 5th International Workshop on Semantic Evaluation, 2010, pp. 33–38.
[29] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al., Huggingface's transformers: State-of-the-art natural language processing, 2019, ArXiv, arXiv–1910.