Joint Entity and Relation Extraction With Set Prediction Networks

Dianbo Sui, Xiangrong Zeng, Yubo Chen, Kang Liu, and Jun Zhao

***Abstract*— Joint entity and relation extraction is an important task in natural language processing, which aims to extract all relational triples mentioned in a given sentence. In essence, the relational triples mentioned in a sentence are in the form of a set, which has no intrinsic order between elements and exhibits the permutation invariant feature. However, previous seq2seq-based models require sorting the set of relational triples into a sequence beforehand with some heuristic global rules, which destroys the natural set structure. In order to break this bottleneck, we treat joint entity and relation extraction as a direct set prediction problem, so that the extraction model is not burdened with predicting the order of multiple triples. To solve this set prediction problem, we propose networks featured by transformers with non-autoregressive parallel decoding. In contrast to autoregressive approaches that generate triples one by one in a specific order, the proposed networks are able to directly output the final set of relational triples in one shot. Furthermore, we also design a set-based loss that forces unique predictions through bipartite matching. Compared with cross-entropy loss that highly penalizes small shifts in triple order, the proposed bipartite matching loss is invariant to any permutation of predictions; thus, it can provide the proposed networks with a more accurate training signal by ignoring triple order and focusing on relation types and entities. Various experiments on two benchmark datasets demonstrate that our proposed model significantly outperforms the current state-of-the-art (SoTA) models. Training code and trained models are now publicly available at http://github.com/DianboWork/SPN4RE.**

***Index Terms*— Bipartite matching, joint entity and relation extraction, non-autoregressive decoder, set prediction.**

I. Introduction

A RELATIONAL triple consists of two entities connected by a semantic relation, which is in the form of ⟨*subject*, *relation*, *object*⟩. Extracting relational triples from unstructured raw texts is a crucial technology for automatic knowledge graph construction and, therefore, has gained tremendous interest in recent years.

There have been several studies addressing technical solutions for relational triple extraction. Early studies, such as [1] and [2], employ a pipeline manner to extract both entities and relations, where entities are recognized first, and then, the relation between the extracted entities is predicted. Such a pipeline ignores the relevance of entity identification and relation prediction [3] and tends to suffer from a severe error propagation problem.

In order to model the dependencies between entity identification and relation prediction explicitly and prevent error propagation in the pipeline manner, subsequent studies propose joint entity and relation extraction. These studies can be roughly categorized into three main paradigms. The first stream of work, such as [4], [5], and [6], treats the joint entity and relation extraction task as an end-to-end table filling problem. Although these methods use shared parameters to represent entities and relations in a single model, the entities and relations are extracted separately, and redundant information is produced [7]. The second stream of work, such as [7], [8], [9], and [10], transforms joint entity and relation extraction into sequence labeling. To do this, human experts need to design a complex tagging schema. The last stream of work, including [11], [12], [13], and [14], is driven by the sequence-to-sequence (seq2seq) model [15] to generate relational triples directly, which is a flexible framework to handle overlapping triples and does not require the substantial effort of human experts.

We follow the seq2seq-based models for joint entity and relation extraction. Despite the success of existing seq2seq-based extraction models, they are still limited by the autoregressive decoder and the cross-entropy loss. The reasons are as follows: the relational triples contained in a sentence are in the form of a set and have no intrinsic order in essence. Considering the example in Table I, predicting ⟨*Clinton*, *President-Country*, *the United States*⟩ first and then ⟨*Jiang Zemin*, *President-Country*, *China*⟩ has no difference from predicting ⟨*Jiang Zemin*, *President-Country*, *China*⟩ first and then ⟨*Clinton*, *President-Country*, *the United States*⟩. However, in order to adapt the autoregressive decoder, whose output is a sequence, the unordered target triples must be sorted in a certain order during the training phase. Meanwhile, cross entropy is a permutation-sensitive loss function, where a penalty is incurred for every triple that is predicted out

Manuscript received 2 October 2021; revised 3 August 2022 and 5 January 2023; accepted 25 March 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0106400, in part by the National Natural Science Foundation of China under Grant 61976211 and Grant 6217625, in part by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA27020000, in part by the Youth Innovation Promotion Association of Chinese Academy of Sciences, and in part by the Yunnan Provincial Major Science and Technology Special Plan Projects under Grant 202202AD080004. *(Corresponding authors: Kang Liu; Jun Zhao.)*

Dianbo Sui is with the Harbin Institute of Technology, Weihai 264200, China (e-mail: suidianbo@hit.edu.cn).

Xiangrong Zeng is with Kuaishou Technology, Beijing 100085, China (e-mail: xiangrongzeng@foxmail.com).

Yubo Chen, Kang Liu, and Jun Zhao are with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: yubo.chen@nlpr.ia.ac.cn; kliu@nlpr.ia.ac.cn; jzhao@nlpr.ia.ac.cn).

Digital Object Identifier 10.1109/TNNLS.2023.3264735

TABLE I
EXAMPLE OF JOINT ENTITY AND RELATION EXTRACTION

Input Sentence:
President Clinton welcomed President Jiang Zemin of China to visit the United States.
Output Relational Triples:
<*Clinton*, *President-Country*, *the United States*> <*Jiang Zemin*, *President-Country*, *China*>

of the position. Consequently, current seq2seq-based models not only need to learn how to generate triples, but also are required to consider the extraction order of multiple triples.

In this work, we formulate the task of joint entity and relation extraction as a set prediction problem, avoiding considering the order of multiple triples. In order to solve the set prediction problem, we propose an end-to-end network featured by transformers with non-autoregressive parallel decoding and bipartite matching training loss. In detail, there are three parts to the proposed set prediction networks (SPNs): a sentence encoder, a set generator, and a set-based loss function. First of all, we adopt the widely used BERT model [16] as the encoder to obtain the context-aware representation of a given sentence. Then, since an autoregressive decoder must generate items one by one in order, such a decoder is not suitable for generating unordered sets. In contrast, we leverage the transformer-based non-autoregressive decoder [17] as the set generator, which can predict all triples at once and avoid sorting triples. Finally, in order to assign a predicted triple to a unique ground-truth triple, we propose the bipartite matching loss function inspired by the assigning problem in operation research [18], [19], [20]. Compared with the cross-entropy loss that highly penalizes small shifts in triple order, the proposed loss function is invariant to any permutation of predictions; thus, it is suitable for evaluating the difference between the ground-truth set and the prediction set.

More specifically, our main contributions can be summarized as follows.

1) We formulate the joint entity and relation extraction task as a set prediction problem.
2) We combine non-autoregressive parallel decoding with the bipartite matching loss function to solve this problem.
3) Our proposed method yields state-of-the-art (SoTA) results on two benchmark datasets, and we perform various experiments to verify the effectiveness of the method.

This article is structured as follows. In Section II, we briefly review the background of this work. In the Section III, we explain the proposed set prediction networks (SPNs) in detail. The experimental evaluations and experimental results are described in Section IV. Besides, we discuss the scalability and limitations of the proposed SPNs in Section V. In the end, Section VI concludes this article.

II. RELATED WORK

Our work builds on prior studies in relation extraction, non-autoregressive decoder, and set prediction.

A. Relation Extraction

Relation extraction is a long-standing natural language process task of mining factual knowledge from free texts. When giving a sentence with annotated entities, this task degenerates into a simple task, namely, relation classification. Some studies leveraged convolutional neural networks [21], [22], recurrent neural networks [23], or graph-based methods [24] to solve the relation classification task. However, these methods must depend on the entities that are extracted in advance and could not truly extract relational triples.

When giving a sentence without any annotated entities, various methods, such as [1], [4], [6], [9], [25], [26], [27], [28], and [29], are proposed to extract entities and relations jointly. Existing studies on the joint entity and relation extraction task could be roughly divided into four main paradigms. First, pipeline-based methods, such as [1] and [2], first recognize entities [30] and then conduct relation classification [21]. Though widely used in practice, this pipeline architecture is inherently prone to error propagation between its components. Second, table filling-based methods, such as [4], [5], and [6], represent entities and relations with shared parameters. These methods extract entities and relations separately, resulting in redundant information during triple construction. Third, tagging-based methods, such as [7], [8], and [9], treat this task as a sequence labeling problem. This line of studies requires experts to design complex tagging schema. Besides, Zheng et al. [7] cannot handle overlap triple, and Wei et al. [9] suffer from cascading errors as the mistakes in subject detection cannot be corrected in later steps. Fourth, Seq2seq-based methods, such as [11], [12], [13], and [14], apply the seq2seq model to generate relational triples directly. Seq2seq-based methods can extract entities and relations jointly and do not suffer from cascading errors. But, seq2seq-based models require sorting the set of relational triples into a sequence beforehand with some heuristic global rules, which destroys the natural set structure of triples. To solve this bottleneck, we treat the joint entity and relation extraction as a set prediction problem and proposed networks featured by on-autoregressive parallel decoding and bipartite matching loss function.

B. Non-Autoregressive Decoder

Autoregressive decoders, such as [15], [31], [32], and [33], generate each token conditioned on the sequence of tokens previously generated, where the distribution over possible output is factored into a chain of conditional probabilities with a left-to-right causal structure

$$p(Y|X) = \prod_{i=1}^{T} p(y_i|y_{<i}, X) \tag{1}$$

where Y, X, and T denote the output sequence, input sequence, and the length of the output sequence, respectively. Compared with the autoregressive decoder, the non-autoregressive decoder [17] generates all the tokens of a target in parallel and can speed up inference. The non-autoregressive decoder assumes that each token is generated independently of each other conditioned on the input sentence.

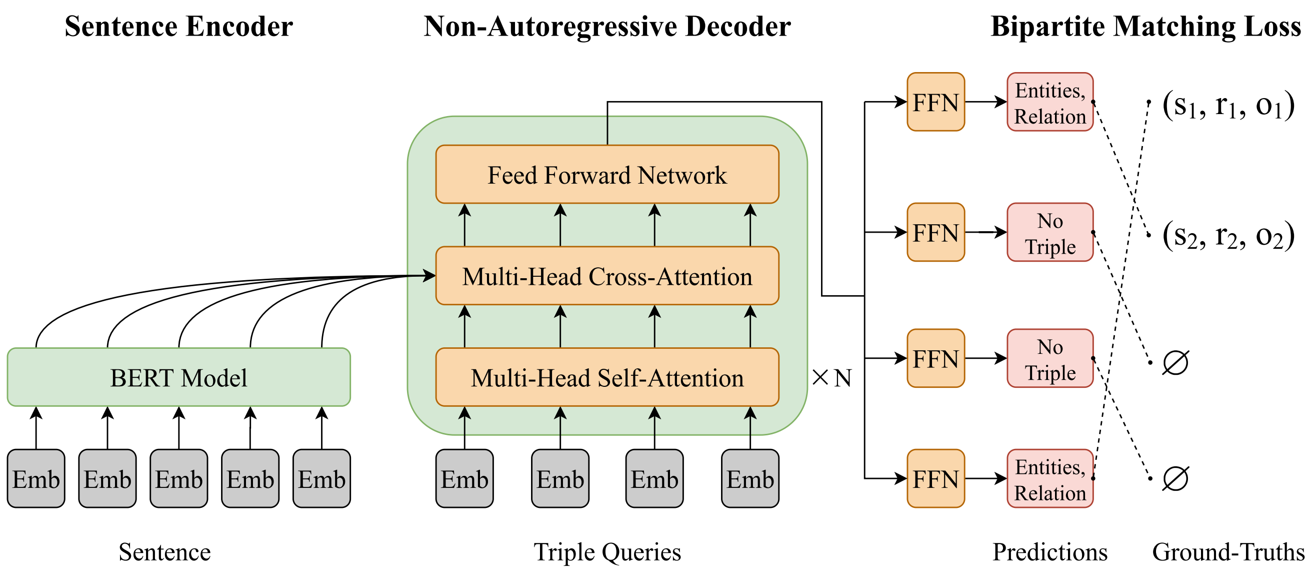


Fig. 1. Main architecture of the proposed SPNs. The SPNs are composed of a sentence encoder, a non-autoregressive decoder, and bipartite matching loss. The input sentences are first passed through the sentence encoder to obtain the context-aware representation. After that, the non-autoregressive decoder takes the context-aware representation and triple queries as the input and outputs the predicted triple set in parallel. In the training phase, bipartite matching uniquely assigns predictions with ground truths to provide accurate training signals.

The non-autoregressive decoder models the distribution over possible output as follows:

$$p(Y|X) = p_L(T|X)\prod_{i=1}^{T} p(y_i|X) \tag{2}$$

where the output sequence length T is modeled with a separate condition distribution $p_L(T|X)$.

Due to the autoregressive decoder featured by lower latency during inference, there is rich literature devoted to the topic, such as [34], [35], [36], [37], and [38]. Nowadays, non-autoregressive decoders have been widely explored in natural language and speech processing tasks, such as neural machine translation [34], [35] and automatic speech recognition [39], [40], [41]. To the best of our knowledge, this is the first work to apply non-autoregressive models to the field of information extraction. In this work, we resort to the non-autoregressive model to generate the set of triples in one shot.

C. Set Prediction

The problem with predicting sets is that the output order of the elements is arbitrary. To date, no canonical deep learning model has been designed specifically to directly predict sets. Densely connected networks are sufficient for constant-size set prediction but suffer from a high-computational cost. An alternative approach is to leverage autoregressive sequence models, such as recurrent neural networks [42], [43], [44], to generate all elements of the set one by one. In all cases, the loss function should be invariant by a permutation of the predictions, since there is no guarantee that the elements in the target set happen to be in the same order as they were generated. Assignment-based losses are a popular choice [45], [46], [47], [48], [49], which could guarantee that each target element has a unique match. Furthermore, a featurewise sort pooling (FSPool)-based method is developed in [50]. Inspired by studies on set prediction, we formulate the joint entity and relation extraction task as a set prediction problem, since the relational triples contained in a sentence are unordered. To the best of our knowledge, the proposed method in this article is the first work to apply a set prediction-based method to the joint entity and relation extraction task.

III. Methodology

The goal of the joint entity and relation extraction task is to identify all possible relational triples mentioned in a given sentence. Formally, given an raw sentence $X = x_1, x_2, \ldots, x_l$, the conditional probability of the target triple set $Y = \{(s_1, r_1, o_1), \ldots, (s_n, r_n, o_n)\}$ can be formulated as follows:

$$P(Y|X;\theta) = p_L(n|X)\prod_{i=1}^{n} p(Y_i|X, Y_{j\neq i};\theta) \tag{3}$$

where $p_L(n|X)$ models the size of the target triple set and $p(Y_i|X, Y_{j\neq i};\theta)$ means that a target triple Y_i is related not only to the given sentence X, but also to the other triples $Y_{j\neq i}$.

In this article, this conditional probability is parameterized using SPNs, which are shown in Fig. 1. Three key components of the proposed networks will be elaborated in the following. Concretely, we first present the sentence encoder, which represents each token in a given sentence based on its bidirectional context. Then, we introduce how to use the non-autoregressive decoder to generate a set of triples in a single pass. Finally, we describe a set-based loss, dubbed as bipartite matching loss, which forces a unique matching between outputs of the model and ground truths.

A. Sentence Encoder

The goal of this component is to get the context-aware representation of each token in an input sentence. Given the impressive performance of recent deep transformers [33] trained on variants of language modeling, we utilize the bidirectional encoder representations from transformers (BERT) [16] as the sentence encoder.

In the BERT encoder, the input sentence is segmented with tokens by the byte pair encoding [51] and then fed into the encoder. In each transformer block of the BERT model, there are two major components: a self-attention sublayer and a positionwise feedforward neural network. The self-attention sublayer is able to assist the encoder look at other words in the input sentence when it encodes a specific word. Specifically, given the output of the previous block (the $(t-1)$th block) in the BERT encoder $\mathbf{H}_\mathbf{e}^{\mathbf{t-1}}$, multihead attention, which is the core of the sublayer, treats $\mathbf{H}_\mathbf{e}^{\mathbf{t-1}} \in \mathbb{R}^{l\times d}$ as the query $\mathbf{Q}$, value $\mathbf{V}$, and key $\mathbf{K}$ and extracts information from

$$\begin{aligned}\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t} &= \text{MultiHead}\left(\mathbf{H}_\mathbf{e}^{\mathbf{t-1}}, \mathbf{H}_\mathbf{e}^{\mathbf{t-1}}, \mathbf{H}_\mathbf{e}^{\mathbf{t-1}}\right)\\ &= \text{Concat}(\text{head}_1, \ldots, \text{head}_A)\mathbf{W}^O \qquad (4)\\ \text{head}_i &= \text{Attention}\left(\mathbf{H}_\mathbf{e}^{\mathbf{t-1}}\mathbf{W}_i^Q, \mathbf{H}_\mathbf{e}^{\mathbf{t-1}}\mathbf{W}_i^K, \mathbf{H}_\mathbf{e}^{\mathbf{t-1}}\mathbf{W}_i^V\right) \qquad (5)\end{aligned}$$

where $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, $\mathbf{W}_i^V$, and $\mathbf{W}^O$ are trainable parameters. The particular attention mechanism used in BERT is called "scaled dot-product attention," which is defined as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \qquad (6)$$

where d_k is the dimension of the key column vector in $\mathbf{K}$. For building a deeper model, a residual connection [52] is employed, followed by a layer normalization [53] module:

$$\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t} = \text{LayerNorm}\left(\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t} + \mathbf{H}_\mathbf{e}^{\mathbf{t-1}}\right). \qquad (7)$$

On top of the self-attention sublayer, there is a fully connected feedforward network in each of the transformer blocks. The fully connected feedforward network contains two linear mapping with a ReLU activation [54] in between and is applied to each position separately and identically

$$\text{FFN}(\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t}) = \text{ReLU}\left(\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t}\mathbf{W}^1 + \mathbf{b}^1\right)\mathbf{W}^2 + \mathbf{b}^2 \qquad (8)$$

where $\mathbf{W}^1$, $\mathbf{W}^2$, $\mathbf{b}^1$, and $\mathbf{b}^2$ are trainable parameters. Another layer normalization and residual connection are also used, which can be written as follows:

$$\mathbf{H}_\mathbf{e}^\mathbf{t} = \text{LayerNorm}\left(\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t} + \text{FFN}\left(\hat{\mathbf{H}}_\mathbf{e}^\mathbf{t}\right)\right). \qquad (9)$$

The output of the BERT model is the context-aware embedding of tokens and is denoted as $\mathbf{H}_e \in \mathbb{R}^{l\times d}$, where l is the sentence length (including [CLS] and [SEP], two special start and end markers) and d is the number of hidden units in the BERT model.

B. Non-Autoregressive Decoder for Triple Set Generation

We regard joint entity and relation extraction as a set prediction problem and use the transformer-based non-autoregressive decoder [17] to directly generate the triple set. Previous studies, such as [11], [12], [13], and [14], transform the triple set to a triple sequence and then leverage the autoregressive decoder to generate triples one by one. In such a way, the conditional probability of the target triple set in (3) is modified into

$$P(Y|X;\theta) = \prod_{i=1}^{n} p(Y_i|X, Y_{j<i};\theta). \qquad (10)$$

In contrast, we use the non-autoregressive decoder to direct model (3). Compared with previous seq2seq-based methods, the non-autoregressive decoder can not only avoid learning the extraction order of multiple triples, but also generate triples based on bidirectional information, not just left-to-right information.

1) Input: Before decoding starts, the decoder needs to know the size of the target set; in other words, $p_L(n|X)$ in (3) is required to be modeled at first. In this work, we simplify $p_L(n|X)$ into a constant by requiring the non-autoregressive decoder to generate a fixed-size set of m predictions for each sentence, where m is set to be significantly larger than the typical number of triples in a sentence. Instead of copying tokens from the encoder side [17], the input of the decoder is initialized by m learnable embeddings that we refer to as triple queries. Note that all sentences share the same triple queries.

2) Decoder Architecture: The non-autoregressive decoder consists of a stack of N identical transformer blocks. In each transformer block, there are a multihead self-attention sublayer to model the relationship between triples and a multihead cross-attention sublayer to fuse the information of the given sentence. Different from the multihead self-attention sublayer, the queries used in the multihead cross-attention sublayer are projected from the outputs of the previous multihead cross-attention sublayer, whereas the keys and values are projected using the outputs of the BERT encoder

$$\begin{aligned}\hat{\mathbf{H}}_\mathbf{d}^\mathbf{t} &= \text{MultiHead}\left(\mathbf{H}_\mathbf{d}^\mathbf{t}, \mathbf{H}_\mathbf{e}, \mathbf{H}_\mathbf{e}\right)\\ &= \text{Concat}(\text{head}_1, \ldots, \text{head}_A)\mathbf{W}^O \qquad (11)\\ \text{head}_i &= \text{Attention}\left(\mathbf{H}_\mathbf{d}^\mathbf{t}\mathbf{W}_i^Q, \mathbf{H}_\mathbf{e}\mathbf{W}_i^K, \mathbf{H}_\mathbf{e}\mathbf{W}_i^V\right) \qquad (12)\end{aligned}$$

where $\mathbf{H}_\mathbf{d}^\mathbf{t}$ is the output of the previous multihead self-attention sublayer, $\mathbf{H}_\mathbf{e}$ is the output the BERT encoder, and $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, $\mathbf{W}_i^V$, and $\mathbf{W}^O$ are trainable parameters. Notably, compared with the autoregressive decoder, the non-autoregressive decoder does not have the constraint of an autoregressive factorization of the output, so there is no need to prevent earlier decoding steps from accessing information from later steps. Thus, there is no casual mask used in the multihead self-attention mechanism. Instead, we use the unmasked self-attention.

Through the non-autoregressive decoder, the m triple queries are transformed into m output embeddings, which are denoted as $\mathbf{H}_d \in \mathbb{R}^{m\times d}$. The output embeddings $\mathbf{H}_d$ are then independently decoded into relation types and entities by feedforward networks (FFNs), resulting m final predicted triples. Concretely, given an output embedding $\mathbf{h}_d \in \mathbb{R}^d$ in $\mathbf{H}_d$, the predicted relation type is obtained by

$$\mathbf{p}^r = \text{softmax}(\mathbf{W_r}\mathbf{h}_\text{d}) \in \mathbb{R}^\text{t} \qquad (13)$$

and the predicted entities (subject and object) are decoded by separately predicting the starting and ending indices with four l-class classifiers

$$\mathbf{p}^{s-\text{start}} = \text{softmax}\left(\mathbf{v}_1^{\mathrm{T}}\tanh(\mathbf{W}_1\mathbf{h}_\mathrm{d} + \mathbf{W}_2\mathbf{H}_\mathrm{e})\right) \in \mathbb{R}^l \quad (14)$$
$$\mathbf{p}^{s-\text{end}} = \text{softmax}\left(\mathbf{v}_2^{\mathrm{T}}\tanh(\mathbf{W}_3\mathbf{h}_\mathrm{d} + \mathbf{W}_4\mathbf{H}_\mathrm{e})\right) \in \mathbb{R}^l \quad (15)$$
$$\mathbf{p}^{o-\text{start}} = \text{softmax}\left(\mathbf{v}_3^{\mathrm{T}}\tanh(\mathbf{W}_5\mathbf{h}_\mathrm{d} + \mathbf{W}_6\mathbf{H}_\mathrm{e})\right) \in \mathbb{R}^l \quad (16)$$
$$\mathbf{p}^{o-\text{end}} = \text{softmax}\left(\mathbf{v}_4^{\mathrm{T}}\tanh(\mathbf{W}_7\mathbf{h}_\mathrm{d} + \mathbf{W}_8\mathbf{H}_\mathrm{e})\right) \in \mathbb{R}^l \quad (17)$$

where $\mathbf{W}_r \in \mathbb{R}^{t\times d}$, $\{\mathbf{W}_i \in \mathbb{R}^{d\times d}\}_{i=1}^{8}$, and $\{\mathbf{v}_i \in \mathbb{R}^{d}\}_{i=1}^{4}$ are learnable parameters, t is the total number of relation types (including a special relation type $\varnothing$ to indicate no triple), l is the sentence length, and $\mathbf{H}_e$ is the output of the BERT model.

C. Bipartite Matching Loss

The main difficulty of training is to score the predicted triples with respect to the ground truths. It is not proper to apply the cross-entropy loss function to measure the difference between two sets, since cross-entropy loss is sensitive to the permutation of the predictions. Inspired by the assigning problem in operation research [18], we propose a set prediction loss that can produce an optimal bipartite matching between predicted and ground-truth triples.

1) Notations: Let us denote by $\mathbf{Y} = \{\mathbf{Y}_i\}_{i=1}^{n}$ the set of ground-truth triples and $\hat{\mathbf{Y}} = \{\hat{\mathbf{Y}}_i\}_{i=1}^{m}$ the set of m predicted triples, where m is larger than n. We consider $\mathbf{Y}$ also as a set of size m padded with $\varnothing$ (no triple). Each element i of the ground-truth set can be seen as a $\mathbf{Y}_i = (r_i, s_i^{\text{start}}, s_i^{\text{end}}, o_i^{\text{start}}, o_i^{\text{end}})$, where r_i is the target relation type (which may be $\varnothing$), and $s_i^{\text{start}}, s_i^{\text{end}}, o_i^{\text{start}}$, and o_i^{end} are the starting or ending indices of subject s or object o. Each element i of the set of predicted triples is denoted as $\hat{\mathbf{Y}}_i = (\mathbf{p}_i^r, \mathbf{p}_i^{s-\text{start}}, \mathbf{p}_i^{s-\text{end}}, \mathbf{p}_i^{o-\text{start}}, \mathbf{p}_i^{o-\text{end}})$, which is calculated based on (13)–(17). The notations can also be found in the Nomenclature.

2) Loss: The process of computing bipartite matching loss is divided into two steps: finding an optimal matching and computing the loss function.

To find an optimal matching between the set of ground-truth triples $\mathbf{Y}$ and the set of predicted triples $\hat{\mathbf{Y}}$, we search for a permutation of elements $\pi^\star$ with the lowest cost

$$\pi^\star = \mathop{\arg\min}_{\pi\in\Pi(m)} \sum_{i=1}^{m} \mathcal{C}_{\text{match}}(\mathbf{Y}_i, \hat{\mathbf{Y}}_{\pi(i)}) \quad (18)$$

where $\Pi(m)$ is the space of all m-length permutations. $\mathcal{C}_{\text{match}}(\mathbf{Y}_i, \hat{\mathbf{Y}}_{\pi(i)})$ is a pairwise matching cost between the ground truth $\mathbf{Y}_i$ and the predicted triple with index $\pi(i)$. By taking into account both the prediction of relation type and the predictions of entity spans, we define $\mathcal{C}_{\text{match}}(\mathbf{Y}_i, \hat{\mathbf{Y}}_{\pi(i)})$ as follows:

$$\begin{aligned}\mathcal{C}_{\text{match}}(\mathbf{Y}_i, \hat{\mathbf{Y}}_{\pi(i)}) = -\mathbb{1}_{\{r_i\neq\varnothing\}}\Big[&\mathbf{p}_{\pi(i)}^{r}(r_i) + \mathbf{p}_{\pi(i)}^{s-\text{start}}\left(s_i^{\text{start}}\right)\\ &+ \mathbf{p}_{\pi(i)}^{s-\text{end}}\left(s_i^{\text{end}}\right) + \mathbf{p}_{\pi(i)}^{o-\text{start}}\left(o_i^{\text{start}}\right)\\ &+ \mathbf{p}_{\pi(i)}^{o-\text{end}}\left(o_i^{\text{end}}\right)\Big].\end{aligned} \quad (19)$$

Finding this optimal assignment $\pi^\star$ can be modeled as the assignment problem[1] in the operation research. In detail, we can view the set of ground truth $\mathbf{Y}$ as a set of people, the set of predicted triples $\hat{\mathbf{Y}}$ as a set of jobs. The cost of assigning $\mathbf{Y}_i$ (the people i) with $\hat{\mathbf{Y}}_\mathbf{j}$ (the job j) is defined as $\mathcal{C}_{\text{match}}(\mathbf{Y}_i, \hat{\mathbf{Y}}_j)$. The optimal assignment with the minimum total cost is easy to be computed via the classical Hungarian algorithm (shown in Algorithm 1) with $O(m^3)$ time complexity.

[1]https://en.wikipedia.org/wiki/Assignment_problem

Algorithm 1 Hungarian Algorithm

```
Input: A cost matrix $\mathcal{C} \in \mathbb{R}^{m\times m}$.
Step 1: Subtract row minima.
        Find the smallest entry in each row of the given cost matrix C, and then subtract the row minima from each entry in that row.
Step 2: Subtract column minima.
        Similarly, find the smallest entry in each column of the given cost matrix $\mathcal{C}$, and then subtract the column minima from each entry in that column.
Step 3: Cover all zeros with a minimum number of lines.
        Cover all zeros in the resulting matrix using a minimum number of lines through the row and columns. If $m$ lines are required, an optimal assignment exists among the zeros. The algorithm is finished. If the number of lines is less than $m$, go to Step 4.
Step 4: Create additional zeros.
        Find the smallest entry that is not covered by any line in Step 3. Subtract this entry from all uncovered entry, and then add it to all entry that are covered twice. Next, go back to Step 3
```

The second step is to compute the loss function for all pairs matched in the previous step. We define the loss as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{Y}, \hat{\mathbf{Y}}) = \sum_{i=1}^{m}\Big\{ &-\log \mathbf{p}_{\pi^\star(i)}^{r}(r_i)\\ &+ \mathbb{1}_{\{r_i\neq\varnothing\}}\Big[-\log \mathbf{p}_{\pi^\star(i)}^{s-\text{start}}\left(s_i^{\text{start}}\right)\\ &- \log \mathbf{p}_{\pi^\star(i)}^{s-\text{end}}\left(s_i^{\text{end}}\right)\\ &- \log \mathbf{p}_{\pi^\star(i)}^{o-\text{start}}\left(o_i^{\text{start}}\right)\\ &- \log \mathbf{p}_{\pi^\star(i)}^{o-\text{end}}\left(o_i^{\text{end}}\right)\Big]\Big\}\end{aligned} \quad (20)$$

where $\pi^\star$ is the optimal assignment computed in the first step (18).

D. Inference

In the inference phase, we pass the given sentence through the proposed SPNs and can obtain the predicted set $\hat{\mathbf{Y}} = \{\hat{\mathbf{Y}}_i\}_{i=1}^{m}$. In the predicted set, we first filter out the predicted triples that meet the following equation:

$$\mathbf{p}_i^r(r_i = \varnothing) = \max\left(\mathbf{p}_i^r\right) \quad (21)$$

where $\mathbf{p}_i^r \in \mathbb{R}^t$, and i indicates the ith element in the predicted set. Then, in the filtered predicted set, we decode the corresponding subjects and objects. In detail, for the ith element in the filtered predicted set, we choose the subject span $x_j, \ldots, x_{j'}$, such that $j \le j' \le j + \max\_\text{len}$, and $\mathbf{p}_i^{s-\text{start}}(j) \times \mathbf{p}_i^{s-\text{end}}(j')$ is maximized, and we choose the object span $x_k, \ldots, x_{k'}$, such that $k \le k' \le k + \max\_\text{len}$, and $\mathbf{p}_i^{o-\text{start}}(k) \times \mathbf{p}_i^{o-\text{end}}(k')$ is maximized. max _len is a predefined constant (e.g., 10), which controls the maximum length of subject spans and object spans.

IV. Experiments

In this section, we carry out an extensive set of experiments with the aim of answering the following research questions.

1) *RQ1:* What is the overall performance of our proposed model in joint entity and relation extraction?
2) *RQ2:* How does each design of our proposed model matter?
3) *RQ3:* How does our proposed model adapt to different overlapping patterns?
4) *RQ4:* What is the performance of our proposed model in sentences annotated with different numbers of triples?
5) *RQ5:* How do different numbers of triple queries affect our proposed model?

In the remainder of this section, we present the datasets, evaluation metrics, experimental setting, and all baselines.

A. Datasets

We evaluate the proposed SPNs on two widely used joint entity and relation extraction datasets: New York Times (NYT) [55] and WebNLG [56].[2]

1) NYT: This dataset is produced by the distantly supervised method, which automatically aligns Freebase with New York Times news articles from 1987 to 2007. There are 24 predefined relation types in total. Following previous studies [7], [11], we ignore the noise in this dataset and treat it as a supervised dataset. Since there are many versions of the NYT dataset, we adopted the preprocessed dataset used in [11], which is publicly available.

2) WebNLG: This dataset is originally designed for natural language generation (NLG) task. In this dataset, an instance includes several standard sentences written by humans and a set of triples. Every standard sentence mentions all triples of this instance. There are 246 predefined relation types in this dataset.

3) Overlap Between Triples: According to the previous work [11] on joint entity and relation extraction, sentences in the NYT dataset and the WebNLG dataset are split into three overlapping patterns: entity pair overlap (EPO), single entity overlap (SEO), and normal. Examples of the overlap pattern between triples are shown in Table II.

Formally, the EPO pattern is defined as a sentence containing some triples with overlapped entity pairs. Also, a sentence is classified to the SEO class if some mentioned triples share an overlapped entity, but these triples do not share overlapped entity pair. The rest of the sentences belong to Normal class, where none of the triples mentioned in these sentences involve overlapped entities. The statistics of overlap patterns between triples in the NYT and the WebNLG are shown in Table III. Note that a sentence can mention triples with the SEO pattern and the EPO pattern at the same time.

B. Evaluation Metrics

Standard micro-average precision, micro-average recall, and micro-average $F1$ scores are adopted to evaluate the performance, which are shown in the following equations:

$$\text{Precision} = \frac{\sum_c \text{TP}_c}{\sum_c \text{TP}_c + \sum_c \text{FP}_c} \tag{22}$$

$$\text{Recall} = \frac{\sum_c \text{TP}_c}{\sum_c \text{TP}_c + \sum_c \text{FN}_c} \tag{23}$$

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{24}$$

where c is the relation label, and TP, FP, and FN are the numbers of true positives, false positives, and false negatives, respectively.

A triple is regarded as a true positive instance if the relation type and the two corresponding entities are all correct. Notably, there are two ways to judge whether the extracted entities are correct. One is partial matching, where the extracted entities are regarded as correct if the predictions of subject and object are the same as the head words of the ground truth. Since the head word of entity is not annotated in most situations, the last word of entity is treated as the head word. The other is exact matching. In this way, only if the predictions of the subject and the object are identical to the ground truth, the extracted entities are treated as correct. Note that the training data under partial matching are different from the one under exact matching. Under partial matching, only head words of entities are annotated, while the whole entities are annotated under exact matching.

[2]These datasets can be found at https://github.com/xiangrongzeng/copy_re.

TABLE II
Examples With Normal Pattern, EPO Pattern, and SEO Pattern. S1 Belongs to Normal Class, Since the Mentioned Triples Do Not Have Overlapped Entities; S2 Belongs to EPO Class, Because the Entity Pair "⟨Beijing, China⟩" Is Overlapped in the Two Triples, and S3 Belongs to SEO Class, Because the Entity "Aarhus" in the Two Triples Is Overlapped, and These Two Triples Do Not Have Overlapped Entity Pair

Normal Pattern	S1: Rotterdam is located in the Netherlands and Frankfurt is located in the Germany.
	{< Rotterdam, *country*, Netherlands >; < Frankfurt, *country*, Germany>}
EPO Pattern	S2: Beijing is the capital of China.
	{<Beijing, *country*, China>; <Beijing, *capital*, China>}
SEO Pattern	S3: Heathrow airport serves London, which is led by Sadiq Khan.
	{<London, *leaderName*, Sadiq Khan>; <Heathrow Airport, *cityServed*, London>}

TABLE III
Statistics of Overlap Patterns Between Triples in the NYT Dataset and the WebNLG Dataset

Dataset		Normal	EPO	SEO	ALL
NYT	Training Set	37013	9782	14735	56195
	Validation Set	3306	849	1350	5000
	Test Set	3266	978	1297	5000
WebNLG	Training Set	1596	227	3406	5019
	Validation Set	182	16	318	500
	Test Set	246	26	457	703

TABLE IV

PRECISION (%), RECALL (%), AND $F1$ SCORE (%) OF OUR PROPOSED SPN AND SoTA METHODS ON THE NYT TEST SET. "†" INDICATES THE RESULT IS REPRODUCED BY US. "-" DENOTES THE METRIC IS NOT REPORTED OR CANNOT BE REPORTED IN THE ORIGINAL PAPER. THERE ARE SOME RESULTS MISSING DUE TO THAT THIS ARTICLE IS THE FIRST ATTEMPT TO PRESENT RESULTS IN BOTH PARTIAL MATCHING AND EXACT MATCHING

Models	**Partial Matching**			**Exact Matching**		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
NovelTagging [7]	62.4	31.7	42	-	-	-
CopyRE-One [11]	59.4	53.1	56.0	-	-	-
CopyRE-Mul[11]	61.0	56.6	58.7	-	-	-
GraphRel-1p [57]	62.9	57.3	60.0	-	-	-
GraphRel-2p [57]	63.9	60.0	61.9	-	-	-
CopyRRL [12]	77.9	67.2	72.1	-	-	-
WDec[13]	-	-	-	88.1	76.1	81.7
PNDec[13]	-	-	-	80.6	77.3	78.9
CopyMTL-One [14]	-	-	-	72.7	69.2	70.9
CopyMTL-Mul [14]	-	-	-	75.7	68.7	72.0
Attention as Relation [58]	-	-	-	88.1	78.5	83.0
CasRel [9]	89.7	89.5	89.6	90.1†	88.5†	89.3 †
SPN (Ours)	**93.5**	**91.8**	**92.7**	**92.5**	**92.2**	**92.3**

C. Implementation Details

To conduct a fair comparison, we use the cased base version of BERT as the sentence encoder in our experiments, which contains 110M parameters. The initial learning rate of BERT is set to 0.00001, and the initial learning of the non-autoregressive decoder is set to 0.00002. The number of stacked bidirectional transformer blocks in the non-autoregressive decoder is set to 3. We use the dropout strategy to mitigate overfitting, where the dropout rate is set to 0.1. Meanwhile, we apply gradient clipping to prevent exploding gradients. The SPNs are trained by minimizing the loss function defined in (20) through stochastic gradient descent over shuffled mini-batch with the AdamW update rule [59]. All experiments are conducted with an NVIDIA GeForce RTX 2080 Ti.

D. Baselines

In the experiments, the following SoTA models are used to compare with our proposed method.

1) NovelTagging [7] involves a novel tagging scheme. By using such a scheme, the joint entity and relation extraction task is transformed into a sequence labeling problem.
2) CopyRE [11] is a seq2seq-based model. Armed with a seq2seq architecture and the copy mechanism, CopyRE can effectively extract overlapping triples.
3) GraphRel [57] is a two-phase model, where the interaction between relations and entities is modeled by a relation-weighted graph convolutional network.
4) CopyRRL [12] is built on CopyRE, but combined the reinforcement learning to automatically learn the extraction order of triples. By using such a way, the interactions among triples can be considered.
5) CopyMTL [14] is based on a multitask learning framework. In this framework, conditional random fields are utilized to discover complete entities, and a seq2seq model is used to extract relational triples.
6) WDec [13] is a seq2seq model fused with a new representation scheme. By using such a scheme, WDec can extract exactly entities with their full names and is able to extract multiple relational triples with overlapping entities.
7) PNDec [13] is a modification of the seq2seq model. In PNDec, pointer networks are adopted. Armed with pointer networks, entities in the given sentence are recognized by detecting their start and end locations.
8) Attention as relation [58] adopts conditional random fields to discovery entities, Beyond the CRF, a multi-head self-attention-based network is leveraged to detect relations.
9) CasRel [9] is a tagging-based method. In CasRel, all possible subjects are first discovered, and then, for each identified subject, all possible relations and the corresponding objects are simultaneously identified by a relation specific tagger.

E. Main Results

To start, we address the research question RQ1. Tables IV and VI present the results of our model against baselines on the NYT dataset and WebNLG dataset. Overall, our proposed model outperforms baselines on these two benchmark datasets.

In the NYT dataset, the proposed SPNs outperform all the baselines in both partial matching and exact matching and achieve 3.1% and 3.0% improvements in $F1$ score, respectively, over the current SoTA method [9]. Combined with Table V, we observe that the proposed SPNs outperform the SoTA model [9] in both entity pair extraction and relation-type extraction, demonstrating the effectiveness of the proposed SPNs. We also observe that there is an obvious gap between the $F1$ score on relation-type extraction and entity pair extraction, but a trivial gap between entity pair extraction and overall extraction. It reveals that entity pair extraction is the main bottleneck of joint entity and relation extraction in the NYT dataset.

In the WebNLG dataset, since there is no unified version of data under exact matching, we only compare the proposed SPNs with baselines under partial matching metric. In general,

TABLE V
RESULTS ON EXTRACTING ELEMENTS OF RELATIONAL TRIPLES

Models	Element	NYT			WebNLG		
		Precision	Recall	F1	Precision	Recall	F1
CasRel [9]	(s,o)	89.2	90.1	89.7	**95.3**	91.7	93.5
	r	96.0	93.8	94.9	**96.6**	91.5	94.0
	Overall	89.7	89.5	89.6	**93.4**	90.1	91.8
SPN (Ours)	(s,o)	**93.2**	**92.7**	**92.9**	95.0	**95.4**	**95.2**
	r	**96.3**	**95.7**	**96.0**	95.2	**95.7**	**95.4**
	Overall	**93.3**	**91.7**	**92.5**	93.1	**93.6**	**93.4**

TABLE VI
PRECISION (%), RECALL (%), AND $F1$ SCORE (%) OF OUR PROPOSED SPN AND SOTA METHODS ON THE WEBNLG TEST SET

Models	Precision	Recall	F1
NovelTagging	52.5	19.3	28.3
CopyRE-One	32.2	28.9	30.5
CopyRE-Mul	37.7	36.4	37.1
GraphRel-1p	42.3	39.2	40.7
GraphRel-2p	44.7	41.1	42.9
CopyRRL	63.3	59.9	61.6
CasRel	**93.4**	90.1	91.8
SPN (Ours)	93.1	**93.6**	**93.4**

TABLE VII
EFFECT OF NON-AUTOREGRESSIVE DECODER ON PRECISION (%), RECALL (%), AND $F1$ SCORE (%). WE TRAIN THESE MODELS WITH BIPARTITE MATCHING LOSS AND KEEP THE OTHER SETTINGS THE SAME

Settings	Precision	Recall	F1
3-layer non-autoregressive decoder	**93.5**	**91.8**	**92.7**
2-layer non-autoregressive decoder	92.7	91.3	92.0
1-layer non-autoregressive decoder	91.9	90.9	91.4
3-layer autoregressive decoder	90.7	90.0	90.3
2-layer autoregressive decoder	90.5	88.8	89.4
1-layer autoregressive decoder	89.9	88.1	89.0

TABLE VIII
EFFECT OF BIPARTITE MATCHING LOSS ON PRECISION (%), RECALL (%), AND $F1$ SCORE (%). IN THESE MODELS, WE ADOPT A THREE-LAYER NON-AUTOREGRESSIVE DECODER AND KEEP THE OTHER SETTINGS THE SAME

Settings	Precision	Recall	F1
Bipartite matching loss	**93.5**	**91.8**	**92.7**
Cross entropy loss	87.2	80.9	84.0

the proposed SPNs can achieve the best $F1$ score under partial matching, which is 93.4%. Compared with the result of the SoTA model (CasRel) [9], there is 1.6% improvement, which is 91.8%. We also find that the proposed SPNs are more balanced in terms of precision and recall compared with the SoTA model. A further analysis combined with Table V shows that the proposed SPNs exhibit balanced results of precision and recall in both entity pair extraction and relation-type extraction, while there is an obvious gap between precision and recall in the results of CasRel [9]. We conjecture that this is largely due to that the number of triple queries is set to be significantly larger than the typical number of triples in a sentence, which ensures that enough triples can be recalled. In addition, we find that the results of entity pair extraction and relation-type extraction are much higher than that of joint entity and relation extraction, which means that the accurate combination of entity pair extraction and relation-type extraction is the key to improve the joint entity and relation extraction.

F. Ablation Studies

Next, we turn to the research question RQ2. To this end, we perform ablation studies to investigate the importance of the non-autoregressive decoder and the bipartite matching loss. To evaluate the importance of the non-autoregressive decoder, we change the number of decoder layers and replace the non-autoregressive decoder with the autoregressive decoder. To evaluate the importance of bipartite matching loss, we replace bipartite matching loss with cross-entropy loss.

Table VII presents the effect of non-autoregressive decoder. From this table, we find that increasing the number of layers of the non-autoregressive decoder can achieve better results. When setting the number of decoder layers to 1, 2, and 3, the best results are 91.4%, 92.0%, and 92.7%, respectively. We conjecture that this is largely due to that with the deepening of the non-autoregressive decoder layers, more multihead self-attention modules allow for better modeling of relationships between triple queries, and more multihead cross-attention modules allow for more complete integration of sentence information into triple queries. Besides, we also find that replacing the non-autoregressive decoder with the autoregressive decoder leads to a notable drop in performance. In detail, when replacing the three-layer non-autoregressive decoder with the three-layer autoregressive decoder, the model suffers a 2.4% absolute drop (from 92.7% to 90.3%). Based on the above findings, we arrive to the following conclusion: the non-autoregressive decoder is effective in the proposed SPNs.

Table VIII presents the effect of bipartite matching loss. Compared the proposed bipartite matching loss with the widely used cross-entropy loss, we find that there is an 8.9% improvement, which indicates bipartite matching loss is very suitable for joint entity and relation extraction. We hypothesize that in order to adapt to cross-entropy loss, the model is required to consider the generative order of triples, which places an unnecessary burden on the model.

G. Detailed Results on Different Overlapping Patterns

Then, to address the research question RQ3, we perform further experiments on the benchmark dataset, NYT and

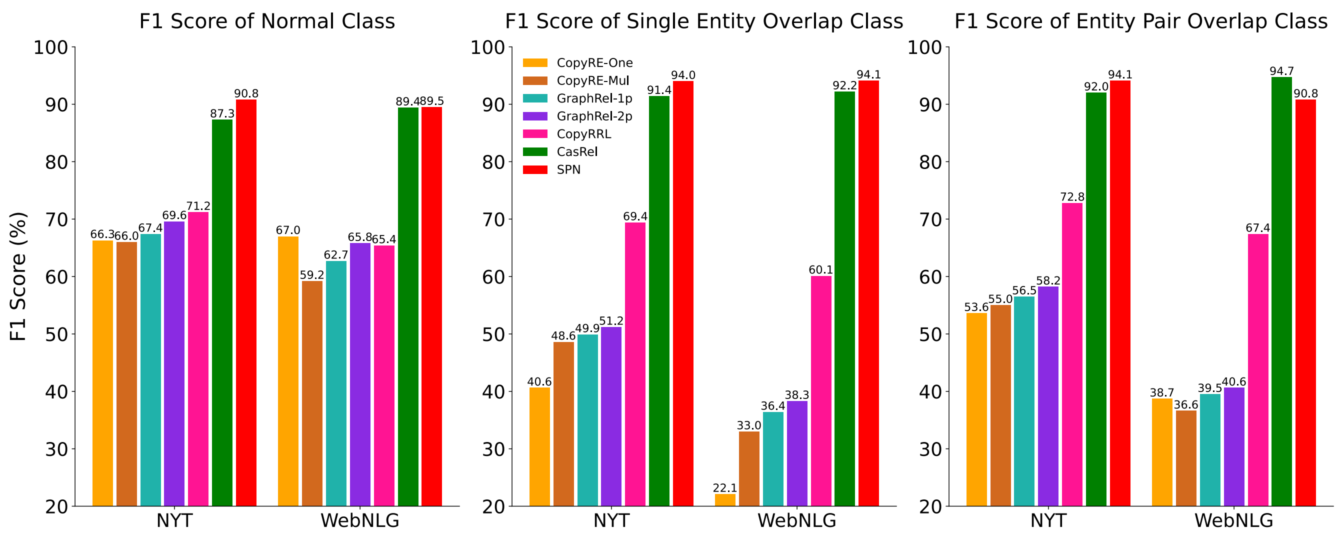


Fig. 2. $F1$ score of extracting relational triples from sentences with the different overlapping patterns.

TABLE IX

Partial Matching $F1$ Score of Conducting Extraction in Sentences That Contains Different Numbers of Triples. The Sentences of the Test Sets Are Divided Into Five Subclasses, Where Each Subclass Contains Sentences That Mention 1, 2, 3, 4, or ≥5 Triples, Respectively

Models	NYT					WebNLG				
	N=1	N=2	N=3	N=4	N≥5	N=1	N=2	N=3	N=4	N≥5
CopyRE-One [11]	66.6	52.6	49.7	48.7	20.3	65.2	33.0	22.2	14.2	13.2
CopyRE-Mul [11]	67.1	58.6	52.0	53.6	30.0	59.2	42.5	31.7	24.2	30.0
GraphRel-1p [57]	69.1	59.5	54.4	53.9	37.5	63.8	46.3	34.7	30.8	29.4
GraphRel-2p [57]	71.0	61.5	57.4	55.1	41.1	66.0	48.3	37.0	32.1	32.1
CopyRRL [12]	71.7	72.6	72.5	77.9	45.9	63.4	62.2	64.4	57.2	55.7
CasRel [9]	88.2	90.3	91.9	94.2	83.7	89.3	90.8	94.2	92.4	90.9
SPN (Ours)	**90.9**	**93.4**	**94.2**	**95.5**	**90.6**	**89.5**	**91.3**	**96.4**	**94.7**	**93.8**

WebNLG, to verify the capacity of the proposed SPNs in handling the overlapping problem. Fig. 2 shows the results of the proposed SPNs and some baseline models in normal pattern, single entity overlap (SEO) pattern, and entity pair overlap (EPO) pattern.

In the NYT dataset, the proposed SPNs outperform all baseline models on these three overlap patterns. In detail, there are 3.5%, 2.6%, and 2.1% improvements compared with the SoTA model [9] in Normal, SEO, and EPO classes, respectively. Such results show that our proposed model is very effective in handling the overlapping problem, which is widely existed in joint entity and relation extraction. In addition, we also observe that the performance of all seq2seq-based models, such as CopyRE [11] and CopyRRL [12], on Normal, EPO, and SEO presents a decreasing trend, which indicates that the Normal class presents a relatively easiest pattern, while EPO and SEO classes are the relatively harder ones for these seq2seq-based models to extract. In contrast, our proposed model attains consistently strong performance over all three overlapping patterns.

In the WebNLG dataset, the proposed SPNs can achieve good results on these three overlap patterns. Specifically, there are 0.1% and 1.9% improvements compared with the SoTA model [9] in Normal and SEO and classes, respectively. In the EPO class, although the proposed networks can substantially outperform most baselines, they underperform compared with the CasRel [9]. We conjecture that is due to underfitting. In detail, as shown in Table III, there are 227 examples with the EPO pattern in the training set, accounting for 4.52% of the total. During the training phase, the proposed networks may pay most attention to modeling the major pattern (the SEO pattern) due to sufficient samples and ignore the minor pattern (the EPO pattern), resulting in relatively poor performance on the EPO pattern.

H. Detailed Results on Sentences Annotated With Different Numbers of Triples

In this section, we answer the research question **RQ4**. To do this, we compare the models' ability on extracting relational triples from sentences annotated with different numbers of triples. The sentences in test sets are divided into five subclasses, and each subclass contains sentences that are annotated with 1, 2, 3, 4, or ≥5 triples. The detailed results on the benchmark datasets, NYT and WebNLG, are presented in Table IX. In general, the proposed SPNs achieve the best results in all subclasses. Besides, we find that compared with the SoTA model [9], the proposed SPNs gain considerable improvements in all five subclasses. From the detailed results, we also notice that the greatest improvement of $F1$ score on the two benchmark datasets both comes from the most difficult subclass ($N \geq 5$); in particular, there is a 6.9% improvement

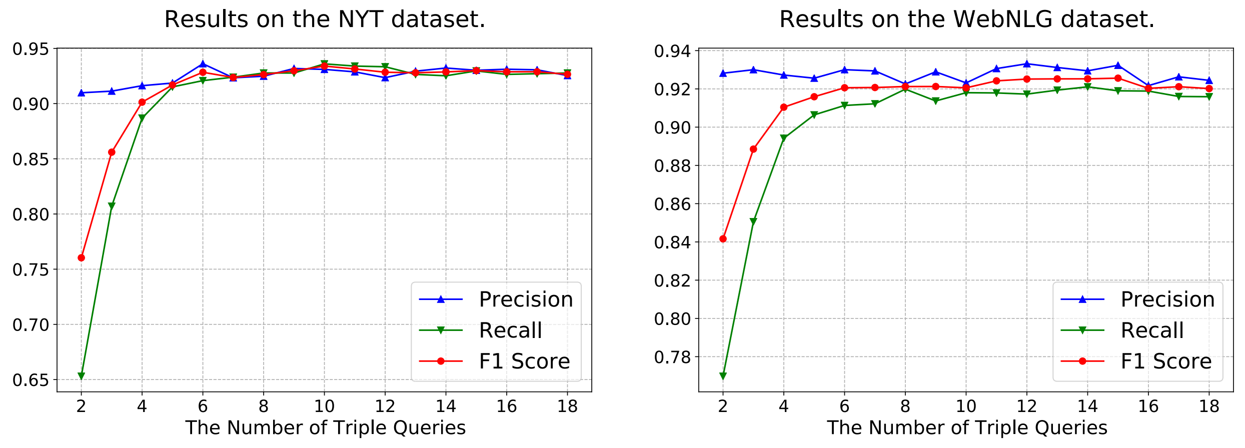


Fig. 3. Results of varying the number of triple queries on the NYT dataset and the WebNLG dataset.

in the NYT data. Such results indicate that the proposed SPNs are more suitable for intricate scenarios than the current SoTA model.

I. Varying the Number of Triple Queries

In this section, we answer the last research question RQ5. To this end, we vary the number of triple queries m from 2 to 18 in both the NYT dataset and the WebNLG dataset. The experimental results are shown in Fig. 3. From Fig. 3, we can find that the following hold: 1) setting too few triples ($m \leq 6$) will have a negative impact, since many triples will not be recalled; 2) as the number of triple queries increases, the recall value is constantly improved, while the precision score is likely to decline; and 3) the optimal number of triple queries varies across datasets, which is a critical hyperparameter that must be adjusted constantly during experiments. In the NYT dataset, the optimal result is achieved with 15 triple queries, while the optimal number of triple queries in the WebNLG dataset is 10. We conjecture that the optimal number of triple queries is related to the average number of triples in the given dataset.

V. Discussion

In this section, we mainly discuss the scalability and limitations of the proposed SPNs.

A. Extraction Beyond Sentence Boundaries

So far, we have only discussed how the proposed SPNs tackle sentence-level joint entity and relation extraction. Therefore, we ask: Could the proposed networks be scaled to the document-level setting?

To answer this question, we conduct some experiments. In detail, first, to arm the networks with the ability of document-level modeling, we replace the sentence encoder with a document encoder. In the experiments, we adopt Longformer [60] (the base version) as the document encoder, which can support sequences of length up to 4096 rather than 512 in BERT. Then, we choose the DIWE dataset [61] as the test bed, since its training set, validation set, and test set are publicly available.[3] But, unlike the entity-centric test protocol in the DIWE, we modify it to the mention-centric protocol[4], since

[3]https://github.com/klimzaporojets/DWIE

[4]The NYT and WebNLG dataset both follow the mention-centric test protocol.

TABLE X
Detailed Results on Extracting Elements of Relational Triples in the DWIE Dataset

Models	Element	Precision	Recall	F1
CasRel [9]	(s,o)	71.7	33.3	45.5
	r	94.6	44.0	60.1
	Overall	61.9	27.9	38.7
SPN (Ours)	(s,o)	**74.4**	**36.6**	**49.1**
	r	**95.1**	**46.8**	**62.7**
	Overall	**65.3**	**32.1**	**43.1**

the proposed networks do not hold the coreference resolution function. In the mention-centric protocol, we assume that if an entity cluster (concept) $\mathcal{C}_i$ has a certain relationship r with another entity cluster $\mathcal{C}_k$, all entity mentions grouped in $\mathcal{C}_i$ have the relationship r with the entity mentions grouped in $\mathcal{C}_j$, and we require to extract all mentioned triple (the subject and the object are both entity mention rather than concept or entity cluster) in the document. Finally, we choose CasRel [9] as the baseline, which is also armed with a Longformer encoder to achieve document modeling.

The experimental results are shown in Table X. From the table, we find that the following hold: 1) our proposed SPNs can handle document-level joint entity and relation extraction; 2) our proposed SPNs outperform CasRel [9] and achieve 4.4% improvements in $F1$ score; and 3) all methods suffer from low recall. The reasons behind that are the following: in terms of SPNs, since there are thousands of triples mentioned in the document, we need to set the number of triple queries m in the Nomenclature to be large enough. However, setting a large m requires large GPU memory. Due to we conduct document-level experiments with an NVIDIA GeForce RTX 3090 (24G memory), m is set to 400, which is much lower than the typical number of triples mentioned in a document. We believe that supported by the GPU with larger memory, our proposed model can perform better. In terms of CasRel, the subject tagger needs to conduct binary classification on each token. However, subjects of triples are scattered and sparse in long documents. Due to the extremely imbalanced ratio between the negative and positive samples, the subject tagger fails to recognize subjects, which leads to a low recall rate.

B. Limitations and Future Work

This task of joint entity and relation extraction is far from being solved. We find that relation types exhibit an imbalanced or long-tailed distribution in the NYT dataset and the WebNLG dataset. However, our proposed method cannot handle such a long-tailed distribution properly. In our future work, we will concentrate on how to combine cost-sensitive learning with the proposed SPNs. Besides, the computational complexity of the attention layer (self-attention + cross attention) in the non-autoregressive decoder is $\mathcal{O}(m^2d + mld)$, where m denotes the number of triple queries, d is the hidden dimension of the decoder, and l represents the length the given sentence. When adopting the proposed networks to the document-level extraction rather than sentence-level extraction, it places higher demands on the memory of GPU. In our future work, we will improve our proposed method with a more lightweight decoder. Furthermore, in this article, we only conduct experiments on the English dataset. In our future work, we will adopt our proposed method in other languages and cross-lingual settings.

VI. Conclusion

In this article, we introduce SPNs for joint entity and relation extraction. Compared with previous seq2seq-based models, we formulate the joint entity and relation extraction task as a set prediction problem. In such a way, the extraction model will be relieved of predicting the extraction order of multiple triples. To solve the set prediction problem, we combine non-autoregressive parallel decoding with bipartite matching loss function. We conduct extensive experiments on two widely used datasets to validate the effectiveness of the proposed SPNs. Experimental results show that our proposed networks outperform SoTA baselines over different scenarios.

References

[1] D. Zelenko, C. Aone, and A. Richardella, "Kernel methods for relation extraction," *J. Mach. Learn. Res.*, vol. 3, pp. 1083–1106, Feb. 2003.

[2] Y. S. Chan and D. Roth, "Exploiting syntactico-semantic structures for relation extraction," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2011, pp. 551–560.

[3] Q. Li and H. Ji, "Incremental joint extraction of entity mentions and relations," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 1–11.

[4] M. Miwa and M. Bansal, "End-to-end relation extraction using LSTMs on sequences and tree structures," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1105–1116.

[5] P. Gupta, H. Schütze, and B. Andrassy, "Table filling multi-task recurrent neural network for joint entity and relation extraction," in *Proc. COLING*, 2016, pp. 2537–2547.

[6] M. Zhang, Y. Zhang, and G. Fu, "End-to-end neural relation extraction with global optimization," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 1730–1740.

[7] S. Zheng, F. Wang, H. Bao, Y. Hao, P. Zhou, and B. Xu, "Joint extraction of entities and relations based on a novel tagging scheme," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 1–10.

[8] D. Dai, X. Xiao, Y. Lyu, S. Dou, Q. She, and H. Wang, "Joint extraction of entities and overlapping relations using position-attentive sequence labeling," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 6300–6308.

[9] Z. Wei, J. Su, Y. Wang, Y. Tian, and Y. Chang, "A novel cascade binary tagging framework for relational triple extraction," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 1476–1488.

[10] Y. Wang, B. Yu, Y. Zhang, T. Liu, H. Zhu, and L. Sun, "TPLinker: Single-stage joint extraction of entities and relations through token pair linking," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 1–11.

[11] X. Zeng, D. Zeng, S. He, K. Liu, and J. Zhao, "Extracting relational facts by an end-to-end neural model with copy mechanism," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 506–514.

[12] X. Zeng, S. He, D. Zeng, K. Liu, S. Liu, and J. Zhao, "Learning the extraction order of multiple relational facts in a sentence with reinforcement learning," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 367–377.

[13] T. Nayak and H. T. Ng, "Effective modeling of encoder–decoder architecture for joint entity and relation extraction," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 8528–8535.

[14] D. Zeng, H. Zhang, and Q. Liu, "CopyMTL: Copy mechanism for joint extraction of entities and relations with multi-task learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 9507–9514.

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1–13.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[17] J. Gu, J. Bradbury, C. Xiong, V. O. K. Li, and R. Socher, "Non-autoregressive neural machine translation," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=B1l8BtlCb

[18] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics*, vol. 52, no. 1, pp. 7–21, Feb. 2005.

[19] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Soc. Ind. Appl. Math.*, vol. 5, no. 1, pp. 32–38, 1957.

[20] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *J. ACM*, vol. 19, no. 2, pp. 248–264, Apr. 1972.

[21] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation classification via convolutional deep neural network," in *Proc. 25th Int. Conf. Comput. Linguistics, Tech. Papers*, 2014, pp. 2335–2344.

[22] Y. Xiao, Y. Jin, and K. Hao, "Adaptive prototypical networks with label words and joint representation learning for few-shot relation classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 3, pp. 1406–1417, Mar. 2023.

[23] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin, "Classifying relations via long short term memory networks along shortest dependency paths," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1785–1794.

[24] W. Li, T. Qian, M. Zhong, and X. Chen, "Interactive lexical and semantic graphs for semisupervised relation extraction," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 10, 2022, doi: 10.1109/TNNLS.2021.3138956.

[25] S. Wang, Y. Zhang, W. Che, and T. Liu, "Joint extraction of entities and relations based on a novel graph scheme," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4461–4467.

[26] K. Dixit and Y. Al-Onaizan, "Span-level model for relation extraction," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 5308–5314.

[27] Y. Yuan, X. Zhou, S. Pan, Q. Zhu, Z. Song, and L. Guo, "A relation-specific attention network for joint entity and relation extraction," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 4054–4060.

[28] S. Zhao, M. Hu, Z. Cai, and F. Liu, "Dynamic modeling cross-modal interactions in two-phase prediction for entity-relation extraction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 3, pp. 1122–1131, Mar. 2023.

[29] J. Wang et al., "TGIN: Translation-based graph inference network for few-shot relational triplet extraction," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Nov. 14, 2022, doi: 10.1109/TNNLS.2022.3218981.

[30] Z. Yang, J. Ma, H. Chen, J. Zhang, and Y. Chang, "Context-aware attentive multilevel feature fusion for named entity recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 8, 2022, doi: 10.1109/TNNLS.2022.3178522.

[31] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. [Online]. Available: https://aclanthology.org/D14-1179, doi: 10.3115/v1/D14-1179.

[32] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.

[33] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[34] J. Lee, E. Mansimov, and K. Cho, "Deterministic non-autoregressive neural sequence modeling by iterative refinement," in *Proc. Conf. Empirical Methods Natural Lang. Process.* Brussels, Belgium: Association for Computational Linguistics, Oct./Nov. 2018, pp. 1173–1182. [Online]. Available: https://aclanthology.org/D18-1149, doi: 10.18653/v1/D18-1149.

[35] X. Ma, C. Zhou, X. Li, G. Neubig, and E. Hovy, "FlowSeq: Non-autoregressive conditional sequence generation with generative flow," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*. Hong Kong: Association for Computational Linguistics, Nov. 2019, pp. 4282–4292. [Online]. Available: https://aclanthology.org/D19-1437, doi: 10.18653/v1/D19-1437.

[36] Y. Ren, J. Liu, X. Tan, Z. Zhao, S. Zhao, and T.-Y. Liu, "A study of non-autoregressive model for sequence generation," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020.

[37] Q. Ran, Y. Lin, P. Li, and J. Zhou, "Learning to recover from multi-modality errors for non-autoregressive neural machine translation," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 3059–3069.

[38] X. Kong, Z. Zhang, and E. Hovy, "Incorporating a local translation mechanism into non-autoregressive translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2020, pp. 1067–1073.

[39] N. Chen, S. Watanabe, J. Villalba, and N. Dehak, "Listen and fill in the missing letters: Non-autoregressive transformer for speech recognition," 2019, *arXiv:1911.04908*.

[40] Z. Tian, J. Yi, J. Tao, Y. Bai, S. Zhang, and Z. Wen, "Spike-triggered non-autoregressive transformer for end-to-end speech recognition," 2020, *arXiv:2005.07903*.

[41] Y. Bai, J. Yi, J. Tao, Z. Tian, Z. Wen, and S. Zhang, "Listen attentively, and spell once: Whole sentence generation via a non-autoregressive architecture for low-latency speech recognition," 2020, *arXiv:2005.04862*.

[42] O. Vinyals, S. Bengio, and M. Kudlur, "Order matters: Sequence to sequence for sets," 2015, *arXiv:1511.06391*.

[43] R. Stewart, M. Andriluka, and A. Y. Ng, "End-to-end people detection in crowded scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2325–2333.

[44] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "GraphRNN: Generating realistic graphs with deep auto-regressive models," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5708–5717.

[45] H. Fan, H. Su, and L. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 605–613.

[46] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 206–215.

[47] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 213–229.

[48] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," 2020, *arXiv:2010.04159*.

[49] Z. Yao, J. Ai, B. Li, and C. Zhang, "Efficient DETR: Improving end-to-end object detector with dense prior," 2021, *arXiv:2104.01318*.

[50] Y. Zhang, J. Hare, and A. Prügel-Bennett, "Deep set prediction networks," 2019, *arXiv:1906.06565*.

[51] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 1715–1725.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[53] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[54] D. E. Rumelhart et al., "A general framework for parallel distributed processing," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1, nos. 45–76. Cambridge, MA, USA: MIT Press, 1986, p. 26.

[55] S. Riedel, L. Yao, and A. McCallum, "Modeling relations and their mentions without labeled text," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Cham, Switzerland: Springer, 2010, pp. 148–163.

[56] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, "Creating training corpora for NLG micro-planners," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 1–10.

[57] T.-J. Fu, P.-H. Li, and W.-Y. Ma, "GraphRel: Modeling text as relational graphs for joint entity and relation extraction," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1409–1418.

[58] J. Liu, S. Chen, B. Wang, J. Zhang, N. Li, and T. Xu, "Attention as relation: Learning supervised multi-head self-attention for relation extraction," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 3787–3793.

[59] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*.

[60] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," 2020, *arXiv:2004.05150*.

[61] K. Zaporojets, J. Deleu, C. Develder, and T. Demeester, "DWIE: An entity-centric dataset for multi-task document-level information extraction," *Inf. Process. Manage.*, vol. 58, no. 4, Jul. 2021, Art. no. 102563.

Dianbo Sui received the bachelor's degree from Nankai University, Tianjin, China, in 2016, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2022.

He is currently a Lecturer with the Harbin Institute of Technology, Weihai. He has published several top-level papers in international conferences, including ACL, EMNLP, NAACL, and COLING. His research interests include natural language processing and information extraction.

Xiangrong Zeng received the bachelor's degree from Central South University, Changsha, China, in 2014, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2019.

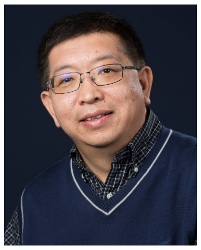
He is currently an Algorithm Engineer with Kuaishou Technology, Beijing. He has published several top-level papers in international conferences, including ACL, EMNLP, AAAI, and EACL. His research interests focus on information extraction.

Yubo Chen received the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2017.

He is currently an Associate Professor with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences. He has published more than 30 papers in international conferences, including ACL, EMNLP, IJCAI, and AAAI. His research interests focus on natural language processing, knowledge graph construction, and event extraction.

Kang Liu received the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2010.

He is currently a Professor with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences. He has published more than 60 papers in international journals and conferences, including TKDE, ACL, EMNLP, IJCAI, and AAAI. His research interests focus on natural language processing, relation extraction, event extraction, and dialog system.



Jun Zhao received the Ph.D. degree from Tsinghua University, Beijing, China, in 1998.

He is currently a Professor with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing. He has published more than 70 papers in international journals and conferences, including TKDE, ACL, EMNLP, IJCAI, AAAI, WWW, and CIKM. His research interests focus on natural language processing, relation extraction, event extraction, question answering, and dialog system.