Span-based Joint Entity and Relation Extraction with Attention-based Span-specific and Contextual Semantic Representations

Bin Ji[†], **Jie Yu**[†], **Shasha Li**[*], **Jun Ma, Qingbo Wu, Yusong Tan, Huijun Liu**[*]
College of Computer,
National University of Defense Technology, Changsha, China
{jibin, yj, shashali, majun}@nudt.edu.cn
{qingbowu, yusongtan, liuhuijun}@nudt.edu.cn

Abstract

Span-based joint extraction models have shown their efficiency on entity recognition and relation extraction. These models regard text spans as candidate entities and span tuples as candidate relation tuples. Span semantic representations are shared in both entity recognition and relation extraction, while existing models cannot well capture semantics of these candidate entities and relations. To address these problems, we introduce a span-based joint extraction framework with attention-based semantic representations. Specially, attentions are utilized to calculate semantic representations, including span-specific and contextual ones. We further investigate effects of four attention variants in generating contextual semantic representations. Experiments show that our model outperforms previous systems and achieves state-of-the-art results on ACE2005, CoNLL2004 and ADE.

1 Introduction

This paper considers intra-sentence joint entity and relation extraction. For joint extraction mode can alleviate cascading errors and promote information utilization compared to pipelined one, this mode has drawn much attention. Typically, the joint extraction task is solved by sequence tagging based methods (Zheng et al., 2017; Chi et al., 2019).

Rather than using sequence tagging based methods, recent works attempt to solve the task with span-based joint extraction mode (Dixit and Al-Onaizan, 2019; Luan et al., 2019). Typically, this mode first processes sentence text into text spans, which are `span-based candidate entities` (**"spans" for short**); then, calculates span semantic representations and performs span classification on them to obtain predicted entities; next, forms `span-based candidate relation` (**"relation" for short**) tuples with spans, and calculates relation semantic representations with corresponding span semantic representations; at last, performs relation classification on relation semantic representations and obtains predicted relation triples. This mode further improves joint extraction performance, whereas exists three problems.

First, different tokens in span should contribute differently to span representation, which we call span-specific features. But existing methods treat each span token equally important (Eberts and Ulges, 2019) or just consider span head and tail tokens (Dixit and Al-Onaizan, 2019), ignoring these significant features. Take the span *"a Palestinian youth"* in sentence 1 of Figure 1 as an example, the *"youth"* should contribute much greater to the span representation than *"a"* and *"Palestinian"* when classifying the span into **"PER"**. Second, local contextual information of relation tuples is omitted (Luan et al., 2018; Dixit and Al-Onaizan, 2019) or just calculated by max pooling way (Eberts and Ulges, 2019) when performing relation classification, which do not sufficiently capture information contained in it. Whereas local context may contain crucial information to help predict the relations that relation tuples hold. A case study is shown in sentence 2 of Figure 1, the *"ownership"* (in red font) can greatly help to determine the relation (**"PART-WHOLE"**) of the relation tuple (namely

† indicates equal contribution
∗ corresponding author

Proceedings of the 28th International Conference on Computational Linguistics, pages 88–99
Barcelona, Spain (Online), December 8-13, 2020

Sentence 1: The army said troops fired, and hit a boy, after [**a Palestinian youth**]PER threw a stone,⋯
Sentence 2: The weak score is primarily the result of [**Starbucks**]ORG current strategy of increasing equity **ownership** of [**several foreign subsidiaries**]ORG,⋯ **Relation**: <several foreign subsidiaries, Starbucks, **PART-WHOLE**>
Sentence 3: [Palestinians]GPE claim [**all of the West Bank and Gaza**]LOC for a **state**,⋯ **Relation**: <all of the West Bank and Gaza, Palestinians, **PART-WHOLE**>

Figure 1: Sentence examples including gold entities and gold relation triples from the ACE2005 dataset, where "PER", "ORG" etc., denote gold entity types, "PART-WHOLE" denotes gold relation type, texts in bracket are spans of gold entities and underlined texts are local contexts of gold relation tuples, "Relation" denotes gold relation triple.

<*several foreign subsidiaries*, *Starbucks*>). Third, sentence-level contextual information is ignored in both span and relation classifications, which may be important compensation information for both ones. An example is given in sentence 3 of Figure 1, *"state"* (in red font) benefits the relation classification of the relation tuple (namely <*all of the West Bank and Gaza*, *Palestinians*>), while *"state"* is neither contained in the relation tuple nor in the local context (namely "claim"), but contained in other part of sentence 3, which is sentence-level.

To address above issues, we introduce a span-based joint extraction model with attention-based span-specific and contextual semantic representations. Specifically, 1) MLP attention is used to calculate span-specific semantic representation; 2) attention-based sentence-level contextual semantic representation for span is calculated by taking span-specific semantic representation as query and sentence token sequence semantic representations as key, value respectively; 3) local and sentence-level contextual semantic representations for relation are obtained by attention calculation with relation tuple semantic representations and corresponding token sequence semantic representations. The advantage of this approach is that we can capture the most useful information to constitute efficient span and relation semantic representations.

We take BERT (Devlin et al., 2019) as the default backbone network, and explore the three research questions. Moreover, we investigate effects of Multi-Head attention (Vaswani et al., 2017), Dot-Product attention (Luong et al., 2015), General attention (Luong et al., 2015) and Additive attention (Bahdanau et al., 2015) in generating contextual semantic representations. Extensive experiments on three benchmark datasets show that our model consistently outperforms previous systems. In addition, the Multi-Head attention firmly improves over other attention variants.

2 Related Works

Traditionally, pipelined entity and relation extraction method is divided into two subtasks, namely entity detection and relation classification. Various neural networks have been widely investigated for the two subtasks, such as RNNs (Huang et al., 2015; Ma and Hovy, 2016), CNNs (Limsopatham and Collier, 2016) for entity detection, and RNNs (Zhang and Wang, 2015), CNNs (Zeng et al., 2014), Transformer (Verga et al., 2018; Wang et al., 2019) for relation classification.

As discussed in §1, joint entity and relation extraction is typically formulated as a sequence tagging based task. Traditionally, table-filling methods have been widely explored (Miwa and Sasaki, 2014; Gupta et al., 2016), where token labels and relation labels fill the diagonal and off-diagonal of the table respectively. Recently, many works concentrate on leveraging deep neural networks to tackle this task, e.g., stacked bidirectional LSTM (Miwa and Bansal, 2016; Zheng et al., 2017), combination of bidirectional LSTM and CNN (Zhou et al., 2017), and combination of bidirectional LSTM and attention mechanism (Chi et al., 2019; Nguyen and Verspoor, 2019). In addition, a novel machine reading comprehension based approach (Li et al., 2019) is proposed, which formulates the task as a **Question** & **Answer** task but still in sequence tagging based mode.

Recently, span-based joint extraction methods have been investigated to tackle problems existing in

sequence tagging based methods, e.g., inability to detect overlapping entities. Specially, Dixit and Al-Onaizan (2019) realize this method by obtaining span semantic representations through a BiLSTM over concatenated ELMo, word and character embeddings, then share them in both span and relation classifications. Luan et al. (2018) obtain span semantic representations generally the same as Lee et al. (2017), but reinforce them by introducing coreference task. Follow Luan et al. (2018), Luan et al. (2019) propose DyGIE, which can capture span interactions through a span graph constructed dynamically. Wadden et al. (2019) deliver further performance increases on DyGIE by replacing the BiLSTM with BERT and introduce DyGIE++. More recently, Eberts and Ulges (2019) propose SpERT, a simple but effective span-based model that takes BERT as backbone and use two FFNNs to classify span and relation respectively. Unlike previous works, SpERT dramatically reduces model training complexity by adopting negative sampling.

Our work follows SpERT, but differs in span-specific and contextual semantic representations. Specifically, our model obtains these semantic representations with attention mechanism. By calculating the matching degree between target sequence semantic representations and source sequence semantic representations, attention mechanism obtains attention scores on the source sequence, which are weight scores in essence. And the more important the information, the higher weight score it holds. Classified by implementation manners of score function, attention mechanism has multiple variants, e.g., Content-Base attention (Graves et al., 2014), Additive attention (Bahdanau et al., 2015), General attention (Luong et al., 2015), Dot-Product attention (Luong et al., 2015), Multi-Head attention (Vaswani et al., 2017).

3 Approach

In the rest, we abbreviate **"semantic representation"** as **"representation"**. Figure 2 gives an overview of our model, which uses BERT as encoder following SpERT: we map word embeddings into BERT embeddings using pre-trained Transformer blocks (Vaswani et al., 2017). Based on the representations, we calculate span representations and perform span classification & filtration (§3.1); Then, we organize relation tuples, calculate relation representations and perform relation classification & filtration (§3.2); Third, we investigate effects of multiple attention variants in generating contextual representations (§3.3); At last, we introduce model training settings (§3.4).

Define a sentence and a span from the sentence to help introduce the rest, where t denotes tokens and subscripts (e.g., 1, 2, 3...) denote token indexes, as:

$$\mathbf{sentence} : \mathcal{S} = (t_1, t_2, t_3, \ldots, t_n)$$

$$\mathbf{span} : \mathbf{s} = (t_i, t_{i+1}, t_{i+2}, ..., t_{i+j})$$

3.1 Span Classification and Filtration

Add **NoneEntity** type to the pre-defined entity types (denoted as η). Spans will be classified into NoneEntity as long as they do not hold any pre-defined entity types.

As Figure 2 shows, span representation for classification composes of four parts, namely **a)** concatenation of span head and tail representations, **b)**span-specific representation, **c)** sentence-level contextual representation, and **d)** span width embedding. We use X_i to denote the BERT embedding of token t_i, and the BERT embedding sequences of $\mathcal{S}$ and $\mathbf{s}$ are defined as follows, where X_0 denotes the BERT embedding of **[CLS]**:

$$\mathcal{B}_{\mathcal{S}} = [X_0, X_1, X_2, X_3, ..., X_n]$$

$$\mathcal{B}_{\mathbf{s}} = [X_i, X_{i+1}, X_{i+2}, ..., X_{i+j}]$$

Concatenation of span head and tail representations. If a span composes of more than one token, then concatenate the BERT embeddings of span head and tail tokens. Else, duplicate the BERT embedding of the single token and concatenate them. The concatenation result for span $\mathbf{s}$ is as:

$$\mathcal{H}_{\mathbf{s}} = [X_i; X_{i+j}] \tag{1}$$

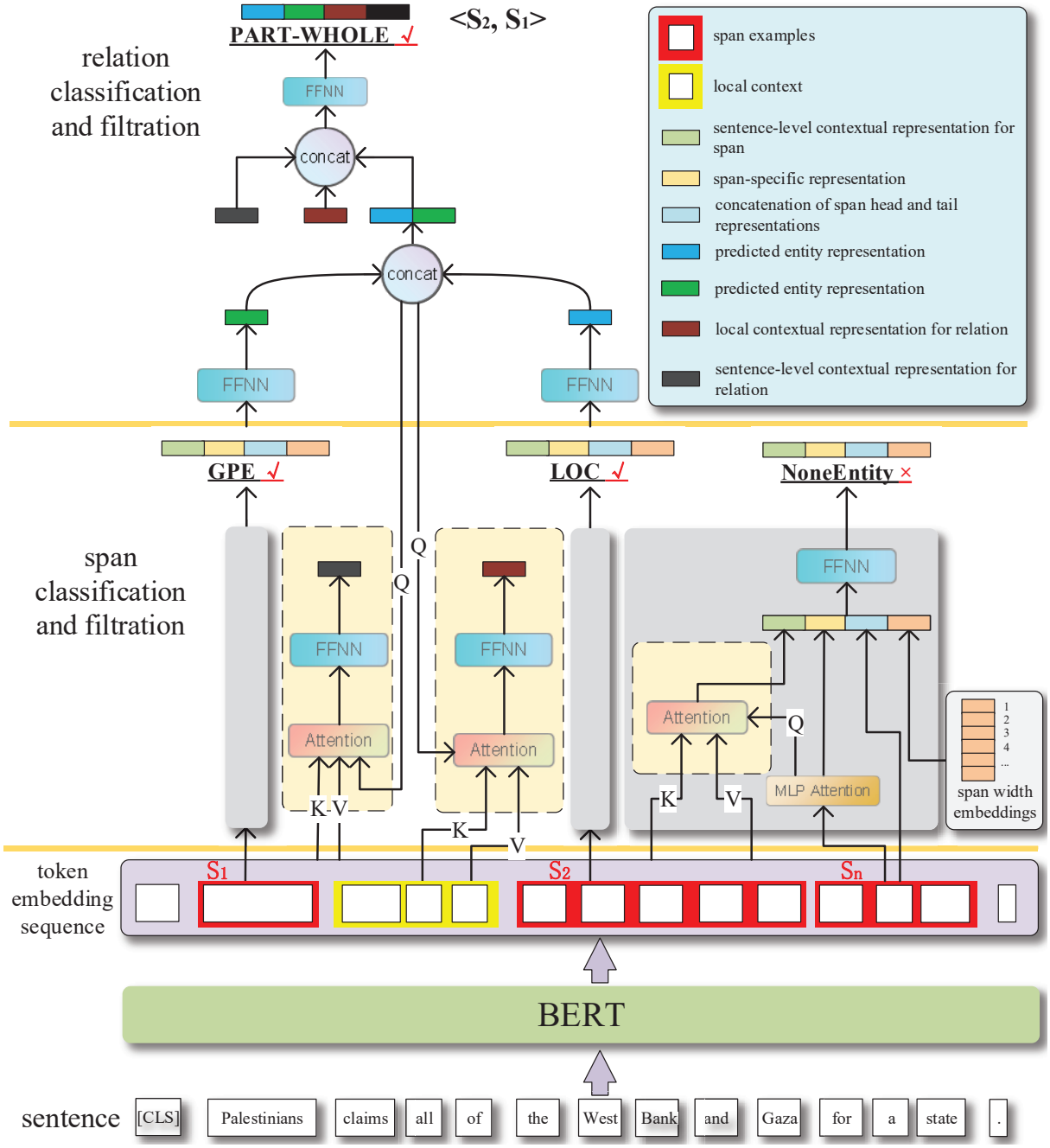

Figure 2: Our joint extraction model with attention-based span-specific and contextual representations. **1**) MLP attention is utilized to obtain span-specific representation; **2**) Sentence-level contextual representation for span is obtained by attention calculation between span-specific representation and sentence token embedding sequence; **3**) Relation local and sentence-level contextual representations are calculated by relation tuple representation attending to corresponding token embedding sequences.

Span-specific representation. Here we use MLP attention (Dixit and Al-Onaizan, 2019) to calculate span-specific representation. Take span $\mathbf{s}$ as an example.

$$\mathcal{V}_{\mathbf{k}} = \mathbf{MLP}_k(X_{\mathbf{k}}) \qquad s.t. \quad \mathbf{k} \in [i, i+j] \tag{2a}$$

$$\alpha_{\mathbf{k}} = \frac{\mathbf{exp}(\mathcal{V}_{\mathbf{k}})}{\sum_{m=i}^{i+j} \mathbf{exp}(\mathcal{V}_{\mathbf{m}})} \tag{2b}$$

$$\mathcal{F}_{\mathbf{s}} = \sum_{m=i}^{i+j} \alpha_{\mathbf{m}} X_{\mathbf{m}} \tag{2c}$$

Where $\mathcal{V}_{\mathbf{k}}$ is a scalar; $\alpha_{\mathbf{k}}$ is the attention weight of $\mathbf{X_k}$, computed by Softmax function; $\mathcal{F}_{\mathbf{s}}$ is the span-specific representation by matrix calculation on attention weights and $\mathcal{B}_{\mathbf{s}}$. By this way, we can evaluate the significance of each span token, and the more important a token, the larger attention weight it holds.

Sentence-level contextual representation. Take $\mathcal{F}_{\mathbf{s}}$ as query, $\mathcal{B}_{\mathcal{S}}$ as key and value respectively, sentence-level contextual representation for span $\mathbf{s}$ is calculated as:

$$\mathcal{T}_{\mathbf{s}} = \mathbf{Attention}(\mathcal{F}_{\mathbf{s}}, \mathcal{B}_{\mathcal{S}}, \mathcal{B}_{\mathcal{S}}) \tag{3}$$

Information beneficial for span classification will be assigned a heavy weight, and the contextual representation will be taken to constitute span representation.

Span width embedding. Span width embedding allow the model to incorporate a prior over the span width. Fixed-size embedding for each span width 1,2,... (Lee et al., 2017) is learned during model training. Thus, we can look up a width embedding $\mathcal{W}_{j+1}$ from the embedding matrix for $\mathbf{s}$.

Span classification. The final span representation for classification is as:

$$\mathcal{R}_{\mathbf{s}} = [\mathcal{T}_{\mathbf{s}};\ \mathcal{F}_{\mathbf{s}};\ \mathcal{H}_{\mathbf{s}};\ \mathcal{W}_{j+1}] \tag{4}$$

$\mathcal{R}_{\mathbf{s}}$ first passes through a multi-layers FFNN and then is fed into a Softmax classifier, which yields a posterior for $\mathbf{s}$ on η (including NoneEntity), as:

$$y_{\mathbf{s}} = \mathbf{Softmax}(\mathbf{FFNN}(\mathcal{R}_{\mathbf{s}})) \tag{5}$$

Span filtration. By searching the highest-scored class, the $y_{\mathbf{s}}$ estimates which entity type does $\mathbf{s}$ holds. We just keep spans that are not classified into NoneEntity, and form a predicted entity set $\mathcal{E}$. Then we perform relation classification on relation tuples derived from $\{\mathcal{E} \otimes \mathcal{E}\}$ to reduce searching space, where $\otimes$ denotes Cartesian Product.

3.2 Relation Classification and Filtration

Add **NoneRelation** type to the pre-defined relation types (denoted as γ). Let s_1, s_2 be two spans, relation tuples taken for relation classification are defined as:

$$< s_1, s_2 > \in \{\mathcal{E} \otimes \mathcal{E}\} \qquad s.t. \quad s_1 \neq s_2$$

As Figure 2 shows, relation representation for classification composes of three parts, namely **a**) concatenation of relation tuple representations, **b**) local contextual representation, **c**) sentence-level contextual representation.

Concatenation of relation tuple representations. Before concatenating $\mathcal{R}_{s_1}$ and $\mathcal{R}_{s_2}$, we first apply a multi-layers FFNN to them to reduce their dimensions. The concatenation result is as:

$$\mathcal{H}_{\mathbf{r}} = [\mathbf{FFNN}(\mathcal{R}_{s_1}); \mathbf{FFNN}(\mathcal{R}_{s_2})] \tag{6}$$

Local contextual representation. Let $\mathcal{B}_{\mathbf{c}}$ denotes the BERT embedding sequence of local context between s_1 and s_2, as:

$$\mathcal{B}_{\mathbf{c}} = (X_m, X_{m+1}, X_{m+2}, \ldots, X_{m+n})$$

The attention-based local contextual representation is calculate by taking $\mathcal{H}_\mathbf{r}$ as query, $\mathcal{B}_\mathbf{c}$ as key and value respectively, as:

$$\mathcal{F}_\mathbf{r} = \mathbf{Attention}(\mathcal{H}_\mathbf{r}, \mathcal{B}_\mathbf{c}, \mathcal{B}_\mathbf{c}) \tag{7}$$

Sentence-level contextual representation. The sentence-level contextual representation is calculated by taking $\mathcal{H}_\mathbf{r}$ as query, $\mathcal{B}_\mathcal{S}$ as key and value respectively, as:

$$\mathcal{T}_\mathbf{r} = \mathbf{Attention}(\mathcal{H}_\mathbf{r}, \mathcal{B}_\mathcal{S}, \mathcal{B}_\mathcal{S}) \tag{8}$$

Relation classification. Before $\mathcal{F}_\mathbf{r}$ and $\mathcal{T}_\mathbf{r}$ are taken to constitute relation representation, we first apply two different multi-layers FFNNs to them to reduce their dimensions, aiming to keep them in a proper proportion in relation representation. The final relation representation for classification is as:

$$\mathcal{R}_\mathbf{r} = [\mathcal{H}_\mathbf{r}; \mathbf{FFNN}_\mathcal{F}(\mathcal{F}_\mathbf{r}); \mathbf{FFNN}_\mathcal{T}(\mathcal{T}_\mathbf{r})] \tag{9}$$

Akin to span classification, $\mathcal{R}_\mathbf{r}$ first passes through a multi-layers FFNN and then is fed into a Softmax classifier, which yields a posterior for $<s_1, s_2>$ on γ (including NoneRelation), as:

$$y_\mathbf{r} = \mathbf{Softmax}(\mathbf{FFNN}(\mathcal{R}_\mathbf{r})) \tag{10}$$

Relation filtration. By searching the highest-scored class, the $y_\mathbf{r}$ estimates which relation type does $<s_1, s_2>$ holds. Only relation tuples that are not classified into NoneRelation are kept and compose predicted relation triples with predicted types.

3.3 Attention Variants

In this paper, we investigate effects of Multi-Head attention, Additive attention, Dot-Product attention and General attention in generating contextual representations, of which score functions are shown below.

$$Multi-Head\ attention:\ \mathbf{score} = \frac{\mathcal{Q} \odot \mathcal{K}}{\sqrt{d_\mathcal{K}}}$$

$$Additive\ attention:\ \mathbf{score} = \mathcal{W}_1 \cdot \mathcal{Q} + \mathcal{W}_2 \cdot \mathcal{K}$$

$$Dot-Product\ attention:\ \mathbf{score} = \mathcal{W} \cdot (\mathcal{Q} \odot \mathcal{K})$$

$$General\ attention:\ \mathbf{score} = \mathcal{Q} \cdot \mathcal{W} \cdot \mathcal{K}$$

Where $\mathcal{Q}, \mathcal{K}$ denote query and key respectively; $\mathcal{W}$ denotes parameter matrix; $d_\mathcal{K}$ denotes dimension of $\mathcal{K}$; $\cdot\ and\ \odot$ denote matrix broadcast and element-wise multiplication respectively. For Multi-Head attention and Dot-Product attention, we first apply different multi-layers FFNNs to $\mathcal{Q}$ and $\mathcal{K}$, aiming to convert them to the same dimension.

3.4 Model Training

Parameter matrices of FFNNs and attentions are learned, and BERT is fine-tuned during model training. The joint loss function of our model is defined as:

$$\mathcal{L} = 0.4\mathcal{L}^\mathbf{s} + 0.6\mathcal{L}^\mathbf{r} \tag{11}$$

Where $\mathcal{L}^\mathbf{s}$ denotes the cross-entropy loss of span classification and $\mathcal{L}^\mathbf{r}$ denotes the binary cross-entropy loss of relation classification. Due to the fact that performance of relation classification is generally worse than span one, we apply a larger weight score to $\mathcal{L}^\mathbf{r}$, aiming to let the model focus more on relation classification.

Negative sampling (Eberts and Ulges, 2019) are adopted during model training to improve model performance and robustness. Unlike previous works, we adopt a dynamic sampling strategy, where the negative examples for both entity and relation are thirtyfold of the ground truth ones in each sentence. By this strategy, our model keeps a much more balanced data distribution on training data.

4 Experiments

4.1 Datasets

We test our model on ACE2005, CoNLL2004 and ADE, which are refered as ACE05, CoNLL04 and ADE respectively in the rest.

- **ACE05** (Doddington et al., 2004) English dataset composes of news articles in multi-domain, e.g., broadcast, newswire, weblog etc.. Seven coarse-grained entity types and six coarse-grained relation types are pre-defined. We follow the training/dev/test split in (Li and Ji, 2014; Li et al., 2019). It has 351 documents for training, 80 for development and 80 for test, of which **437 contain overlapping entities**.

- **CoNLL04** (Roth and Yih, 2004) composes of news articles from outlets such as WSJ and AP, We follow the training/dev/test split in (Adel and Schütze, 2017; Bekoulis et al., 2018), which consists 910 articles for training, 243 for development and 288 for test.

- **ADE** (Gurulingappa et al., 2012) aims to extract drug-related adverse effects from medical text, including two pre-defined entity types (namely Adverse-Effect and Drug) and a single relation type i.e., Adverse-Effect. It consists of 4272 sentences, of which **1695 contain overlapping entities**. We conduct 10-fold cross validation following (Bekoulis et al., 2018; Eberts and Ulges, 2019).

For ACE05, following (Li et al., 2019; Luan et al., 2019), an entity is considered correct if we can identify its head region and type correctly. A relation is considered correct if we can identify its argument entities and type correctly. For CoNLL04 and ADE, following (Li et al., 2019; Eberts and Ulges, 2019), we treat an entity as correct when its type and entity region match ground truth, and treat a relation as correct when its type and argument entities match ground truth.

4.2 Experimental Settings

We build our model upon English cased version of $\mathbf{BERT_{BASE}}$. We set the negative sampling rate to 30, batch size for model training to 8, dropout to 0.2 and width embedding dimension to 50. For Multi-Head attention, the head number is set to 8. $\mathbf{FFNN}_{\mathcal{F}}$ and $\mathbf{FFNN}_{\mathcal{T}}$ contain three fully connected layers; and all the other FFNNs contain two layers. We set different epochs in case of different datasets. For all datasets, the span width threshold is initialized to 10. In our model, we adopt weight loss setting, as shown in equ.(11), and follow Eberts and Ulges (2019) for other hyperparameter settings.

4.3 Baseline Models

We compare our model with the following models.

- **DyGIE** (Luan et al., 2019) is the current span-based state-of-the-art model on ACE05. It reinforces span and relation representations by introducing coreference task.

- **Multi-turn QA** (Li et al., 2019) is the current sequence tagging based state-of-the-art model on ACE05 and CoNLL04. It formulates joint entity and relation extraction as a multi-turn question and answer task, but still in sequence tagging based mode.

- **SpERT** (Eberts and Ulges, 2019) is the current span-based state-of-the-art model on ADE and CoNLL04. Our work follows this model but adopts attention-based span-specific and contextual representations.

- **Relation-Metric** (Tran and Kavuluru, 2019) is a sequence tagging based model in multi-task learning scheme. It reports performances on ADE and CoNLL04, and achieves state-of-the-art on ADE.

Dataset	Method	Entity			Relation		
		P	R	F1	P	R	F1
ACE05	Multi-turn QA †	84.7	84.9	84.8	64.8	56.2	60.2
	DyGIE ‡	-	-	88.4	-	-	63.2
	$\mathbf{SPAN}_{Multi-Head}$	**89.32**	**89.86**	**89.59**	**71.22**	**60.19**	**65.24**
CoNLL04	Multi-turn QA †	89.0	86.6	87.8	69.2	68.2	68.9
	SpERT ‡	88.25	89.64	88.94	73.04	70.00	71.47
	$\mathbf{SPAN}_{Multi-Head}$	**90.11**	**90.36**	**90.23**	**76.96**	**71.88**	**74.33**
ADE	Relation-Metric †	86.16	88.08	87.11	77.36	77.25	77.29
	SpERT ‡	88.99	89.59	89.28	77.77	79.96	78.84
	$\mathbf{SPAN}_{Multi-Head}$	**89.88**	**91.32**	**90.59**	**79.56**	**81.93**	**80.73**

Table 1: Results of different models on ACE05, CoNLL04 and ADE test sets. $\mathbf{SPAN}_{Multi-Head}$ outperforms previous SOTA models by +1.19, +1.29, +1.31 in entity recognition, and +2.04, +2.86, +1.89 in relation extraction. (previous sequence tagging based SOTA †; previous span-based SOTA ‡)

Method	ACE05		CoNLL04		ADE	
	Entity (F1)	Relation (F1)	Entity (F1)	Relation (F1)	Entity (F1)	Relation (F1)
$\mathbf{SPAN}_{Multi-Head}$	**89.59**	**65.24**	**90.23**	**74.33**	**90.59**	**80.73**
$\mathbf{SPAN}_{Dot-Product}$	87.94	62.88	88.23	70.89	88.15	77.31
$\mathbf{SPAN}_{General}$	88.66	63.56	88.96	73.48	89.93	80.14
$\mathbf{SPAN}_{Additive}$	89.07	64.53	89.17	71.36	89.68	79.75

Table 2: Performance comparations of attention variants. Multi-Head attention firmly outperforms the others, with an absolute F1 increase up to 2.44(ADE) in entity recognition and 3.44(CoNLL04) in relation extraction.

4.4 Main Results

We compare our model with both sequence tagging based methods and span-based methods on the three benchmark datasets, and show the results in Table 1. We denote our method as $\mathbf{SPAN}_{Multi-Head}$, which means we use Multi-Head attention to calculate contextual representations. For ACE05 and CoNLL04, we report performance under **micro-average metrics**, and apply **macro-average metrics** to ADE performance, which follow prior works. For ACE05 and ADE, all the reported performances take overlapping entities into consideration.

$\mathbf{SPAN}_{Multi-Head}$ consistently outperforms both sequence tagging based and span-based SOTA methods on the three benchmark datasets. Compared to SpERT, $\mathbf{SPAN}_{Multi-Head}$ delivers performance increases in entity recognition by 1.29(CoNLL04) and 1.31(ADE), while better ones in relation extraction, by 2.86(CoNLL04) and 1.89(ADE). We owe these performance increases to efficient span-specific representation and contextual representations. Moreover, $\mathbf{SPAN}_{Multi-Head}$ delivers solid performance increases compared to DyGIE by 1.19 and 2.04 in entity recognition and relation extraction on ACE05. However, it's worth noting that DyGIE adopts a multi-task learning scheme, reinforcing span representations by introducing coreference task, which is absent in our method.

Besides Multi-Head attention, we investigate Dot-Product attention, General attention and Additive attention in our method. Table 2 shows performances of these attention variants on the three benchmark datasets. $\mathbf{SPAN}_{Multi-Head}$ consisiently outperforms the other three methods. One possible reason is that in Multi-Head attention, the eight attention heads attend to different contextual information and learn features from different representation spaces. Thus Multi-Head attention based contextual representations can better compensate span and relation representations.

Method	Entity (F1)	Relation (F1)
$\mathbf{SPAN}_{Multi-Head}$	88.10	62.13
-SpanSpecific	86.78	60.21
-SentenceLevel	87.57	61.12
base	85.80	59.00

Table 3: Ablations on ACE05 dev set with different span-specific and span sentence-level contextual representation settings.

Method	Entity (F1)	Relation (F1)
$\mathbf{SPAN}_{Multi-Head}$	88.10	62.13
-local	87.96	60.56
-SentenceLevel	88.21	61.77
base	87.91	59.66

Table 4: Ablations on ACE05 dev set with different relation local and sentence-level contextual representation settings.

5 Ablation Study

Based on $\mathbf{SPAN}_{Multi-Head}$, we conduct ablations on the ACE2005 dev set to analyze effects of different model components.

5.1 Span-specific and Sentence-level Contextual Representations for Span

Table 3 shows effects of span-specific representation and sentence-level contextual representation for span in our model, where -SpanSpecific denotes ablating the span-specific representation by replacing $[\mathcal{F}_\mathbf{s}; \mathcal{H}_\mathbf{s}]$ in equ.(4) with the max pooling of $\mathcal{B}_\mathbf{s}$; -SentenceLevel denotes ablating the sentence-level contextual representation by replacing $\mathcal{T}_\mathbf{s}$ in equ.(4) with the BERT embedding of [CLS] generated on $\mathcal{B}_\mathcal{S}$; base is the model by performing above two ablations, which is the default span representation settings in SpERT. For ACE05, we observe that both span-specific representation and sentence-level contextual representation are helpful for both entity recognition and relation extraction. This is due to that span representations are shared in the two subtasks.

5.2 Local and Sentence-level Contextual Representations for Relation

Table 4 shows effects of local and sentence-level contextual representations for relation in our model, where -local denotes ablating local representation by replacing $\mathbf{FFNN}_\mathcal{F}(\mathcal{F}_\mathbf{r})$ in equ.(9) with the max pooling of $\mathcal{B}_\mathbf{c}$; -SentenceLevel denotes ablating sentence-level contextual representation by removing $\mathbf{FFNN}_\mathcal{T}(\mathcal{T}_\mathbf{r})$ from equ.(9); base is the model by performing above two ablations, which is the default relation representation settings in SpERT. For ACE05, we observe that both local and sentence-level contextual representations apparently benefit relation extraction, while have negligible influence on entity recognition. A convincing explanation is that these representations directly constitute relation representation, while affect span representation only by backpropagation.

It is worth noting that local contextual representation has a greater impact on relation extraction compared to sentence-level one. One reason for this is that information determining relation type mainly exists in relation tuples and the local context. Another reason is that as compensation information, sentence-level contextual representation occupies a relative small proportion in relation representation, aiming to avoid introducing noise into relation representation.

6 Conclusion

We introduce attention-based semantic representation generating methods in span-based joint entity and relation extraction method. We apply MLP attention to capture span-specific features aiming to obtain semantic rich span representation, and calculate task-specific contextual representations with attention architecture to further reinforce span and relation representations. Our approach firmly outperforms both the sequence tagging based and span-based SOTA methods on three benchmark datasets, creating new state-of-the-art results. As future work, we would like to consider further improving relation classification performance by reducing span classification errors. We also plan to explore more advanced methods for encoding efficient span and relation representations.

Acknowledgements

The work is supported by the National Key Research and Development Program of China (2018YFB1004502) and the National Natural Science Foundation of China (61532001).

References

Heike Adel and Hinrich Schütze. 2017. Global normalization of convolutional neural networks for joint entity and relation classification. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1723–1729. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Syst. Appl.*, 114:34–45.

Renjun Chi, Bin Wu, Linmei Hu, and Yunlei Zhang. 2019. Enhancing joint entity and relation extraction with language modeling and hierarchical attention. In Jie Shao, Man Lung Yiu, Masashi Toyoda, Dongxiang Zhang, Wei Wang, and Bin Cui, editors, *Web and Big Data - Third International Joint Conference, APWeb-WAIM 2019, Chengdu, China, August 1-3, 2019, Proceedings, Part I*, volume 11641 of *Lecture Notes in Computer Science*, pages 314–328. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Kalpit Dixit and Yaser Al-Onaizan. 2019. Span-level model for relation extraction. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5308–5314. Association for Computational Linguistics.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. Weischedel. 2004. The automatic content extraction (ACE) program - tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*. European Language Resources Association.

Markus Eberts and Adrian Ulges. 2019. Span-based joint entity and relation extraction with transformer pre-training. *arXiv preprint arXiv:1909.07755*, abs/1909.07755.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2537–2547. ACL.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. 2012. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *J. Biomed. Informatics*, 45(5):885–892.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*, abs/1508.01991.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 188–197. Association for Computational Linguistics.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 402–412. The Association for Computer Linguistics.

Xiaoya Li, Fan Yin, Zijun Sun, Xiayu Li, Arianna Yuan, Duo Chai, Mingxin Zhou, and Jiwei Li. 2019. Entity-relation extraction as multi-turn question answering. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1340–1350. Association for Computational Linguistics.

Nut Limsopatham and Nigel Collier. 2016. Bidirectional LSTM for named entity recognition in twitter messages. In Bo Han, Alan Ritter, Leon Derczynski, Wei Xu, and Tim Baldwin, editors, *Proceedings of the 2nd Workshop on Noisy User-generated Text, NUT@COLING 2016, Osaka, Japan, December 11, 2016*, pages 145–152. The COLING 2016 Organizing Committee.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3219–3232. Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3036–3046. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Xuezhe Ma and Eduard H. Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1858–1869. ACL.

Dat Quoc Nguyen and Karin Verspoor. 2019. End-to-end neural relation extraction using deep biaffine attention. In Leif Azzopardi, Benno Stein, Norbert Fuhr, Philipp Mayr, Claudia Hauff, and Djoerd Hiemstra, editors, *Advances in Information Retrieval - 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14-18, 2019, Proceedings, Part I*, volume 11437 of *Lecture Notes in Computer Science*, pages 729–738. Springer.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *Proceedings of the Eighth Conference on Computational Natural Language Learning, CoNLL 2004, Held in cooperation with HLT-NAACL 2004, Boston, Massachusetts, USA, May 6-7, 2004*, pages 1–8. ACL.

Tung Tran and Ramakanth Kavuluru. 2019. Neural metric learning for fast end-to-end relation extraction. *arXiv preprint arXiv:1905.07458*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 872–884. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5783–5788. Association for Computational Linguistics.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019. Extracting multiple-relations in one-pass with pre-trained transformers. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1371–1377. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In Jan Hajic and Junichi Tsujii, editors, *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 2335–2344. ACL.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*, abs/1508.01006.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1227–1236. Association for Computational Linguistics.

Peng Zhou, Suncong Zheng, Jiaming Xu, Zhenyu Qi, Hongyun Bao, and Bo Xu. 2017. Joint extraction of multiple relations and entities by using a hybrid neural network. In Maosong Sun, Xiaojie Wang, Baobao Chang, and Deyi Xiong, editors, *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data - 16th China National Conference, CCL 2017, - and - 5th International Symposium, NLP-NABD 2017, Nanjing, China, October 13-15, 2017, Proceedings*, volume 10565 of *Lecture Notes in Computer Science*, pages 135–146. Springer.