SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

May 2024, Vol. 67, Iss. 5, 152105:1–152105:15
https://doi.org/10.1007/s11432-022-3608-y

Span-based joint entity and relation extraction augmented with sequence tagging mechanism

Bin JI, Shasha LI*, Hao XU*, Jie YU*, Jun MA, Huijun LIU* & Jing YANG

College of Computer, National University of Defense Technology, Changsha 410073*, China*

Received 21 February 2022/Revised 27 May 2022/Accepted 28 August 2022/Published online 3 April 2024

Abstract Span-based joint extraction simultaneously conducts named entity recognition (NER) and relation extraction (RE) in a text span form. However, since previous span-based models rely on span-level classifications, they cannot benefit from token-level label information, which has been proven advantageous for the task. In this paper, we propose a sequence tagging augmented span-based network (STSN), a span-based joint model that can make use of token-level label information. In STSN, we construct a core neural architecture by deep stacking multiple attention layers, each of which consists of three basic attention units. On the one hand, the core architecture enables our model to learn token-level label information via the sequence tagging mechanism and then uses the information in the span-based joint extraction; on the other hand, it establishes a bi-directional information interaction between NER and RE. Experimental results on three benchmark datasets show that STSN consistently outperforms the strongest baselines in terms of F1, creating new state-of-the-art results.

Keywords joint extraction, named entity recognition, relation extraction, span, sequence tagging mechanism

1 Introduction

The joint entity and relation extraction task extract both entities and semantic relations between entities from raw texts. It acts as a stepping stone for a variety of downstream natural language processing (NLP) tasks [1], such as question answering. According to the classification methods, we divide the existing models for the task into two categories: sequence tagging-based models [2–5] and span-based models [6–10]. The former is based on the sequence tagging mechanism and performs token-level classifications. The latter is based on the span-based paradigm and performs span-level classifications. Since the sequence tagging mechanism and the span-based paradigm are considered to be distinct methodologies, existing joint extraction models permit the use of just one of them. Specifically, the span-based paradigm consists of three typical steps: it first splits raw texts into text spans (a.k.a. candidate entities), such as the "Jack" and "Harvard University" in Figure 1; it then constructs ordered span pairs (a.k.a. candidate relation tuples), such as the ⟨"Jack", "Harvard University"⟩ and ⟨"Harvard University", "Jack"⟩; and finally, it jointly classifies spans and span pairs. For example, it classifies the "Jack" and "Harvard University" into `PER` and `ORG`, respectively. And it classifies the ⟨"Jack", "Harvard University"⟩ and ⟨"Harvard University", "Jack"⟩ into `WORK` and `NoneType`, respectively[1)].

The majority of span-based models [7,8,10] use pre-trained language models (PLMs) as their encoders directly, which relies on the encoding ability of PLMs heavily, resulting in insufficient span semantic representations and poor model performance. To alleviate this problem, some span-based models [11,12] make attempts to incorporate other related NLP tasks into this task, such as event detection and coreference resolution. By using carefully designed neural architectures, these models enable span semantic representation to incorporate information shared from the added tasks. However, these additional tasks require extra data annotations such as event annotations, which are inaccessible in most datasets for the task, such as SciERC [6], DocRED [13], TACRED [14], NYT [15], WebNLG [16], SemEval [17], CoNLL04 [18], and ADE [19].

* Corresponding author (email: shashali@nudt.edu.cn, xuhao@nudt.edu.cn, yj@nudt.edu.cn, liuhuijun@nudt.edu.cn)

1) The span-based paradigm assigns the `NoneType` to spans that are not entities, as well as span pairs that do not hold relations.

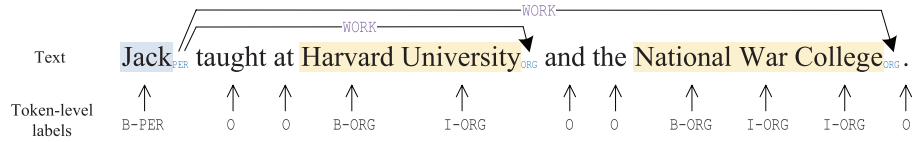

Figure 1 (Color online) A span-based joint extraction example, which contains three gold entities and two gold relations. Tokens in shade are span examples, `PER` and `ORG` are entity types, and `WORK` is a relation type. We also label the text with token-level labels via the sequence tagging mechanism, such as `B-PER` and `B-ORG`.

Previous sequence tagging-based joint models [2, 4, 20, 21] demonstrate that token-level labels convey critical information, which can be used to compensate for span-level semantic representations. For example, if a span-based model is aware that the "Jack" is a person entity (labeled with the `PER` label) and the "Harvard University" is an organization entity (labeled with the `ORG` label) beforehand, it may readily infer that they have a `WORK` relation. Unfortunately, as far as we know, existing span-based models neglect this critical information due to their inability to produce token-level labels. Additionally, existing sequence tagging-based models establish a unidirectional information flow from named entity recognition (NER) to relation extraction (RE) by using the token-level label information in the relation classification, hence enhancing information sharing. Due to the lack of token-level labels, previous span-based models are unable to build such an information flow, let alone a more effective bi-directional information interaction.

In this paper, we explore using the token-level label information in the span-based joint extraction, aiming to improve the performance of the span-based joint extraction. To this end, we propose a sequence tagging augmented span-based network (STSN) where the core module is a carefully designed neural architecture, which is achieved by deep stacking multiple attention layers. Specifically, the core architecture first learns three types of semantic representations: label representations for classifying token-level labels, and token representations for span-based NER and RE, respectively; it then establishes information interactions among the three learned representations. As a result, the two types of token representations can fully incorporate label information. Thus, span representations constructed with the above token representations are also enriched with label information. Additionally, the core architecture enables our model to build an effective bi-directional information interaction between NER and RE.

For the above purposes, each attention layer of the core architecture consists of three basic attention units. (1) Entity&relation to label attention (E&R-L-A) enables label representations to attend to the two types of token representations. The reason for doing this is two-fold: one is that E&R-L-A enables label representations to incorporate task-specific information effectively; the other is that E&R-L-A is essential to construct the bi-directional information interaction between NER and RE. (2) Label to entity attention (L-E-A) enables token representations for NER to attend to label representations with the goal of enriching the token representations with label information. (3) Label to relation attention (L-R-A) enables token representations for RE to attend to label representations with the goal of enriching the token representations with label information. In addition, we establish the bi-directional information interaction by taking the label representation as a medium, enabling the two types of token representations to attend to each other. We have validated the effectiveness of the bi-directional information interaction in Subsection 4.4.2. Moreover, to enable STSN to use token-level label information of overlapping entities, we extend the BIO tagging scheme and discuss more details in Subsection 4.1.2.

In STSN, aiming to train token-level label information in a supervised way, we add a sequence tagging-based NER decoder to the span-based model. And we use entities and relations extracted by the span-based model to evaluate the model performance. Experimental results on ACE05, CoNLL04, and ADE demonstrate that STSN consistently outperforms the strongest baselines in terms of F1, creating new state-of-the-art performance[2)].

In sum, we summarize the contributions as follows. (1) We propose an effective method to augment the span-based joint entity and relation extraction model with the sequence tagging mechanism. (2) We carefully design the deep-stacked attention layers, enabling the span-based model to use token-level label information and establish a bi-directional information interaction between NER and RE. (3) Experimental results on three datasets demonstrate that STSN creates new state-of-the-art results.

2) For reproducibility, our code for this paper will be publicly available at https://github.com/jibin/STSN.

2 Related work

2.1 Span-based joint extraction

Models for span-based joint entity and relation extraction have been widely studied. Luan et al. [6] proposed almost the first published span-based model, which is drawn from two models for coreference resolution [22] and semantic role labeling [23], respectively. With the advent of PLMs, span-based models directly take PLMs as their encoders, such as Dixit and Al-Onaizan [7] proposed a span-based model which takes ELMo [24] as the encoder; Eberts and Ulges [8] proposed SpERT, which takes BERT [25] as the encoder; Zhong and Chen [10] proposed PURE which takes ALBERT [26] as the encoder. However, these models rely heavily on the encoding ability of PLMs, leading to insufficient span semantic representations and finally resulting in poor model performance. Some models [11, 12] make attempts to alleviate this issue by adding additional NLP tasks to the task, such as coreference resolution or event detection. These models enable span semantic representations to incorporate information derived from the added tasks through complicated neural architectures. However, the added tasks need extra data annotations (such as event annotations are required in joint entity-relation extraction datasets), which are unavailable in most cases. Compared to these models, our model enriches span semantic representations with token-level label information without additional data annotations.

2.2 Token-level label

Numerous work has demonstrated that token-level label information benefits the joint extraction task a lot. For example, the models reported in [2–4, 20] train fixed-size semantic representations for token-level labels and use them in relation classification by concatenating them to relation semantic representations, delivering promising performance gains. However, Zhao et al. [21] demonstrated that the above shallow semantic concatenation cannot make full use of the label information. Therefore, they carefully design a deep neural architecture to capture fine-grained token-label interactions and deep infuse token-level label information into token semantic representations, delivering more promising performance gains. Unfortunately, previous span-based joint extraction models cannot benefit from the token-level label information since they completely give up the sequence tagging mechanism. In contrast, we propose a sequence tagging augmented span-based joint extraction model, which generates token-level label information via the sequence tagging mechanism and further infuses the information into token semantic representations via deep infusion.

3 Approach

In this section, we will describe the STSN in detail. As Figure 2 shows, STSN consists of three components: a BERT-based embedding layer, an encoder composed of deep-stacked attention layers, and three separate linear decoders for sequence tagging-based NER, span-based NER, and span-based RE, respectively.

3.1 Embedding layer

In STSN, we use BERT [25] as the default embedding generator. For a given text $\mathcal{T} = (t_1, t_2, t_3, \ldots, t_n)$ where t_i denotes the i-th token, BERT first tokenizes it with the WordPiece vocabulary [27] to obtain an input sequence. For each element of the sequence, its representation is the element-wise addition of WordPiece embedding, positional embedding, and segment embedding. Then a list of input embeddings $\boldsymbol{H} \in \mathbb{R}^{\text{len}\times\text{hid}}$ are obtained, where len is the sequence length and hid is the size of hidden units. A series of pre-trained Transformer [28] blocks are then used to project $\boldsymbol{H}$ into a BERT embedding sequence (denoted as $\boldsymbol{E}_{\mathcal{T}}$):

$$\boldsymbol{E}_{\mathcal{T}} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3, \ldots, \boldsymbol{e}_{\text{len}}\}. \tag{1}$$

BERT may tokenize one token into several sub-tokens to alleviate the out-of-vocabulary (OOV) problem, leading to that $\mathcal{T}$ cannot align with $\boldsymbol{E}_{\mathcal{T}}$, i.e., $n \neq \text{len}$. To achieve alignment, we propose an Align module, which applies the max-pooling function to the BERT embeddings of tokenized sub-tokens to obtain token embeddings. We define the aligned embedding sequence for $\mathcal{T}$ as

$$\hat{\boldsymbol{E}}_{\mathcal{T}} = \{\hat{\boldsymbol{e}}_1, \hat{\boldsymbol{e}}_2, \hat{\boldsymbol{e}}_3, \ldots, \hat{\boldsymbol{e}}_n\}, \tag{2}$$

where $\hat{\boldsymbol{E}}_{\mathcal{T}} \in \mathbb{R}^{n\times d}$ and d is the BERT embedding dimension. $\hat{\boldsymbol{e}}_i$ denotes the BERT embedding of t_i.

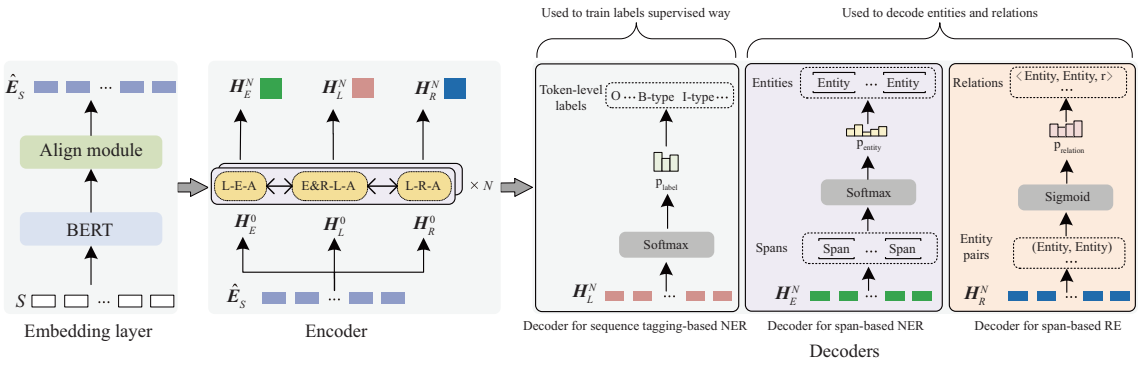

Figure 2 (Color online) The illustration of STSN, which consists of a BERT-based embedding layer, an encoder, and three separate linear decoders. We solely use the decoder for sequence tagging-based NER to train token-level label semantics ($\boldsymbol{H}_L$) in a supervised way. And entities and relations decoded by the span-based NER and RE decoders are used to evaluate model performance.

3.2 Encoder

The encoder is a deep neural architecture, which is achieved by stacking multiple (N) attention layers in depth.

3.2.1 *Deep neural architecture*

We deep stack multiple attention layers to build the deep neural architecture, where each layer is composed of three basic attention units, as shown in Figure 2.

The deep neural architecture learns three types of semantic representations: label representations (denoted as $\boldsymbol{H}_L$) used to classify token-level labels for sequence tagging-based NER, token representations (denoted as $\boldsymbol{H}_E$) for span-based NER, and token representations (denoted as $\boldsymbol{H}_R$) for span-based RE. The three representations have the same embedding dimension d. Additionally, we define the concatenation of $\boldsymbol{H}_E$ and $\boldsymbol{H}_R$ as $\boldsymbol{H}_C$ and convert its embedding dimension to d via a feed forward network (FFN):

$$\boldsymbol{H}_C = [\boldsymbol{H}_E; \boldsymbol{H}_R]\boldsymbol{W}_C + \boldsymbol{b}_C, \tag{3}$$

where $\boldsymbol{W}_C \in \mathbb{R}^{2d\times d}$ and $\boldsymbol{b}_C \in \mathbb{R}^d$ are trainable FFN parameters.

We formulate the first attention layer as follows:

$$[\text{Layer}]^1 = \begin{cases} \boldsymbol{H}_C^0 = [\boldsymbol{H}_E^0; \boldsymbol{H}_R^0]\boldsymbol{W}_C^0 + \boldsymbol{b}_C^0, \\ \boldsymbol{H}_L^1 = \text{E\&R-L-A}(\boldsymbol{H}_L^0, \boldsymbol{H}_C^0, \boldsymbol{H}_C^0), \\ \boldsymbol{H}_E^1 = \text{L-E-A}(\boldsymbol{H}_E^0, \boldsymbol{H}_L^1, \boldsymbol{H}_L^1), \\ \boldsymbol{H}_R^1 = \text{L-R-A}(\boldsymbol{H}_R^0, \boldsymbol{H}_L^1, \boldsymbol{H}_L^1), \end{cases} \tag{4}$$

where $\hat{\boldsymbol{E}}_{\mathcal{T}}$ is mapped to $\boldsymbol{H}_L^0$, $\boldsymbol{H}_E^0$, and $\boldsymbol{H}_R^0$, respectively. $\boldsymbol{H}_L^1$, $\boldsymbol{H}_E^1$, and $\boldsymbol{H}_R^1$ are the outputs of the first layer.

Then $\boldsymbol{H}_L^1$, $\boldsymbol{H}_E^1$, and $\boldsymbol{H}_R^1$ are passed to the next layer. We recursively repeat the above procedure until we obtain the outputs of the N-th layer, namely $\boldsymbol{H}_L^N$, $\boldsymbol{H}_E^N$, and $\boldsymbol{H}_R^N$. Now we assume that $\boldsymbol{H}_E^N$ and $\boldsymbol{H}_R^N$ have fully incorporated token-level label information. And they will be used for span-based NER and RE, respectively. $\boldsymbol{H}_L^N$ will be used to classify token BIO[3)] labels for sequence tagging-based NER.

As Figure 2 shows, we establish information interactions among the three types of representations in each attention layer. Specifically, $\boldsymbol{H}_E$ and $\boldsymbol{H}_L$ can interact with each other directly, as well as $\boldsymbol{H}_R$ and $\boldsymbol{H}_L$. Therefore by taking $\boldsymbol{H}_L$ as a medium, $\boldsymbol{H}_E$ and $\boldsymbol{H}_R$ can also interact with each other, which establishes a bi-directional information interaction between span-based NER and span-based RE in essence.

3.2.2 *Basic attention units*

As Figure 3 shows, the three types of basic attention units share a common neural architecture but differ in model inputs. The common architecture is composed of two sub-layers: multi-head attention and position-wise FFN. A residual connection is adopted around each sub-layer, followed by layer normalization.

3) 'B' denotes 'Beginning', 'I' denotes 'Inside', and 'O' denotes 'Outside'.

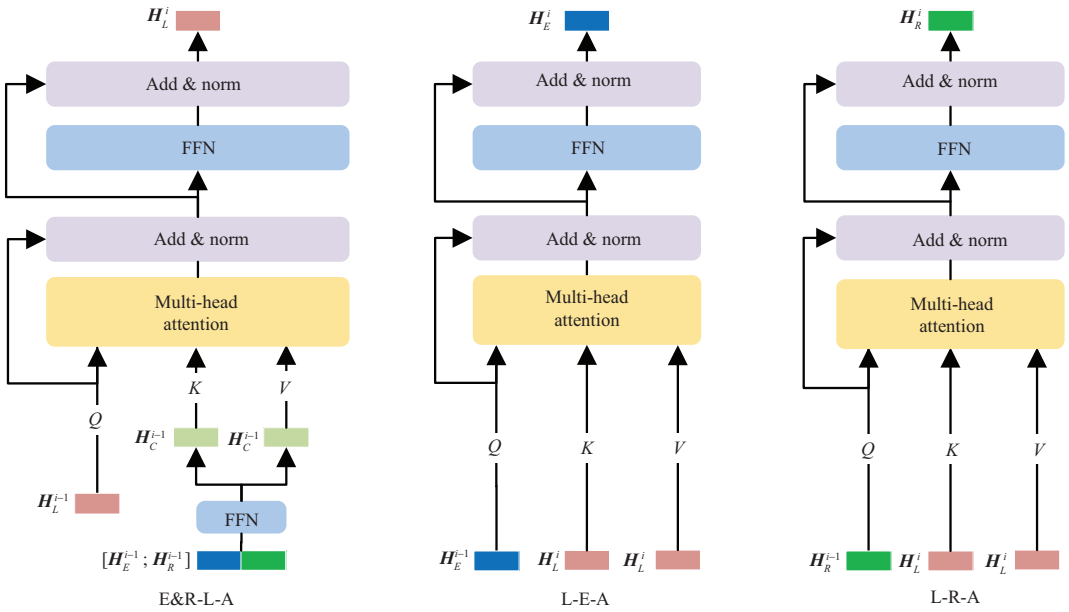

Figure 3 (Color online) Neural architectures of E&R-L-A, L-E-A, and L-R-A. The three units share a common architecture but differ in inputs.

Multi-head attention has been proven effective in capturing long-range dependencies by explicitly attending to all positions in various feature spaces. It has a series of h parallel heads and requires three inputs, i.e., Query ($\boldsymbol{Q}$), Key ($\boldsymbol{K}$) and Value ($\boldsymbol{V}$):

$$\text{head}^i = \text{softmax}\left(\frac{(\boldsymbol{Q}\boldsymbol{W}_Q^i)(\boldsymbol{K}\boldsymbol{W}_K^i)^{\mathrm{T}}}{\sqrt{d/h}}(\boldsymbol{V}\boldsymbol{W}_V^i)\right), \tag{5}$$

$$\boldsymbol{I} = \text{concat}(\text{head}^1, \ldots, \text{head}^h)\boldsymbol{W}_O, \tag{6}$$

where $\{\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}\} \in \mathbb{R}^{n\times d}$, $\{\boldsymbol{W}_Q^i, \boldsymbol{W}_K^i, \boldsymbol{W}_V^i\} \in \mathbb{R}^{d\times(d/h)}$, and $\boldsymbol{W}_O \in \mathbb{R}^{d\times d}$ are trainable parameters. $\boldsymbol{I} \in \mathbb{R}^{n\times d}$ is the output. Multi-head attention learns the pairwise relationship between $\boldsymbol{Q}$ and $\boldsymbol{K}$ and outputs weighted summation across all instances. Then residual connection conducts element-wise addition of $\boldsymbol{I}$ and $\boldsymbol{Q}$.

Position-wise FFN contains two linear transformations with a ReLU activation between them:

$$\text{FFN}(\boldsymbol{I}) = \max(0, \boldsymbol{I}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2, \tag{7}$$

where $\{\boldsymbol{W}_1, \boldsymbol{W}_2\} \in \mathbb{R}^{d\times d}$ and $\{\boldsymbol{b}_1, \boldsymbol{b}_2\} \in \mathbb{R}^d$ are trainable FFN parameters.

Figure 3 shows the detailed implementations of the three units. To be specific, (1) E&R-L-A takes $\boldsymbol{H}_L$ as $\boldsymbol{Q}$, and $\boldsymbol{H}_C$ as $\boldsymbol{K}$ and $\boldsymbol{V}$, respectively. It enables label representations to attend to the two types of token representations, aiming to make label representations incorporate task-specific information well. (2) L-E-A takes $\boldsymbol{H}_E$ as $\boldsymbol{Q}$, and $\boldsymbol{H}_L$ as $\boldsymbol{K}$ and $\boldsymbol{V}$, respectively. It enables token representations for span-based NER to attend to label representations, aiming to infuse label information into the token representations. (3) L-R-A takes $\boldsymbol{H}_R$ as $\boldsymbol{Q}$, and $\boldsymbol{H}_L$ as $\boldsymbol{K}$ and $\boldsymbol{V}$, respectively. It enables token representations for span-based RE to attend to label representations, aiming to infuse label information into the token representations.

3.3 Decoders

We design three separate linear decoders for sequence tagging-based NER, span-based NER and RE, respectively.

3.3.1 *Decoder for sequence tagging-based NER*

This encoder aims to train label representations in a supervised way. The decoder first uses an FFN to convert the embedding space of label representations (d) to the embedding space of BIO labels. It then

uses the softmax function to calculate probability distributions on the BIO label space:

$$\hat{\boldsymbol{y}}_L = \text{softmax}(\boldsymbol{H}_L^N \boldsymbol{W}_L + \boldsymbol{b}_L), \tag{8}$$

where $\boldsymbol{W}_L \in \mathbb{R}^{d*l}$ and $\boldsymbol{b}_L \in \mathbb{R}^l$ are trainable FFN parameters. l is the count of BIO label types.

The training objective is to minimize the following cross-entropy loss:

$$\mathcal{L}_L = -\frac{1}{M_L} \sum_{i=1}^{M_L} \boldsymbol{y}_L^i \log \hat{\boldsymbol{y}}_L^i, \tag{9}$$

where $\boldsymbol{y}_L$ is the one-hot vector of the gold token BIO label. M_L is the count of token-label instances.

3.3.2 *Decoder for span-based NER*

This decoder classifies span representations to obtain entities. These entities will be used for RE and model performance evaluation. We first add the `NoneEntity` type to the pre-defined entity types. Our model will be trained to classify spans into `NoneEntity` if they are not entities. We formulate the definition of span as

$$s = (t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+j}) \quad \text{s.t.} \;\; 1 \leqslant i \leqslant i+j \leqslant n, \tag{10}$$

where span width is restricted by a threshold ϵ and $j < \epsilon$. We obtain the span representation of s (denoted as $\boldsymbol{E}_s$) by concatenating semantic representations of span head and tail tokens, and the span width embedding:

$$\boldsymbol{E}_s = [\boldsymbol{H}_{E,i}^N; \boldsymbol{H}_{E,i+j}^N; \boldsymbol{W}_{j+1}], \tag{11}$$

where $\boldsymbol{H}_{E,i}^N$ and $\boldsymbol{H}_{E,i+j}^N$ are the i-th and $(i+j)$-th embeddings in $\boldsymbol{H}_E^N$. $\boldsymbol{W}_{j+1}$ is the fixed-size span width embedding, which is trained during model training.

$\boldsymbol{E}_s$ first passes through an FFN and then is fed into the softmax function, yielding a posterior on the space of entity types (including `NoneEntity`):

$$\hat{\boldsymbol{y}}_s = \text{softmax}(\boldsymbol{E}_s \boldsymbol{W}_s + \boldsymbol{b}_s), \tag{12}$$

where $\boldsymbol{W}_s$ and $\boldsymbol{b}_s$ are trainable FFN parameters. The training objective is to minimize the following cross-entropy loss:

$$\mathcal{L}_E = -\frac{1}{M_E} \sum_{i=1}^{M_E} \boldsymbol{y}_s^i \log \hat{\boldsymbol{y}}_s^i, \tag{13}$$

where $\boldsymbol{y}_s$ is the one-hot vector of the gold span type. M_E is the number of span instances.

We filter spans that are predicted as entities and build an entity set $\mathbb{S}_e$.

3.3.3 *Decoder for span-based RE*

This decoder classifies relation representations to obtain relations. These relations will be used for model performance evaluation. As relations exist between entities, only spans predicted as entities are used for the classification. We formulate the definition of ordered entity pairs (a.k.a. candidate relation tuple) as

$$r = \langle e_1, e_2 \rangle \quad \text{s.t.} \;\; e_1, e_2 \in \mathbb{S}_e, \;\; e_1 \neq e_2, \tag{14}$$

where e_1 and e_2 are the head and tail entities, respectively.

We obtain relation representations (denoted as $\boldsymbol{E}_r$) by concatenating semantic representations of head entity, tail entity, and relation context:

$$\boldsymbol{E}_r = [\boldsymbol{E}_{e_1}; \boldsymbol{E}_{e_2}; \boldsymbol{C}_r], \tag{15}$$

where $\boldsymbol{E}_{e_1}$ and $\boldsymbol{E}_{e_2}$ are semantics of e_1 and e_2, respectively. We obtain them using (11) with $\boldsymbol{H}_R^N$. Following the previous study [8], we obtain $\boldsymbol{C}_r$ by applying the max-pooling function to the embedding sequence of the relation context.

$\boldsymbol{E}_r$ first passes through an FFN and then is fed into the sigmoid function:

$$\hat{\boldsymbol{y}}_r = \sigma(\boldsymbol{E}_r \boldsymbol{W}_r + \boldsymbol{b}_r), \tag{16}$$

Figure 4 (Color online) An example of overlapping entities which are tagged by the extended BIO tagging scheme, where "Codeine" is the overlapping token, and `DRUG` and `AE` are entity types.

where σ is the sigmoid. $\boldsymbol{W}_r$ and $\boldsymbol{b}_r$ are trainable FFN parameters.

Any high response in the sigmoid outputs indicates that a corresponding relation is held between e_1 and e_2. Given a confidence threshold α, any relation with a score $\geqslant \alpha$ is considered activated.

The training objective is to minimize the following binary cross-entropy loss:

$$\mathcal{L}_R = -\frac{1}{M_R}\sum_{i=1}^{M_R}(\boldsymbol{y}_r^i \log \hat{\boldsymbol{y}}_r^i + (1-\boldsymbol{y}_r^i)\log(1-\hat{\boldsymbol{y}}_r^i)), \tag{17}$$

where $\boldsymbol{y}_r$ is the one-hot vector of the gold relation type. M_R is the number of entity pair instances.

3.3.4 *Model training*

During model training, we optimize the following joint training objective:

$$\mathcal{L}_{\text{joint}}(\boldsymbol{W};\theta) = \mathcal{L}_L + \mathcal{L}_E + \mathcal{L}_R. \tag{18}$$

4 Experiments

4.1 Experimental setup

4.1.1 *Datasets*

We evaluate STSN on ACE05 [29], CoNLL04 [18], and ADE [19] and use the same entity and relation types, data splits, and pre-processing following the established line of study [30]. Moreover, for a fair comparison with the previous study [8], we maintain a full version of the ADE dataset, which includes 119 instances containing overlapping entities.

4.1.2 *Extended BIO tagging scheme*

To make STSN use token-level label information of overlapping entities, we extend the BIO tagging scheme, which cannot tag overlapping entities initially. We begin by establishing two definitions.

- Definition 1. Two-fold overlapping entities. A pair of overlapping entities where the overlapping tokens are not contained in any other entities.
- Definition 2. Preceding entity. An entity with a preceding head location. If two entities have the same head location, the entity with a longer length is chosen.

Figure 4 gives a typical example: "Codeine" and "Codeine intoxication" are two-fold overlapping entities, and "Codeine intoxication" is the preceding entity.

The detailed tagging principle is that we first tag the preceding entity with the BIO tagging scheme. Then for the overlapping entity, we append its BIO labels to existing labels, separated by "/". For example, "Codeine" is tagged with `B-AE/B-DRUG`. As all overlapping entities in the full ADE dataset are two-fold, we tag the dataset with the extended BIO tagging scheme. For other datasets, we tag them with the BIO tagging scheme.

4.1.3 *Evaluation metrics*

Following the established line of study [9, 10], we use the standard precision (P), recall (R), and F1 to evaluate the model performance. For NER, a predicted entity is considered correct if its type and boundaries (entity head for ACE05) match the ground truth. For RE, we adopt two evaluation metrics: (1) A predicted relation is considered correct if the relation type and boundaries of the two entities match the ground truth. We define this metric as RE. (2) A predicted relation is considered correct if both the relation type and the two entities match ground truth. We define this metric as RE+. More discussion of evaluation settings can be found in [30].

Table 1 Model comparisons on ACE05 using the micro-averaged F1[a)]

Model	PLM	NER			RE			RE+		
		P	R	F1	P	R	F1	P	R	F1
Li and Ji [1]	–	85.2	76.9	80.8	68.9	41.9	52.1	65.4	39.8	49.5
Katiyar et al. [3]	–	84.0	81.3	82.6	57.9	54.0	55.9	55.5	51.8	53.6
Miwa et al. [2]	–	82.9	83.9	83.4	–	–	–	57.2	54.0	55.6
Sun et al. [31]	–	83.9	83.2	83.6	–	–	–	64.9	55.1	59.6
Li et al. [32]	BERT	84.7	84.9	84.8	–	–	–	64.8	56.2	60.2
Dixit and Al-Onaizan [7]	ELMo	85.9	86.1	86.0	68.0	58.4	62.8	–	–	–
Shen et al. [33]	BERT	87.7	87.5	87.6	–	–	–	62.2	63.7	62.8
Luan et al. [11]	–	–	–	88.4	–	–	–	–	–	63.2
Wadden et al. [12]	BERT	–	–	88.6	–	–	–	–	–	63.4
Lin et al. [5]	BERT	–	–	88.8	–	–	–	–	–	67.5
Wang and Lu [30]	ALBERT	–	–	89.5	–	–	67.6	–	–	64.3
Ji et al. [9]	BERT	89.3	89.9	89.6	–	–	–	**71.2**	60.2	65.2
Ren et al. [34]	ALBERT	–	–	89.9	–	–	–	–	–	68.0
Zhong et al. [10]	BERT	–	–	90.1	–	–	67.7	–	–	64.8
Zhong et al. [10]	ALBERT	–	–	90.9	–	–	69.4	–	–	67.0
STSN (ours)	BERT	90.9	89.9	90.4	77.8	60.7	68.2	69.4	64.4	66.8
STSN (ours)	ALBERT	**92.7**	**90.5**	**91.6**	**80.2**	**64.2**	**71.3**	69.5	**68.7**	**69.1**

a) Bold values denote the state-of-the-art results.

4.1.4 *Implementation details*

We build STSN by deep stacking three attention layers and evaluate it with `bert-base-cased` [25] and `albert-xxlarge-v1` [26] on a single NVIDIA RTX 3090 GPU. We optimize STSN using AdamW for 100 epochs with a learning rate of 5E−5, a linear scheduler with a warmup ratio of 0.1, and a weight decay of 1E−2. We set the training batch size to 4, dimension of $\boldsymbol{W}_{j+1}$ to 150, h of multi-head attention to 8, span width threshold ϵ to 10, and relation threshold α to 0.4. Following the established line of study [8,9], we adopt a negative sampling strategy and set the number of the negative entity and relation samples per data entry to 100, respectively.

Across all the three datasets, we use the training set to train STSN and use the test set to report model evaluation performance. For ACE05 and CoNLL04, we run STSN 20 times and report averaged results of the best 5 runs. For ADE, we adopt the 10-fold cross-validation, run each fold 20 times, and report averaged results of the best 5 runs.

4.2 Main results

Tables 1–3 [31–42] show the model comparison results. We have the following observations. (1) Our best model consistently surpasses all the selected baselines in terms of F1. (2) On ACE05, compared to the strongest baselines [10, 34], our best model obtains +0.7%, +1.9%, and +1.1% F1 gains on NER, RE, and RE+, respectively. (3) On CoNLL04, compared to the strongest baselines [21, 38], our best model obtains +1.0% and +1.7% micro-averaged F1 gains on NER and RE+, respectively. And compared to the strongest baselines [8,30], our model obtains +2.5% and +3.1% macro-averaged F1 gains on NER and RE, respectively. (4) On ADE (without overlapping entities), compared to the strongest baseline [39], our best model obtains +1.0% and +1.3% F1 gains on NER and RE, respectively. (5) On the full ADE (with overlapping entities), compared to the strongest baseline [40], our best model obtains +1.1% and +2.7% F1 gains on NER and RE, respectively.

We attribute these performance gains to (1) the success of using token-level label information in span-based joint extraction; (2) the bi-directional information interaction between NER and RE; (3) the effectiveness of the extended BIO tagging scheme. Additionally, we report concrete positive and negative case studies to help understand our model, as shown in Subsection 4.5.

Table 2 Model comparisons on CoNLL04[a)]

Model	PLM	NER			RE+		
		P	R	F1	P	R	F1
Bekoulis et al. [20] ▲	–	83.4	84.1	83.9	63.8	60.4	62.0
Nguyen et al. [35] ▲	–	–	–	86.2	–	–	64.4
Eberts et al. [8] ▲	BERT	85.8	86.8	86.3	74.8	71.5	72.9
Wang and Lu [30] ▲	ALBERT	–	–	86.9	–	–	75.4
Miwa et al. [36] △	–	81.2	80.2	80.7	76.0	50.9	61.0
Zhang et al. [37] △	–	–	–	85.6	–	–	67.8
Li et al. [32] △	BERT	89.0	86.6	87.8	69.2	68.2	68.9
Eberts et al. [8] △	BERT	88.3	89.6	88.9	73.0	70.0	71.5
Wang and Lu [30] △	ALBERT	–	–	90.1	–	–	73.6
Ji et al. [9] △	BERT	90.1	90.4	90.2	77.0	71.9	74.3
Shen et al. [33] △	BERT	90.3	90.3	90.3	73.0	71.6	72.4
Zhao et al. [21] △	ELMO	–	–	90.6	–	–	73.0
Huguet et al. [38] △	BART	–	–	–	75.6	75.1	75.4
STSN (Ours) △	BERT	90.6	**91.2**	90.9	76.1	73.9	75.0
STSN (ours) △	ALBERT	**92.4**	90.8	**91.6**	**76.8**	**77.4**	**77.1**
STSN (ours) ▲	BERT	88.5	87.9	88.2	77.5	77.1	77.3
STSN (ours) ▲	ALBERT	**89.8**	**89.0**	**89.4**	**79.0**	**78.0**	**78.5**

a) △ denotes using the micro-averaged F1; ▲ denotes using the macro-averaged F1; bold values denote the state-of-the-art results.

Table 3 Model comparisons on ADE[a)]

Model	PLM	NER			RE+		
		P	R	F1	P	R	F1
Eberts et al. [8] ♠	BERT	89.0	89.6	89.3	77.8	78.0	78.8
Ji et al. [9] ♠	BERT	89.9	91.3	90.6	79.6	81.9	80.7
Lai et al. [40] ♠	BERT	–	–	90.7	–	–	81.7
Li et al. [41]	–	79.5	79.6	79.5	64.0	62.9	63.4
Li et al. [42]	–	82.7	86.7	84.6	67.5	75.8	71.4
Bekoulis et al. [20]	–	84.7	88.2	86.4	72.1	77.2	74.6
Eberts et al. [8]	BERT	89.3	89.3	89.3	78.1	80.4	79.2
Zhao et al. [21]	ELMo	–	–	89.4	–	–	81.1
Yan et al. [39]	BERT	–	–	89.6	–	–	80.0
Wang and Lu [30]	ALBERT	–	–	89.7	–	–	80.1
Shen et al. [33]	BERT	89.5	91.3	90.4	84.2	83.4	80.7
Huguet et al. [38]	BART	–	–	–	81.5	83.1	82.2
Yan et al. [39]	ALBERT	–	–	91.3	–	–	83.2
STSN (ours)	BERT	91.3	91.9	91.6	82.8	**84.6**	83.7
STSN (ours)	ALBERT	**91.5**	**93.1**	**92.3**	**84.8**	84.2	**84.5**
STSN (ours) ♠	BERT	90.8	91.4	91.1	83.3	83.7	83.5
STSN (ours) ♠	ALBERT	**91.6**	**92.0**	**91.8**	**85.0**	**83.8**	**84.4**

a) ♠ denotes evaluating STSN on the full ADE dataset (with overlapping entities); bold values denote the state-of-the-art results.

4.3 Analysis

We report analysis results on the dev set of ACE05 and the test sets of CoNLL04 and ADE[4)]. And we take SpERT [8] as the baseline, which is the closest span-based model to ours. SpERT uses BERT as the encoder and two linear decoders to classify spans and span pairs. For a fair comparison, we use our BERT-based STSN.

4) Following previous studies [8, 9, 21, 30], we combine the training and dev sets of CoNLL04 to train our STSN. Thus we use the test set for the analysis. And since ADE does not contain a dev set, we also use the test set for the analysis.

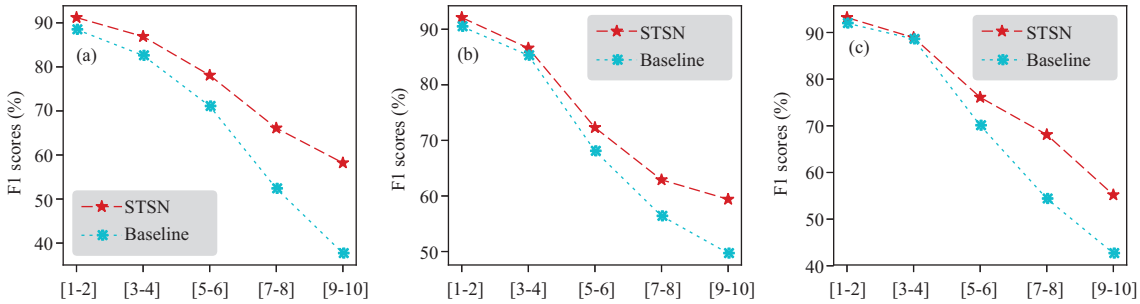

Figure 5 (Color online) NER performance comparisons under various grouped entity lengths across the three datasets. (a) ACE05; (b) CoNLL04; (c) ADE.

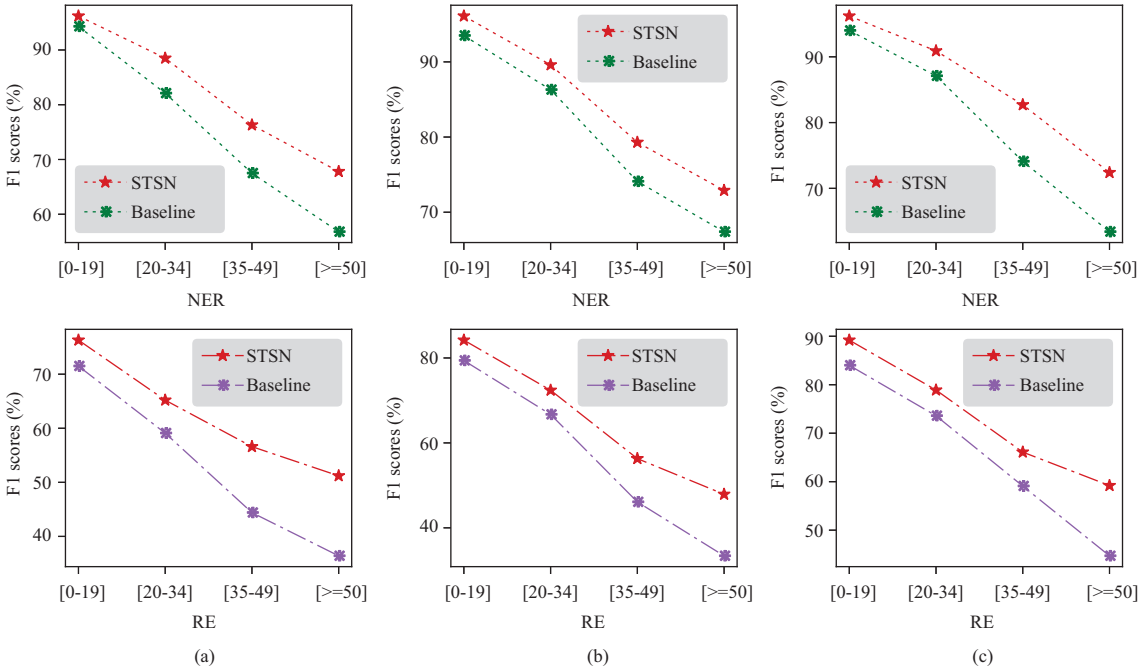

Figure 6 (Color online) NER and RE performance comparisons under various grouped text lengths across the three datasets. (a) ACE05; (b) CoNLL04; (c) ADE.

4.3.1 *Performance against entity length*

Figure 5 shows performance comparisons on NER under various entity lengths. We divide all entity lengths, which is restricted by span width threshold ϵ, into [1–2], [3–4], [5–6], [7–8], and [9–10]. We have the following observations: across all datasets, (1) STSN consistently outperforms the baseline under all length intervals; (2) performance improvements brought by STSN are generally further enhanced when the entity length increases. Specifically, STSN obtains much higher F1 gains under [7–8] and [9–10] than the ones under [1–2] and [3–4], demonstrating that STSN is more effective in terms of long entities.

4.3.2 *Performance against text length*

We compare STSN with the baseline under grouped text lengths. As Figure 6 shows, we divide text lengths into [0–19], [20–34], [35–49], and [$\geqslant$50]. We have the following observations: across the three datasets, (1) STSN performs way better than the baseline under all text lengths on both NER and RE; (2) performance gains brought by STSN are generally further enhanced when text length increases. In particular, STSN obtains the best performance gains under [$\geqslant$50] on both NER and RE, demonstrating that STSN is more effective in terms of long texts.

Table 4 Ablation study on attention layer numbers. We solely report the F1 scores and consider the averaged score of the 6 F1 scores in each row to be Ave. F1, which is used as an overall evaluation metric[a)].

STSN + deep stacking	ACE05		CoNLL04		ADE		Ave. F1
	NER	RE+	NER	RE+	NER	RE+	
`1 AttentionLayer`	87.6	59.2	87.4	72.1	88.9	81.1	79.4
`2 AttentionLayers`	88.7	60.5	90.0	73.9	89.5	80.4	80.5
`3 AttentionLayers`	89.5	62.6	90.9	75.0	91.6	83.7	**82.2**
`4 AttentionLayers`	89.2	62.5	91.3	75.2	90.5	83.8	82.1
`5 AttentionLayers`	88.9	62.6	90.4	74.2	90.7	83.2	81.7
`6 AttentionLayers`	89.1	62.0	90.4	74.4	90.5	82.9	81.6

a) The bold value denotes the best result.

4.4 Ablation study

We conduct ablation studies on our BERT-based STSN and report ablation results on the dev set of ACE05 and the test sets of CoNLL04 and ADE.

4.4.1 *Ablations on various attention layers*

We conduct ablations on attention layer numbers by deep stacking various attention layers in STSN. Table 4 shows the ablation results, from which we can observe that: across the three datasets, (1) when deep stacking three attention layers, STSN performs the best (82.2% Ave. F1); (2) STSN with only one attention layer performs the worst, which we attribute to the fact that one layer cannot fully infuse token-level label information into token semantic representations; (3) when the number of attention layers increases, the model performance generally first drastically increases and then slightly decreases. We attribute this to the fact that deeper models make it easier to fully infuse token-level label information into token semantic representations, while much deeper models tend to infuse more noisy information, which harms the model performance.

4.4.2 *Ablations on model components*

Table 5 reports the ablation results across the three datasets.

(1) "`w/o Label`" denotes ablating token-level label information. We realize this ablation by removing the stacked attention layers and the decoder for sequence tagging-based NER from STSN. After doing this, our model cannot benefit from the token-level label information. The results show that using the token-level label information boosts the model performance by delivering +2.7% to +3.1% F1 gains on NER and +4.2% to +6.0% F1 gains on RE+.

(2) "`w/o Bi-Interaction`†" denotes removing the information flow from RE to NER but keeping the information flow from NER to RE, as shown in Figure 7(a). We realize this ablation by making $\boldsymbol{K}$ and $\boldsymbol{V}$ of E&R-L-A be $\boldsymbol{H}_E$ and $\boldsymbol{Q}$ of E&R-L-A be $\boldsymbol{H}_L$. Thus $\boldsymbol{H}_L$ no longer attends to $\boldsymbol{H}_R$ and solely attends to $\boldsymbol{H}_E$. The results show that the information flow from RE to NER brings +1.1% and +0.8% averaged F1 gains on NER and RE, respectively.

(3) "`w/o Bi-Interaction`‡" denotes removing the information flow from NER to RE but keeping the information flow from RE to NER, as shown in Figure 7(b). We realize this ablation by making $\boldsymbol{K}$ and $\boldsymbol{V}$ of E&R-L-A be $\boldsymbol{H}_R$ and $\boldsymbol{Q}$ of E&R-L-A be $\boldsymbol{H}_L$. Thus $\boldsymbol{H}_L$ no longer attends to $\boldsymbol{H}_E$ and solely attends to $\boldsymbol{H}_R$. The results show that the information flow from NER to RE brings +0.4% and +1.3% averaged F1 gains on NER and RE, respectively.

(4) "`w/o Interaction`" denotes removing the information interactions between NER and RE, as shown in Figure 7(c). We realize this ablation by making $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ of E&R-L-A be $\boldsymbol{H}_L$. In other words, E&R-L-A is the self-attention in the current scenario, disabling the information interactions between NER and RE. The results show that the bi-directional information interactions bring +1.2% and +1.5% averaged F1 gains on NER and RE, respectively.

Based on these observations, we can conclude that the performance gains mainly benefit from using the token-level label information, revealing that our motivation is sufficient. Moreover, the bi-directional information interaction is consistently superior to the two unidirectional information flows, validating the effectiveness of our novel bi-directional design.

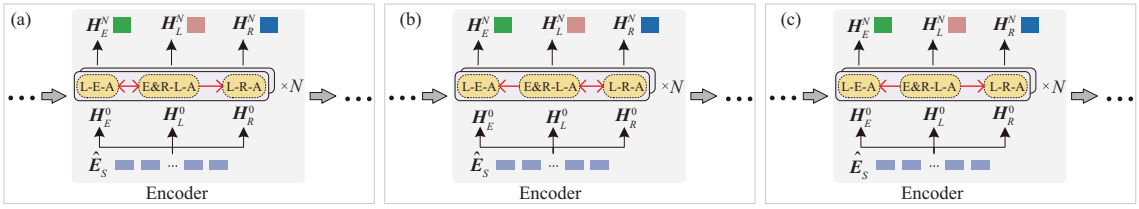

Figure 7 (Color online) Removing the information (a) from RE to NER; (b) from NER to RE; (c) interactions, as the red lines shown.

Table 5 Ablation results. We solely report the F1 scores. The values in parentheses denotes the F1 score decreases (compared to STSN) caused by corresponding ablation settings.

Method	ACE05		CoNLL04		ADE	
	NER	RE+	NER	RE+	NER	RE+
STSN	89.5	62.6	90.9	75.0	91.6	83.7
`w/o Label`	86.4 (−3.1)	56.6 (−6.0)	88.1 (−2.7)	70.8 (−4.2)	88.7 (−2.9)	78.5 (−5.2)
`w/o Bi-Interaction`†	89.0 (−0.5)	61.6 (−1.0)	89.6 (−1.3)	74.4 (−0.6)	90.1 (−1.5)	82.9 (−0.8)
`w/o Bi-Interaction`‡	89.2 (−0.3)	61.4 (−1.2)	90.2 (−0.7)	73.5 (−1.5)	91.5 (−0.1)	82.4 (−1.3)
`w/o Interaction`	88.9 (−0.6)	61.7 (−0.9)	89.5 (−1.4)	73.4 (−1.6)	90.1 (−1.5)	81.6 (−2.1)

4.5 Case study

We conduct qualitative analysis on concrete examples to help understand our model. We take SpERT as the baseline, which is the closest span-based model to ours. For a fair comparison, we use our BERT-based STSN.

4.5.1 *Positive example*

Table 6 reports four positive examples. In Text 1, SpERT mistakenly predicts "House of Delegates" as a `LOC` entity, while STSN correctly predicts it as a `ORG` entity. We attribute it to the fact that STSN enables span representations to incorporate token-level label information in the case that STSN correctly tags "House of Delegates" with `ORG` labels. Moreover, STSN correctly predicts that ⟨"House of Delegates", "Maryland"⟩ holds a `OrgBased_In` relation. Text 2 shows a similar example, where STSN correctly predicts "La." as a `LOC` entity, as well as the `Located_In` relation hold by ⟨"Grand Isle", "La."⟩.

Texts 3 and 4 mainly show the effects of using token-level label information in relation classification. For example, both SpERT and STSN correctly predict all entities of Text 3, but SpERT mistakenly predicts that there is no relation between these entities. In contrast, STSN correctly predicts the two `Located_In` relations. We attribute it to using token-level label information in relation representations, enabling our model to know detailed entity types beforehand.

4.5.2 *Negative example*

We also report a negative example, as Table 7 shows. In this example, STSN mistakenly tags a token label: "president" is tagged with `I-ORG`, which is supposedly tagged with `O`. However, STSN still correctly predicts all entities and relations of this Text. Moreover, we find that STSN successfully tackles most of the similar cases (97.56%) across the three datasets. We attribute it to the fact that STSN learns only to incorporate useful label information, enabling our model to avoid suffering from wrong label predictions.

5 Conclusion

In this paper, we propose an STSN for the joint entity and relation extraction task. STSN enables the span-based joint extraction model to use token-level label information, which is achieved by deep stacking multiple attention layers. Moreover, STSN establishes bi-directional information interactions between NER and RE, which is proven effective. Furthermore, we extend the BIO tagging scheme, allowing STSN to use the label information of overlapping entities. Experiments on three datasets show that STSN consistently outperforms other competing models in terms of F1. Since STSN only considers

Table 6 Positive examples regarding using token-level label information in the span-based joint extraction, where all labels, entities, and relations in the STSN rows are predicted correctly[a)]

Text 1		**Judith Toth** says she returned for a fourth term in **Maryland**'s **House of Delegates**
SpERT	Entity	[Judith Toth]$_{\texttt{PER}}$ [Maryland]$_{\texttt{LOC}}$ [House of Delegates]$_{\underline{\texttt{LOC}}}$
	Relation	⟨House of Delegates, Maryland, Located_In⟩
STSN	Token label	B-PER I-PER O O O O O O O B-LOC B-ORG I-ORG I-ORG
	Entity	[Judith Toth]$_{\texttt{PER}}$ [Maryland]$_{\texttt{LOC}}$ [House of Delegates]$_{\texttt{ORG}}$
	Relation	⟨House of Delegates, Maryland, OrgBased_In⟩
Text 2		One man was lost from an oil rig off **Grand Isle**, **La.**, as the storm moved in
SpERT	Entity	[Grand Isle]$_{\texttt{LOC}}$
	Relation	No relation
STSN	Token label	O O O O O O O O B-LOC I-LOC B-LOC O O O O O
	Entity	[Grand Isle]$_{\texttt{LOC}}$ [La.]$_{\texttt{LOC}}$
	Relation	⟨Grand Isle, La., Located_In⟩
Text 3		**Seattle** has a hour-glass figure, squeezed between **Puget Sound** and **Lake Washington**
SpERT	Entity	[Seattle]$_{\texttt{LOC}}$ [Puget Sound]$_{\texttt{LOC}}$ [Lake Washington]$_{\texttt{LOC}}$
	Relation	No relation
STSN	Token label	B-LOC O O O O O O B-LOC I-LOC O B-LOC I-LOC
	Entity	[Seattle]$_{\texttt{LOC}}$ [Puget Sound]$_{\texttt{LOC}}$ [Lake Washington]$_{\texttt{LOC}}$
	Relation	⟨Puget Sound, Seattle, Located_In⟩ , ⟨Lake Washington, Seattle, Located_In⟩
Text 4		An enraged **Khrushchev** instructed **Soviet** ships to ignore **Kennedy**'s naval blockade
SpERT	Entity	[Khrushchev]$_{\texttt{PER}}$ [Soviet]$_{\texttt{LOC}}$ [Kennedy]$_{\texttt{PER}}$
	Relation	No relation
STSN	Token label	O O B-PER O B-LOC O O O B-PER O O
	Entity	[Khrushchev]$_{\texttt{PER}}$ [Soviet]$_{\texttt{LOC}}$ [Kennedy]$_{\texttt{PER}}$
	Relation	⟨Khrushchev, Soviet, Live_In⟩

a) The underline denotes that entities or relations are mistakenly predicted, and the bold denotes corresponding entities located in texts.

Table 7 A negative example, in which STSN mistakenly predicts a token label (i.e., the I-ORG), but it still correctly predicts all entities and relations

Text		But **Jack Frazier**, **Rotary Club** president, said volunteers picked up the ducks
SpERT	Entity	[Jack Frazier]$_{\texttt{PER}}$ [Rotary Club]$_{\texttt{ORG}}$
	Relation	⟨Jack Frazier, Rotary Club, Work_For⟩
STSN	Token label	O B-PER I-PER B-ORG I-ORG I-ORG O O O O O O
	Entity	[Jack Frazier]$_{\texttt{PER}}$ [Rotary Club]$_{\texttt{ORG}}$
	Relation	⟨Jack Frazier, Rotary Club, Work_For⟩

the two-fold overlapping entities, we will investigate upgrading our model in the future to extract other overlapping entities.

Acknowledgements This work was supported by Hunan Provincial Natural Science Foundation (Grant Nos. 2022JJ30668, 2022JJ30046).

References

1 Li Q, Ji H. Incremental joint extraction of entity mentions and relations. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, 2014. 402–412
2 Miwa M, Bansal M. End-to-end relation extraction using LSTMs on sequences and tree structures. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, 2016. 1105–1116
3 Katiyar A, Cardie C. 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, Vancouver, 2017. 917–928
4 Ye W, Li B, Xie R, et al. Exploiting entity BIO tag embeddings and multi-task learning for relation extraction with imbalanced data. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, 2019. 1351–1360
5 Lin Y, Ji H, Huang F, et al. A joint neural model for information extraction with global features. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020. 7999–8009
6 Luan Y, He L, Ostendorf M, et al. Multi-task identification of entities, relations, and coreference for scientific knowledge

graph construction. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, 2018. 3219–3232

7 Dixit K, Al-Onaizan Y. Span-level model for relation extraction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, 2019. 5308–5314

8 Eberts M, Ulges A. Span-based joint entity and relation extraction with transformer pre-training. In: Proceedings of the 24th European Conference on Artificial Intelligence, Santiago de Compostela, 2020. 1–8

9 Ji B, Yu J, Li S, et al. Span-based joint entity and relation extraction with attention-based span-specific and contextual semantic representations. In: Proceedings of the 28th International Conference on Computational Linguistics, 2020. 88–99

10 Zhong Z, Chen D. A frustratingly easy approach for entity and relation extraction. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2021. 50–61

11 Luan Y, Wadden D, He L, et al. A general framework for information extraction using dynamic span graphs. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, 2019. 3036–3046

12 Wadden D, Wennberg U, Luan Y, et al. Entity, relation, and event extraction with contextualized span representations. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, 2019. 5784–5789

13 Yao Y, Ye D, Li P, et al. DocRED: a large-scale document-level relation extraction dataset. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, 2019. 764–777

14 Zhang Y, Zhong V, Chen D, et al. Position-aware attention and supervised data improve slot filling. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, 2017. 35–45

15 Riedel S, Yao L, McCallum A. Modeling relations and their mentions without labeled text. In: Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Berlin, 2010. 148–163

16 Zeng X, Zeng D, He S, et al. 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, Melbourne, 2018. 506–514

17 Hendrickx I, Kim S, Kozareva Z, et al. SemEval-2010 task 8: multi-way classification of semantic relations between pairs of nominals. In: Proceedings of the 5th International Workshop on Semantic Evaluation, Uppsala, 2010. 33–38

18 Roth D, Yih W. A linear programming formulation for global inference in natural language tasks. In: Proceedings of the 8th Conference on Computational Natural Language Learning at HLT-NAACL, Boston, 2004. 1–8

19 Gurulingappa H, Rajput A M, Roberts A, et al. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. J Biomed Inf, 2012, 45: 885–892

20 Bekoulis G, Deleu J, Demeester T, et al. Joint entity recognition and relation extraction as a multi-head selection problem. Expert Syst Appl, 2018, 114: 34–45

21 Zhao S, Hu M, Cai Z, et al. Modeling dense cross-modal interactions for joint entity-relation extraction. In: Proceedings of the 29th International Joint Conference on Artificial Intelligence, 2020. 4032–4038

22 Lee K, He L, Zettlemoyer L. Higher-order coreference resolution with coarse-to-fine inference. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, 2018. 687–692

23 He L, Lee K, Levy O, et al. Jointly predicting predicates and arguments in neural semantic role labeling. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, 2018. 364–369

24 Peters M, Neumann M, Iyyer M, et al. Deep contextualized word representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, 2018. 2227–2237

25 Devlin J, Chang M, Lee K, et al. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, 2019. 4171–4186

26 Lan Z, Chen M, Goodman S, et al. ALBERT: a lite BERT for self-supervised learning of language representations. In: Proceedings of the 8th International Conference on Learning Representations, 2020. 1–17

27 Wu Y, Schuster M, Chen Z, et al. Google's neural machine translation system: bridging the gap between human and machine translation. 2016. ArXiv:1609.08144

28 Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Proceedings of the 31st Conference on Neural Information Processing Systems, Long Beach, 2017. 5998–6008

29 Doddington G, Mitchell A, Przybocki M, et al. The automatic content extraction (ACE) program — tasks, data, and evaluation. In: Proceedings of the 4th International Conference on Language Resources and Evaluation, Lisbon, 2004. 837–840

30 Wang J, Lu W. Two are better than one: joint entity and relation extraction with table-sequence encoders. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, 2020. 1706–1721

31 Sun C, Wu Y, Lan M, et al. Extracting entities and relations with joint minimum risk training. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, 2018. 2256–2265

32 Li X, Yin F, Sun Z, et al. Entity-relation extraction as multi-turn question answering. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, 2019. 1340–1350

33 Shen Y, Ma X, Tang Y, et al. A trigger-sense memory flow framework for joint entity and relation extraction. In: Proceedings of the Web Conference 2021, 2021. 1704–1715

34 Ren L, Sun C, Ji H, et al. HySPA: hybrid span generation for scalable text-to-graph extraction. In: Findings of the Association for Computational Linguistics: ACL-IJCNLP, 2021. 4066–4078

35 Nguyen D Q, Verspoor K. End-to-end neural relation extraction using deep biaffine attention. In: Proceedings of the 41st European Conference on Information Retrieval, Cologne, 2019. 729–738

36 Miwa M, Sasaki Y. Modeling joint entity and relation extraction with table representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, 2014. 1858–1869

37 Zhang M, Zhang Y, Fu G. End-to-end neural relation extraction with global optimization. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, 2017. 1730–1740

38 Huguet P, Navigli R. REBEL: relation extraction by end-to-end language generation. In: Findings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, 2021. 2370–2381

39 Yan Z, Zhang C, Fu J, et al. A partition filter network for joint entity and relation extraction. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Punta Cana, 2021. 185–197

40 Lai T, Ji H, Zhai C, et al. Joint biomedical entity and relation extraction with knowledge-enhanced collective inference. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, 2021. 6248–6260

41 Li F, Zhang Y, Zhang M, et al. Joint models for extracting adverse drug events from biomedical text. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence, New York, 2016. 2838–2844

42 Li F, Zhang M, Fu G, et al. A neural joint model for entity and relation extraction from biomedical text. BMC Bioinf, 2017, 18: 198