Pre-training Entity Relation Encoder with Intra-span and Inter-span Information

Yijun Wang[1, 2], **Changzhi Sun**[3], **Yuanbin Wu**[3], **Junchi Yan**[1, 2], **Peng Gao**[4], **and Guotong Xie**[4]

[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[3]School of Computer Science and Technology, East China Normal University
[4]Ping An Technology (Shenzhen) Co. Ltd. Shenzhen, China
{yijunwang.cs, czsun.cs}@gmail.com ybwu@cs.ecnu.edu.cn
yanjunchi@sjtu.edu.cn {gaopeng712, xieguotongg}@pingan.com.cn

Abstract

In this paper, we integrate span-related information into pre-trained encoder for entity relation extraction task. Instead of using general-purpose sentence encoder (e.g., existing universal pre-trained models), we introduce a span encoder and a span pair encoder to the pre-training network, which makes it easier to import intra-span and inter-span information into the pre-trained model. To learn the encoders, we devise three customized pre-training objectives from different perspectives, which target on tokens, spans, and span pairs. In particular, a span encoder is trained to recover a random shuffling of tokens in a span, and a span pair encoder is trained to predict positive pairs that are from the same sentences and negative pairs that are from different sentences using contrastive loss. Experimental results show that the proposed pre-training method outperforms distantly supervised pre-training, and achieves promising performance on two entity relation extraction benchmark datasets (ACE05, SciERC).

1 Introduction

Extraction of entities and relations from free texts is an important task in NLP. Its goal is to recognize text spans with specific types (*entities*) and semantic relations among those entities (*relations*). Current state-of-the-art systems usually employ the supervised joint learning algorithm (Miwa and Bansal, 2016; Sun et al., 2018, 2019a), which can alleviate error propagation caused by the pipeline method. In this paper, we focus on joint entity relation extraction.

Recently, pre-trained models (Devlin et al., 2018; Dong et al., 2019) have substantially advanced a variety of NLP tasks, including entity relation extraction (Li et al., 2019; Wadden et al., 2019). (Wadden et al., 2019) adopt BERT as a sentence encoder and build a multi-task framework for information extraction. However, universal pre-trained models are usually trained without explicitly handling text spans and relation among text span pairs. For example, the objectives of BERT are masked language model and next sentence prediction, which are defined at the token level and sentence level, respectively. It rarely considers incorporating span-related knowledge, which can provide rich information for better extracting entities and relations (Table 1).

	BERT	SpanBERT	ERNIE	Ours
Token Level	✓	✓	✓	✓
Span Level		✓	$\underline{\checkmark}$	✓
Span Pair Level				✓
Sentence Level	✓		✓	

Table 1: Comparison between pre-training objectives. $\underline{\checkmark}$ means that additional annotations (entities) are used.

The traditional way to introduce more entity relation related information is through distant supervision, which aligns triples in knowledge bases and free texts. However, the distantly supervised dataset contains lots of noise samples, which may have a negative impact on other datasets as prior works (Sun and Wu, 2019). Besides, the distantly supervised dataset's annotated labels are usually inconsistent with that of the target dataset. As expected, in the preliminary experiment, we observe that the performance of the model directly pre-trained with annotated data provided by distantly supervised dataset (such as NYT) is not improved or even gets worse when it is fine-tuned on other entity relation dataset (such as ACE05). In addition, there are several existing works for incorporating entity information into pre-training objectives (Zhang et al., 2019; Sun et al., 2019b). However, these methods rely on entity annotations, which brings additional cost.

In this work, we focus on the unsupervised pre-training objectives. We present a novel pre-training network architecture customized for entity relation extraction. In addition to the default sentence encoder in existing pre-trained models (e.g., the Transformer encoder of BERT), we also pre-train *a span encoder* and *a span pair encoder*. To learn the two encoders, we propose three pre-training objectives corresponding to three levels: token boundary objective (token level), span permutation objective (span level), and contrastive span pair objective (span pair level). Token boundary objective can help to enhance the representation of the first subtoken of each token. Span encoder is trained by recovering the correct order of span tokens from its random shufflings. Span pair encoder is trained by the contrastive loss. Specifically, the predictions are made discriminatively with a sampled-softmax that contrasts positive pairs against negative pairs. Positive pairs are from the same sentences, while negative pairs are from different sentences. These three objectives share parameters and will be trained jointly.

A closely related work to span level pre-training objective is SpanBERT (Joshi et al., 2020), which adopts the span boundary objective to incorporate the span information. Different from (Joshi et al., 2020), we introduce not only a new objective at the span level but also a new objective at the span pair level (Table 1). Inspired by the recently proposed InfoWord (Kong et al., 2019), we use the contrastive loss to learn a better span pair representation. To utilize a large set of negative pairs without requiring large training batches, we extend the MoCo (He et al., 2019) framework to the proposed span pair objective. In summary, our main contributions are in the following [1]:

• We introduce a span encoder and a span pair encoder to incorporate intra-span and inter-span information in the pre-training network architecture, which is ignored in universal pre-trained models.

• We devise three novel objectives, token boundary objective, span permutation objective, and contrastive span pair objective, to learn the better encoders.

• The experimental results demonstrate that the proposed method not only exceeds the strong BERT baseline in entity relation extraction task but also achieves significant improvements (3% absolute) on the ACE05 dataset, and is comparable with the state-of-the-art on the SciERC dataset.

[1]Source code and pre-trained models are available at https://github.com/Receiling/PSPE.

2 Background of Contrastive Learning

InfoNCE Contrastive learning is a framework that builds representations by learning to encode what makes two things similar or dissimilar [2]. Recently, (He et al., 2019) regrad contrastive learning as a *dictionary look-up* task. An effective contrastive loss function, called InfoNCE (Oord et al., 2018), is as follows. Formally, for any data point X, to learn a query encoder f_q and a key encoder f_k (the two encoders can be different, partially shared, or identical, we adopt two identical encoders), InfoNCE is to minimize the following loss function [3]

$$-\mathop{\mathbb{E}}_{X,X^+}\left[f_q(X)\cdot f_k(X^+) - \log Z\right]$$
$$Z = \exp(f_q(X)\cdot f_k(X^+)) + \sum_{X^-}\exp(f_q(X)\cdot f_k(X^-))$$

where X is a query sample and $\{X^+, X^-\}$ are key samples. X^+ is a similar key to query X and X^- is presumably dissimilar to X. Thus, X, X^+, X^- are referred to as *anchor*, *positive*, *negative* respectively in the parlance of contrastive learning.

Momentum Contrast (MoCo) Contrastive learning tends to work better with more negative examples, since presumably negative examples can decide the quality of the underlying representations learned. In the usual formulation of contrastive learning, the gradients flow back through both the query encoder and the key encoder, which means that the number of negative samples is restricted to the mini-batch size. Thus, the MoCo framework (He et al., 2019) is devised to process a large set of negative samples without requiring large training batches. Specifically, Instead of updating the key encoder with gradients back-propagation, MoCo periodically updates the key encoder using a momentum update:

$$\theta_k = m\theta_k + (1-m)\theta_q$$

Here, θ_k denotes the parameters of the key encoder (also called momentum encoder), and θ_q denotes the parameters of the query encoder (Figure 1). $m \in [0, 1)$ is a momentum coefficient (e.g.,

[2]For more theoretical understanding, please refer to (Arora et al., 2019).

[3]Minimizing the InfoNCE loss maximizes a lower bound on the mutual information between $f_q(X)$ and $f_k(X^+)$ (Kong et al., 2019).

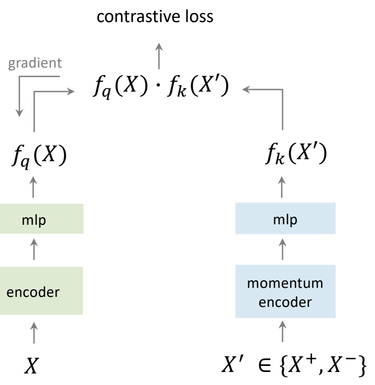


Figure 1: Framework of MoCo with an MLP projection.

$m = 0.999$, our default). MoCo only updates parameters θ_q with back-propagation, and decouples the size of key samples from the mini-batch size. Thus, MoCo can maintain a large queue of key samples, which contains one positive key sample and lots of negative key samples to one query. In addition, introducing an MLP (multi-layer perceptron) head projection between the representation and the contrastive loss substantially improves the quality of the learned representations (Chen et al., 2020a,b).

3 Approach

Given an input sentence $x = x_1, \ldots, x_{|x|}$ and a set of spans $\mathcal{S}$ (randomly sampling) in x, the target of our pre-training model is to obtain a contextualized vector representation for each span $s \in \mathcal{S}$, and a contextualized vector representation for each span pair (s_1, s_2). As shown in Figure 2, the pre-training task optimizes a shared Transformer (Vaswani et al., 2017) network, a span level CNN and attention parameters with respect to a token boundary objective, a span permutation objective, and a contrastive span pair objective. Different from universal pre-trained language models (Devlin et al., 2018; Peters et al., 2018), the proposed network incorporates rich intra-span and inter-span information [4]. Once our network is pre-trained, we can fine-tune it for entity relation extraction task.

3.1 Pre-training Network Architecture

This section presents the overall pre-training network architecture for sentence encoder, span encoder, and span pair encoder. The next section will describe the objectives for training the three components.

Sentence Encoder To obtain the contextual representations $\mathbf{h}_i$ for each token in the sentence x, we use multi-layer Transformer (Vaswani et al., 2017) as basic encoder like previous pre-training models, such as UniLM, BERT, and XLM. The output of the multi-layer Transformer is computed via:

$$\{\mathbf{h}_1, \ldots, \mathbf{h}_{|x|}\} = \texttt{Transformer}(\{\mathbf{x}_1, \ldots, \mathbf{x}_{|x|}\})$$

The word representation $\mathbf{x}_i$ of x_i follows that of BERT (Devlin et al., 2018), which sums the corresponding token, segment and position embeddings.

Span Encoder Given a span $s \in \mathcal{S}$ in the sentence x, to compute the corresponding contextual span representation $\mathbf{h}_s$, we employ a CNN (a single convolution layer with a max-pooling layer) followed by an MLP on vectors $\{\mathbf{h}_i | x_i \in s\}$, as shown in the right part of Figure 2.

Span Pair Encoder Given a span pair $p = (s_1, s_2)$ in the sentence x, the sentence x is split into five spans, namely, *left context (L)*, s_1, *middle context (M)*, s_2 and *right context (R)*. To obtain the corresponding contextual span pair representation $\mathbf{h}_{s_1,s_2}$, we first employ the span encoder to extract five feature vectors regarding the five spans. Let $\mathbf{h}_L, \mathbf{h}_{s_1}, \mathbf{h}_M, \mathbf{h}_{s_2}, \mathbf{h}_R$ be the corresponding representations computed by span encoder. To allow the model to focus on more informative spans, we then represent the span pair p as a weighted sum of its contextualized span representations with a position-aware attention mechanism as

$$\mathbf{h}_p = \sum_{j \in \{L, s_1, M, s_2, R\}} a_j \mathbf{h}_j,$$

where the attention score a_j is computed as

$$\begin{aligned} a_j &= \text{Softmax}(e_j), \\ e_j &= \mathbf{v}^T \tanh(\mathbf{W}_h \mathbf{h}_j + \mathbf{W}_{\text{cls}} \mathbf{h}_{\text{cls}} + \\ &\quad \mathbf{W}_{s_1} \mathbf{p}_j^{s_1} + \mathbf{W}_{s_2} \mathbf{p}_j^{s_2}), \end{aligned}$$

where $\mathbf{W}_*$ and $\mathbf{v}$ are parameters and $\mathbf{h}_{\text{cls}}$ is the output of the first token ([CLS]). Following (Zhang et al., 2017), $\mathbf{p}_j^{s_1}$ and $\mathbf{p}_j^{s_2}$ are the relative position embedding with repsect to s_1 and s_2.

3.2 Pre-training Objectives

Learning powerful representations of span and span pair is crucial for the entity relation extraction task,

[4] We extract entities and relations for each sentence, so we omit the next sentence prediction in BERT, which is a sentence level objective.

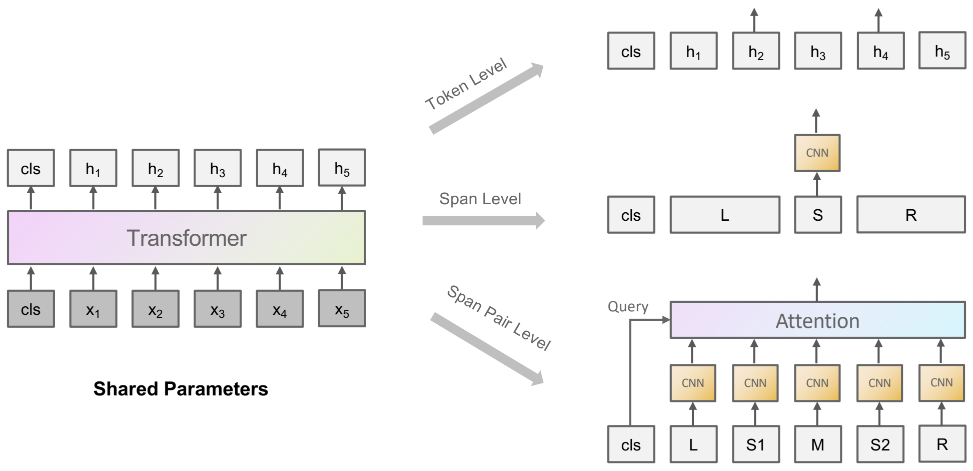


Figure 2: Overview of our pre-training. The Transformer parameters are shared across three objectives (i.e., token level, span level and span pair level).

which is not explicitly considered in the universal pre-trained models such as BERT. Here, we aim to design several tailored pre-training objectives that can guide the model to learn more powerful representations of spans and span pairs. The three tasks share parameters and are trained jointly (weighted sum of the objective functions).

Token Boundary Objective (TBO) In practice, masked language modeling (MLM) is usually applied at the sub-token level. Given the input sub-token sequence, a certain portion of sub-tokens are replaced by a special symbol `[M]`. The model is trained to recover the original sub-tokens from the corrupted version. In the downstream tasks, we simply take the first sub-token representation as the token representation. To enhance the first sub-token representation and maintain the token level information, we propose a variant MLM. Specifically, for each token, we mask the sub-tokens except the first sub-token, and then predict the masked sub-tokens with the first sub-token representation and corresponding position embedding. In experiments, for each sentence, we randomly select 15% of the sub-tokens to perform this objective.

Span Permutation Objective (SPO) Inspired by the recent SpanBERT (Joshi et al., 2020), we propose a different strategy to incorporate the intra-span information into our pre-training model. SpanBERT still focuses on enhancing single token representation, while we emphasize the contextual representation of the whole span. Instead of predicting each token of a masked span in SpanBERT, we shuffle the tokens in the span and then expect the model can recognize the disruption. Correctly, let $s = (x_{start}, x_{middle}, x_{end})$ be a span in the sentence x, where $start, end$ indicates its start and end position, and $middle$ indicates its middle positions (may contain multiple tokens). Let $\mathcal{P}$ be the set of all possible permutation of the three parts. Obviously, the number of all possible permutations is 3! ($|\mathcal{P}| = 6$). For each permutation $p \in \mathcal{P}$, we first assign it a unique permutation class $N_p (1 \leq N_p \leq |\mathcal{P}|)$, and then extract feature vectors regarding span s with the span encoder to predict the permutation class. The objective is to optimize the cross-entropy loss computed using the predicted permutation class and the gold permutation class. In the implementation, we sample n_p permutations (we always include the correct permutation).

Contrastive Span Pair Objective (CSPO) Previous pre-trained models only consider a single token or single span in the pre-training objective, and ignore the role of span pairs. For entity relation extraction task, it often involves predicting whether a relation exists on an entity pair. Thus, if we have a better-pre-trained span pair encoder, we may get a better entity relation extraction performance in the fine-tuning step. To this end, we propose a novel span pair level objective based on the contrastive learning framework. Inspired by InfoWord (Kong et al., 2019), it views spans and their matching contexts (i.e., contexts in the same sentences) as positive pairs, otherwise as negatives pairs. We extend this idea to the span pair level.

Formally, given a span pair $p = (s_1, s_2)$ in the sentence x, we consider the sentence with masked p (denoted by x_p^{context}) and the sentence with masked context of p (denoted by x_p^{target}) to be a positive pair (Figure 3). If both come from

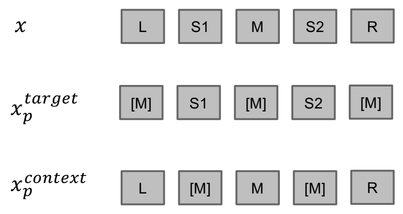


Figure 3: Examples of x_p^{target} and x_p^{context}.

two different sentences, it is a negative pair. In other words, $(x_p^{\text{context}}, x_p^{\text{target}})$ is a positive pair, and $(x_p^{\text{context}}, \hat{x}_{p'}^{\text{target}})$, $(\hat{x}_{p'}^{\text{context}}, x_p^{\text{target}})$ are negative pair, where p' is a span pair in another sentence $\hat{x}$. Next, we describe how to adapt the MoCo framework to achieve our span pair level objective.

To obtain the representations of x_p^{context} and x_p^{target}, we can apply the span pair encoder on the masked sentence. In expectation, the span pair encoder will learn a better representation from contrastive loss. We adopt the span pair encoder followed by an MLP as the two identical encoders f_q and f_k of the MoCo.

$$f_q(X) = f_k(X) = \texttt{MLP}(\texttt{SpanPairEncoder}(X))$$

We think of two situations for (X, X^+, X^-) as follows:

- We first consider x_p^{context} as the *anchor* data point X, i.e., $X = x_p^{\text{context}}$, then $X^+ = x_p^{\text{target}}$ and $X^- = \hat{x}_{p'}^{\text{target}}$;

- We can also consider x_p^{target} as the *anchor* data point X, i.e., $X = x_p^{\text{target}}$, then $X^+ = x_p^{\text{context}}$ and $X^- = \hat{x}_{p'}^{\text{context}}$.

where x and $\hat{x}$ are two different sentences. Given the input (X, X^+, X^-), the training objective is to minimize

$$- \sum_{(X,X^+,X^-)\in\mathcal{X}} \left\{ \mathop{\mathbb{E}}_{X,X^+} \left[f_q(X) \cdot f_k(X^+) - \log Z \right] \right\}$$
$$Z = \exp(f_q(X) \cdot f_k(X^+)) + \sum_{X^-} \exp(f_q(X) \cdot f_k(X^-))$$
$$\mathcal{X} = \{(x_p^{\text{context}}, x_p^{\text{target}}, \hat{x}_{p'}^{\text{target}}), (x_p^{\text{target}}, x_p^{\text{context}}, \hat{x}_{p'}^{\text{context}})\}$$

Appendix D provides the PyTorch-like pseudocode of MoCo for our proposed span pair task. For the current mini-batch, we encode the X and X^+, which form the positive sample pairs. The negative samples are from the queue (we maintain the two queues).

3.3 Pre-training Setup

Within one training batch, we optimize the weighted sum of three objectives. We use GELU as the activation function. The sentence encoder is initialized by BERT$_{\text{BASE}}$. We generate spans similar to (Joshi et al., 2020). For distantly supervised pre-training, we train our model for ten epochs with linear warm up rate over the first 20% steps and linear decay. For our unsupervised pre-training on the distantly supervised corpus (NYT), we train our model for ten epochs with linear warm up rate over the first 10% steps and linear decay. In order to achieve more training data, we sample sentences from English Wikipedia [5] and BooksCorpus, which has been processed similarly as (Devlin et al., 2018), and construct a dataset (4.8M sentences total) with 70M words. So the total iterations of pre-training are smaller than BERT$_{\text{BASE}}$. The vocabulary size is 28996. The maximum length of the input sequence is 128. We train our model for 40,000 steps with linear warm up rate over the first 18,000 steps and linear decay. Adam (Kingma and Ba, 2014) with $\beta_1 = 0.9, \beta_2 = 0.999$ is used for optimization. The learning rates of NYT and WikiBook are 5e-5 and 1e-4, respectively. The dropout rate is 0.1. The weight decay is 0.01. The batch size is 256 with gradient accumulation. It takes about 22 hours for 10, 000 steps using 1 Nvidia Tesla T4 16GB GPU.

3.4 Fine-tuning for Entity Relation Extraction

We define the entity relation extraction task as (Sun et al., 2019a) [6]. First, we perform entity span detection, which is tackled using the sequence labeling framework. We use the sentence encoder's output as the representation of words and feed it to a randomly initialized softmax classifier. We adopt cross-entropy loss for training the entity span detector. Then, for each detected entity span, we predict its entity type using a softmax classifier. The classifier takes its input from the pre-trained span encoder. Similarly, we predict its relation type for each detected entity span pair using a softmax classifier, which takes its input from the span pair encoder. We also adopt cross-entropy loss in these two tasks. Overall, three objectives are optimized simultaneously in the fine-tuning step.

[5] Wikipedia version: enwiki-20190301.

[6] The neural architecture for fine-tuning is provided in Appendix A.

Scheduled sampling was used for the entity model similar to (Miwa and Bansal, 2016). We adopt the discriminative fine-tuning strategy as (Howard and Ruder, 2018). We employ the early stop strategy and select models based on performances on the development set.

4 Experiments

We conduct experiments on two benchmark entity relation extraction datasets: ACE05 and SciERC. For the distantly supervised dataset, we choose the NYT dataset.

ACE05 The ACE05 dataset [7] annotates entity and relation types for a collection of documents. It is a standard corpus for entity relation extraction task. There are 7 entity types and 6 relation types in the corpus. We use the same data split of the ACE05 dataset (351 training, 80 validating, and 80 testing) as (Miwa and Bansal, 2016).

SciERC The SciERC dataset [8] provides entity, coreference and relation annotations for 500 scientific abstracts, which are taken from AI conference/workshop proceedings. We only use the annotations of entities and relations. The corpus contains 6 scientific entity types and 7 relation types. We use the same data split of SciERC dataset (350 training, 50 validating, and 100 testing) as (Luan et al., 2019).

NYT The NYT dataset[9] is a large-scale corpus which annotates 3 types of entities and 12 types of relations for New York Times news articles. The training set is automatically generated by distant supervision. (Jia et al., 2019) provides validation and testing data that are manually labeled. We do not use the testing data for pre-training. We choose the latest version NYT released by (Jia et al., 2019).

Evaluation. We evaluate F1 score as previous works (Miwa and Bansal, 2016; Sun et al., 2019a). Specifically, an output entity is correct if its type and boundary are correct, and an output relation is correct if its type and its two-argument entities are correct (i.e., exactly match). Some previous works (Luan et al., 2019; Wadden et al., 2019; Sanh et al., 2019) do not consider entity type for relation evaluation. We also report this result for comparison.

[7]https://github.com/tticoin/LSTM-ER

[8]http://nlp.cs.washington.edu/sciIE/

[9]https://github.com/PaddlePaddle/models/tree/develop/PaddleNLP/Research/ACL2019-ARNOR/

Model	Entity	Relation	Ent + Rel
Sun, 2019a	84.2	–	59.1
Li, 2019$^{\diamond}$	84.8	–	60.2
Sanh, 2019$^{\star,\circ}$	87.5	62.7	–
Luan, 2019$^{\star,\circ}$	88.4	63.2	–
Wadden, 2019 $^{\diamond,\circ}$	**88.6**	63.4	–
SPE$^{\diamond}$	87.2	**66.7**	**63.2**

Table 2: Results on the ACE05 test data. $\diamond$ means that the model use BERT. $\circ$ trains the model in multi-task learning way. $\star$ uses ELMo as token embeddings. "SPE" is the proposed model pre-trained on Wikipedia and BooksCorpus.

Model	Entity	Relation	Ent + Rel
BERT$^{\diamond}$	87.3	65.4	61.7
SpanBERT	**87.9**	65.3	62.2
SPE$^{\diamond}$	87.2	**66.7**	**63.2**
SPE(NYT)$^{\diamond}$	87.4	65.9	63.0
SPE-DS $^{\diamond}$	87.1	64.1	60.1

Table 3: Results on the ACE05 test data. "BERT" is our method without pre-training, which is initialized by BERT$_{\text{BASE}}$ and fine-tuned on ACE05 dataset. "SpanBERT" is similar to "BERT" and is initialized by SpanBERT$_{\text{BASE}}$. "SPE(NYT)" is the proposed model pre-trained on NYT dataset. "SPE-DS" is the proposed model pre-trained on NYT dataset with distantly supervised objectives.

4.1 Results on ACE05

First, we compare our methods with previous works in Table 2. In general, our proposed pre-training method "SPE " [10] achieves significant improvements over all the existing models in two ways of relation evaluation. Particularly, it achieves an improvement of 4.1 units (exactly match) over the LSTM-based GCN joint model (Sun et al., 2019a) and outperforms 3.0 percent (exactly match) comparing with the BERT-based QA model (Li et al., 2019). Comparing with multi-task learning based on ELMo and BERT (Sanh et al., 2019; Luan et al., 2019; Wadden et al., 2019), it also achieves a significant improvement. It is worth noting that our entity detection result underperforms (Luan et al., 2019; Wadden et al., 2019). The major reason is that we do not introduce additional supervision signals in the fine-tuning step, such as coreference resolution and event extraction. However, even without additional multi-task training data, we still achieve the best relation performance, demonstrating the effectiveness of the proposed pre-training method for the entity relation extraction task.

[10]Span and span Pair Encoder (SPE).

Model	Entity	Relation	Ent + Rel
BERT$^\diamond$	87.3	65.4	61.7
SPE$^\diamond$	87.2	**66.7**	**63.2**
- TBO	**87.4**	63.7	61.2
- SPO	87.3	64.4	61.1
- CSPO	87.3	64.5	61.4
- CNN	87.1	64.6	61.2

Table 4: Results on the ACE05 test data in different settings. "BERT" is our method without pre-training, which is initialized by BERT$_{\text{BASE}}$ and fine-tuned on ACE05 dataset. - * is the SPE without * objective, where $* \in \{\text{TBO, SPO, CSPO}\}$; - CNN is the SPE with the pre-trained Transformer, and the rest components of the encoder are randomly initialized.

Next, we compare our method with different pre-training in Table 3. "BERT" and "SpanBERT" have similar relation performances, and "SpanBERT" gets a better entity performance. Our "SPE" outperforms both in terms of relation performance, showing the contribution of the span and span pair representations learned by the proposed objectives. Comparing with distantly supervised per-training, for a fair comparison, we also use the NYT dataset as our pre-training corpus (line 4). In fact, we observe that pre-training results on Wiki and NYT are similar although the NYT data size is smaller (line 3 and line 4). Surprisingly, the distantly supervised pre-training (initialized by BERT$_{\text{BASE}}$) performs poorly even worse than the "BERT" baseline (line 1 and line 5). Our explanation of this phenomenon is that distant supervision introduces larger noisy samples, which results in a negative transfer. Comparing with distantly supervised pre-training, our method does not access any supervised signals that may be noisy, and achieves larger improvement in relation performance over the "BERT" baseline.

Thirdly, we analyze the contributions and effects on different settings (Table 4). We have several observations.

• Comparing with "BERT", "SPE" achieves comparable entity performance and outperforms it with 1.3 points in relation performance. This observation indicates the proposed pre-training objectives can help better learning span-related information, which is crucial for the entity relation extraction task. In addition, entity performance is insensitive to all models (all lines). They fluctuate at 0.4 points.

• When one of the pre-training objectives (line 3-5) is removed, we find the relation performance declines with varying degrees. In particular, the relation performance of "- SPO" drops largely (2.3 points and 2.1 points for both relation evaluations respectively). It demonstrates that the span encoder is quite effective for relation extraction.

• Comparing with "SPE", the relation performance of the "- CNN" decreases sharply (line 6). It shows that, with automatically annotated entities in free texts, the pre-training objectives are able to grasp some useful context information for identifying entities and relations.

	Model	Entity	Relation	Ent + Rel
10%	BERT	73.3	35.2	29.5
	SPE	**74.0**	**40.7**	**34.1**
	SPE(NYT)	74.0	35.5	29.7
	SPE-DS	70.0	27.4	22.3
20%	BERT	75.2	42.8	37.1
	SPE	75.6	**45.6**	**41.9**
	SPE(NYT)	**76.2**	42.8	37.2
	SPE-DS	71.0	33.5	28.5
50%	BERT	**86.1**	62.4	58.7
	SPE	85.4	**62.5**	**58.9**
	SPE(NYT)	85.6	61.9	58.1
	SPE-DS	85.4	60.9	56.8

Table 5: Results on the ACE05 test data by varying on the size of training data in fine-tuning step.

Model	Entity	Relation	Ent + Rel
SPE$^\diamond$	**87.2**	**66.7**	**63.2**
w/o MLP Head	87.0	65.1	61.9
Momentum encoder	**87.2**	65.4	61.7

Table 6: Results on the ACE05 test data in different settings of MoCo framework.

Fourthly, we study the influences of fine-tuning data size. In Table 5, we can see that increasing the size of training data, in general, improve the performances of the entity relation extraction task. When the training data in the fine-tuning step is very small (10%, 20%), our pre-trained model is obviously better than the "BERT" baseline. We attribute these results to the powerful representations learned by our pre-training objectives. In addition, distantly supervised pre-training also have poor performances.

Finally, we test the influences of the component of the MoCo (Table 6) [11]. If we remove the last MLP head projection, the relation performance drops largely, which shows that MLP head pro-

[11]More disscussions and detailed error analyses are in the Appendix B and Appendix C.

Model	Entity	Relation	Ent + Rel
Luan, 2019$^{\star,\circ}$	65.2	41.6	–
Wadden, 2019 $^{\diamond,\circ}$	67.5	**48.4**	–
BERT$^{\diamond}$	67.1	43.6	33.0
SPE$^{\diamond}$	66.9	**45.6**	33.6
SPE(NYT)$^{\diamond}$	**67.7**	44.1	**33.9**
SPE-DS$^{\diamond}$	65.5	45.1	30.8

Table 7: Results on the SciERC test data.

jection substantially improves the quality of the learned span and spar pair representations. There is a similar conclusion on computer vision (Chen et al., 2020a,b). For MoCo framework, we can get two encoders, an encoder updated by back-propagation and a momentum encoder updated by momentum update. In our experiments, we find the former performs better than the latter.

4.2 Results on SciERC

The baseline methods are (Luan et al., 2019), which learns multiple tasks with ELMo embeddings, and (Wadden et al., 2019) which also adopts multi-task learning with BERT. From the upper part of Table 7, both "BERT" and "SPE" significantly outperform (Luan et al., 2019) in entity performance and relation performance. We attribute the phenomenon to the strong ability of BERT. "SPE" performs better than "BERT", which shows the proposed objectives are useful for entity relation extraction, and can integrate span information into the pre-trained model. Our pre-trained models can match previous state-of-the-art method (Wadden et al., 2019), without additional multi-task learning data. In addition, distantly supervised pre-training also have poor performances.

5 Related Work

Research on entity relation extraction has been extensively investigated. Early pipeline methods suffer the error propagation problem (Chan and Roth, 2011; Lin et al., 2016). Joint model can make better use of the complementarity between the entity model and the relation model to alleviate error propagation. A simple method is joint learning through sharing parameters, which means the entity model and the relation model can share some input vectors or sentence encoder. The typical works include tree LSTM-based model over dependency tree (Miwa and Bansal, 2016) and attention-based model without dependency tree (Katiyar and Cardie, 2017). However, this kind of method does not perform joint decoding, and it can not fully exploit the interaction between output entities and relations. To mitigate the above question, many joint decoding algorithms (Fu et al., 2019; Ren et al., 2017; Li et al., 2019) are applied into this task, such as ILP-based joint decoding algorithms (Yang and Cardie, 2013), joint sequence labelling tag set (Zheng et al., 2017), structured perceptron (Li and Ji, 2014), joint MRT (Sun et al., 2018), joint relational triplets extracting (Chen et al., 2019; Zeng et al., 2018), and transition system (Wang et al., 2018). Besides, (Sun et al., 2019a) perform the joint type inference with GCN on an entity-relation bipartite graph. Especially, our joint model for entity relation extraction is derived from (Sun et al., 2019a) without GCN. In this work, we mainly investigate whether pre-training can help entity relation extraction task. For simplicity, our joint model does not perform joint decoding (only sharing parameters). In addition, transfer learning (Sun and Wu, 2019), multi-task learning (Sanh et al., 2019; Wadden et al., 2019; Luan et al., 2019), and reinforcement learning (Takanobu et al., 2019) were also studied.

Pre-trained models have achieved impressive performance on a wide range of downstream tasks in NLP. Different training objectives have been used for different pre-trained models. For example, ELMo (Peters et al., 2018), BERT (Devlin et al., 2018) and UniLM (Dong et al., 2019) learn different token level objectives and sentence level objectives. Despite their great success, they ignore the incorporation of span-related information, which is vital for the entity relation extraction task. (Zhang et al., 2019) demonstrate that informative entities in KGs can enhance language representation with external knowledge. (Sun et al., 2019b) propose entity-level masking and phrase-level masking to enhance language representation. Comparing with (Zhang et al., 2019; Sun et al., 2019b; Joshi et al., 2020), the proposed objectives can not only integrate span information, but also span pair information. Recently, unsupervised contrastive pre-training methods, MoCo (He et al., 2019), SimCLR (Chen et al., 2020a), InfoWord (Kong et al., 2019) and CURL (Srinivas et al., 2020), have led to great empirical success in computer vision, reinforcement learning and NLP. Constrastive learning learns representations by contrasting positive and negative samples. Inspired by (Kong et al., 2019), we learn better span pair representations with contrastive methods, utilizing a large set of negatives without requiring large training batches and extending the MoCo framework with the proposed span pair objective.

6 Conclusion

We propose a pre-training network architecture with three objectives, which can incorporate intra-span and inter-span information into pre-trained models. In comparison to universal pre-trained model, we introduce a span encoder and a span pair encoder. By designning three pre-training objectives, we can learn better pre-trained encoders customized for entity relation extraction task. Experiments on two benchmark datasets demonstrate the effectiveness of the proposed pre-training method.

Acknowledgement

The authors wish to thank the reviewers for their helpful comments and suggestions. This research is (partially) supported by NSFC (62076097, U1609220, 61672231, 61972250), National Key Research and Development Program of China (2018AAA0100704), STCSM (18ZR1411500), the Foundation of State Key Laboratory of Cognitive Intelligence, iFLYTEK(COGOS-20190003), and an open research fund of KLATASDS-MOE. The corresponding authors are Junchi Yan, Yuanbin Wu and Guotong Xie.

References

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. 2019. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint arXiv:1902.09229*.

Yee Seng Chan and Dan Roth. 2011. Exploiting syntactico-semantic structures for relation extraction. In *Proc. of ACL*, pages 551–560.

Jiayu Chen, Caixia Yuan, Xiaojie Wang, and Ziwei Bai. 2019. Mrmep: Joint extraction of multiple relations and multiple entity pairs based on triplet attention. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 593–602.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020b. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *Proc. of ACL*, pages 1409–1418.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2019. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Wei Jia, Dai Dai, Xinyan Xiao, and Hua Wu. 2019. Arnor: Attention regularization based noise reduction for distant supervision relation classification. In *Proc. of ACL*, pages 1399–1408.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Arzoo Katiyar and Claire Cardie. 2017. Going out on a limb: Joint extraction of entity mentions and relations without dependency trees. In *Proc. of ACL*, pages 917–928.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lingpeng Kong, Cyprien de Masson d'Autume, Wang Ling, Lei Yu, Zihang Dai, and Dani Yogatama. 2019. A mutual information maximization perspective of language representation learning. *arXiv preprint arXiv:1910.08350*.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proc. of ACL*, pages 402–412.

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. *arXiv preprint arXiv:1905.05529*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proc. of ACL*, pages 2124–2133.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Xiang Ren, Zeqiu Wu, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, Tarek F Abdelzaher, and Jiawei Han. 2017. Cotype: Joint extraction of typed entities and relations with knowledge bases. In *Proc. of WWW*, pages 1015–1024.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proc. of AAAI*, pages 6949–6956.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. 2020. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019a. Joint type inference on entities and relations via graph convolutional networks. In *Proc. of ACL*, pages 1361–1370.

Changzhi Sun and Yuanbin Wu. 2019. Distantly supervised entity relation extraction with adapted manual annotations. In *Proc. of AAAI*, volume 33, pages 7039–7046.

Changzhi Sun, Yuanbin Wu, Man Lan, Shiliang Sun, Wenting Wang, Kuang-Chih Lee, and Kewen Wu. 2018. Extracting entities and relations with joint minimum risk training. In *Proc. of EMNLP*, pages 2256–2265.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019b. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ryuichi Takanobu, Tianyang Zhang, Jiexi Liu, and Minlie Huang. 2019. A hierarchical framework for relation extraction with reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7072–7079.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Shaolei Wang, Yue Zhang, Wanxiang Che, and Ting Liu. 2018. Joint extraction of entities and relations based on a novel graph scheme. In *IJCAI*, pages 4461–4467.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *Proc. of ACL*, pages 1640–1649.

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*.

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. *arXiv preprint arXiv:1706.05075*.

Appendices

A Neural Architecture for Fine-tuning

Figure 4 displays our neural architecture adopted in the fine-tuning step. The fine-tuning step can be divided into three subtasks as follow:

1. **Entity Span Detection** Training an entity span detector with the sequence labeling framework.
2. **Entity Recognition** Predicting the entity type for each detected entity span.
3. **Relation Classification** Predicting the relation type for each detected entity span pair.

The three subtasks correspond to three objectives, which are optimized simultaneously in fine-tuning.

B More Evaluations

We list performances on each entity type and relation type in Table 8 and Table 9 respectively. Table 8 shows that "BERT" and "SPE" have similar entity performance on different entity types. While Table 9 demonstrates that "SPE" significantly improves both precision and recall on different relation types except for "GEN-AFF".

Figure 5(a) illustrates the relation performaces (exactly match) of "BERT" and "SPE" with respect to the number of entities for each sentence. In general, our "SPE" almost outperforms "BERT" when the number of entities is less than 9. Moreover, the performance of "BERT" and "SPE" both rise with as the number of entities increases, which suggests that more entities will help to identify relations. This result proves that the proposed "SPE" model is able to encode more powerful representations of span and span pair when lacking entity information. In other words, our "SPE" is robust to more sparse situations that are common in reality.

Figure 5(b) illustrates the relation performaces (exactly match) of "BERT" and "SPE" with respect to the sentence length of each sentence. We find that "SPE" achieves superior performances compared to "BERT" on different sentence length except that sentence length is between 31 and 50. This result demonstrates that our "SPE" model can handle too long or too short sentences, while the performance sharply decreases when sentence length is between 31 and 50. We think the superior performances of "SPE" verify the effectiveness of the proposed method. Meanwhile, it is valuable to solve the problem of performance decline on medium sentence length in future work.

Entity Type	Model	P	R	F
WEA (109)	BERT	77.7	**79.8**	78.7
	SPE	**81.7**	78.0	**79.8**
FAC (286)	BERT	77.2	**78.3**	**77.8**
	SPE	**77.7**	76.6	77.1
VEH (116)	BERT	**85.3**	80.2	**82.7**
	SPE	81.7	**81.0**	81.4
LOC (136)	BERT	70.8	75.0	72.9
	SPE	**71.0**	**77.2**	**73.9**
PER (2928)	BERT	90.2	**93.6**	**91.8**
	SPE	**90.4**	93.0	91.7
GPE (1013)	BERT	**88.5**	88.6	**88.5**
	SPE	85.3	**90.2**	87.7
ORG (817)	BERT	78.0	74.9	76.4
	SPE	**79.6**	**75.2**	**77.3**

Table 8: The entity performance of "BERT" and "SPE" on different entity types. The numbers in the first column are counts of entities in the ACE05 test set.

Relation Type	Model	P	R	F
ART (146)	BERT	**69.0**	41.1	51.5
	SPE	64.4	**44.5**	**52.6**
PART-WHOLE (175)	BERT	56.8	57.1	57.0
	SPE	**60.3**	**60.0**	**60.2**
PER-SOC (73)	BERT	68.5	68.5	68.5
	SPE	**72.0**	**74.0**	**73.0**
PHYS (278)	BERT	53.4	47.8	50.5
	SPE	**60.7**	**53.2**	**56.7**
GEN-AFF (99)	BERT	**62.8**	**49.5**	**55.4**
	SPE	56.8	46.5	51.1
ORG-AFF (354)	BERT	72.0	71.2	71.6
	SPE	**76.7**	**72.3**	**74.4**

Table 9: The relation performance (exactly match) of "BERT" and "SPE" on different relation types. The numbers in the first column are counts of relations in the ACE05 test set.

Finally, we report the performance of "SciBERT" on the SciERC dataset in Table 10. In the experiment, we directly replace BERT with SciBERT and train the model on the SciERC dataset to get the result. We find that the performance of "SciERC" is superior to "BERT" and "SPE", which suggests that the data resources from specific domains are quite important for some tasks. But we believe that we can achieve a higher score on the SciERC dataset when we use the same pre-training dataset as "SciBERT".

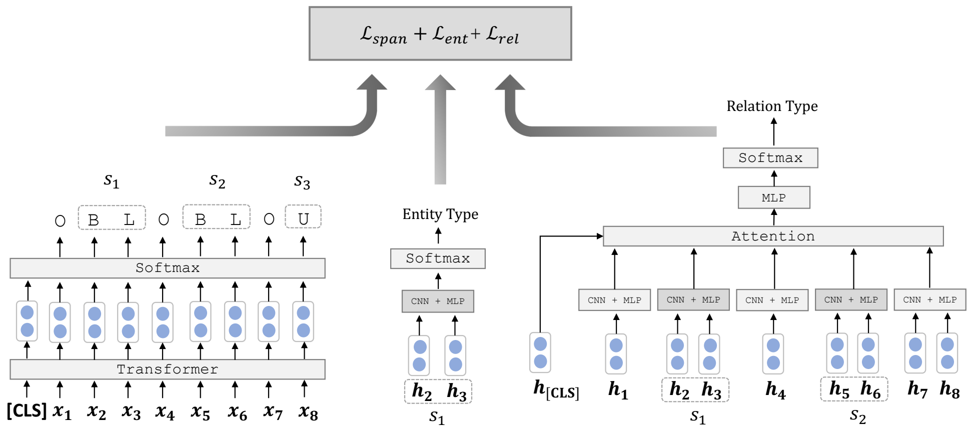
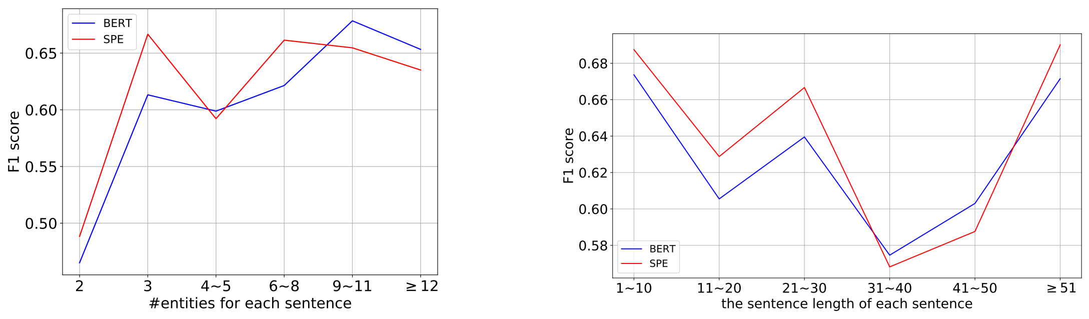

$\mathcal{L}_{span} + \mathcal{L}_{ent} + \mathcal{L}_{rel}$
Relation Type
Softmax
MLP
Attention
CNN + MLP
Entity Type
Softmax
CNN + MLP
s_1 s_2 s_3
O B L O B L O U
Softmax
Transformer
[CLS] x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
h_2 h_3
$h_{[CLS]}$ h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8

Figure 4: Overview of our neural architecture for fine-tuning. Fine-tuning loss consists of three parts: entity span detection loss, entity recognition loss, and relation classification loss.



(a) The number of entities for each sentence.

(b) The sentence length of each sentence.

Figure 5: The relaiton F1 score (exactly match) with respect to the number of entities and the sentence length for each sentence on ACE05 test data.

Model	Entity	Relation	Ent + Rel
BERT$^\diamond$	67.1	43.6	33.0
SciBERT$^\diamond$	**68.0**	**47.6**	**34.6**
SPE$^\diamond$	66.9	45.6	33.6

Table 10: Results on the SciERC test data.

C Error Analyses

In this section, we compare the performances of "BERT" and "SPE" on concrete examples. These examples are excerpted from the results of "BERT" and "SPE" on ACE05 test data. In the following examples, we will use notations like "[entity span]$^{\texttt{ENT-TYPE}}_{\texttt{[REL-TYPE\_REL-ID]}}$". It means that an entity mention ("entity span") has an entity type `ENT-TYPE`, and (optionally) participates in one or more relations, which can be identified with `REL-TYPE` and `REL-ID`.

First, "SPE" can detect more entities participating in relations. For S1, "SPE" identifies [aol time warner] as the entity `ORG`, while "BERT" identifies [time warner] as the entity `ORG`. Then "SPE" further identifies a `ORG-AFF` relation between [his] and [aol time warner] but "BERT" does not. This may be one of the reasons why "BERT" and "SPE" have similar entity performance while the relation performance of "SPE" is significantly superior to "BERT".

Next, we give two examples in more complex sentences. For S2, [army] participates two relations (`PART-WHOLE-1` and `PART-WHOLE-2`), "BERT" identifies an wrong relation `ORG-AFF-1` while the results of "SPE" are correct. For S3, the distance between [ship] and [terrorists] is quite long, "SPE" correctly find the relation `ART-2` while "BERT" does not. Hence, we think that our "SPE" can handle more complex situations such as relation overlapping and distant entities.

D Pseudocode (PyTorch-like)

Figure 6 shows the PyTorch-like pseudocode of our span pair contrastive learning.

S1	. . . our founder here at cnn , ted tuener , has sold more than half 0 [his]$^{\texttt{PER}}_{\texttt{ORG-AFF-1}}$ stake in [aol time warner]$^{\texttt{ORG}}_{\texttt{ORG-AFF-1}}$.
BERT	. . . our founder here at cnn , ted tuener , has sold more than half 0 [his]$^{\texttt{PER}}$ stake in aol [time warner]$^{\texttt{ORG}}$.
SPE	. . . our founder here at cnn , ted tuener , has sold more than half 0 [his]$^{\texttt{PER}}_{\texttt{ORG-AFF-1}}$ stake in [aol time warner]$^{\texttt{ORG}}_{\texttt{ORG-AFF-1}}$.

S2	. . . troops from the [u.s]$^{\texttt{GPE}}_{\texttt{PART-WHOLE-1}}$. [army]$^{\texttt{ORG}}_{\texttt{PART-WHOLE-1\|PART-WHOLE-2}}$'s [101st airborne division]$^{\texttt{ORG}}_{\texttt{PART-WHOLE-2}}$ went to the site on friday . . .
BERT	. . . troops from the [u.s]$^{\texttt{GPE}}_{\texttt{PART-WHOLE-1}}$. [army]$^{\texttt{ORG}}_{\texttt{PART-WHOLE-1\|ORG-AFF-1}}$'s [101st airborne division]$^{\texttt{ORG}}_{\texttt{ORG-AFF-1}}$ went to the site on friday . . .
SPE	. . . troops from the [u.s]$^{\texttt{GPE}}_{\texttt{PART-WHOLE-1}}$. [army]$^{\texttt{ORG}}_{\texttt{PART-WHOLE-1\|PART-WHOLE-2}}$'s [101st airborne division]$^{\texttt{ORG}}_{\texttt{PART-WHOLE-2}}$ went to the site on friday . . .

S3	this was the [italian]$^{\texttt{GPE}}_{\texttt{ART-1}}$ [ship]$^{\texttt{VEH}}_{\texttt{ART-1\|ART-2}}$ that was taken – that was captured by [palestinian]$^{\texttt{PER}}_{\texttt{GEN-AFF-1}}$ [terrorists]$^{\texttt{GPE}}_{\texttt{GEN-AFF-1\|ART-2}}$ back in 1985 .
BERT	this was the [italian]$^{\texttt{GPE}}_{\texttt{ART-1}}$ [ship]$^{\texttt{VEH}}_{\texttt{ART-1}}$ that was taken – that was captured by [palestinian]$^{\texttt{PER}}_{\texttt{GEN-AFF-1}}$ [terrorists]$^{\texttt{GPE}}_{\texttt{GEN-AFF-1}}$ back in 1985 .
SPE	this was the [italian]$^{\texttt{GPE}}_{\texttt{ART-1}}$ [ship]$^{\texttt{VEH}}_{\texttt{ART-1\|ART-2}}$ that was taken – that was captured by [palestinian]$^{\texttt{PER}}_{\texttt{GEN-AFF-1}}$ [terrorists]$^{\texttt{GPE}}_{\texttt{GEN-AFF-1\|ART-2}}$ back in 1985 .

Table 11: Examples from the ACE05 dataset with label annotations from "BERT" and "SPE" for comparison.

```
# f_m: momentum span pair encoder
# f_e: span pair encoder
# span_pair_queue: dictionary as a que of span pair representation (CxK)
# context_queue: dictionary as a que of context representation (CxK)
# m: momentum
# t: temperature

f_m.params = f_e.params  # initialize
for x in loader:  # load a minibatch x with N samples
    x_span_pair = mask(x)  # mask context part
    x_context = mask(x)  # mask span pair

    span_pair_q = f_e.forward(x_span_pair)  # span pair queries: NxC
    context_q = f_e.forward(x_context)  # context queries: NxC
    span_pair_k = f_m.forward(x_span_pair)  # span pair keys: NxC
    context_k = f_m.forward(x_span_pair)  # context keys: NxC
    span_pair_k = span_pair_k.detach()  # no gradient to span pair keys
    context_k = context_k.detach()  # no gradient to context keys

    # span pair positive logits: Nx1
    l_span_pair_pos = bmm(span_pair_q.view(N, 1, C), context_k.view(N, C, 1))
    # context positive logits: Nx1
    l_context_pos = bmm(context_q.view(N, 1, C), span_pair_k.view(N, C, 1))

    # span pair negative logits: NxK
    l_span_pair_neg = mm(span_pair_q.view(N, C), context_queue.view(C, K))
    # context negative logits: NxK
    l_context_neg = mm(context_q.view(N, C), span_pair_queue.view(C, K))

    # span pair logits: Nx(1+K)
    span_pair_logits = cat([l_span_pair_pos, l_span_pair_neg], dim=1)
    # context logits: Nx(1+K)
    context_logits = cat([l_context_pos, l_context_neg], dim=1)

    # contrastive loss
    labels = zeros(N)  # positives are the 0-th
    span_pair_loss = CrossEntropyLoss(span_pair_logits / t, labels)
    context_loss = CrossEntropyLoss(context_logits / t, labels)
    loss = span_pair_loss + context_loss

    # gradient back-propagation: span pair encoder f_e
    loss.backward()
    update(f_e.params)

    # momentum update: key network
    f_m.params = m * f_m.params + (1 - m) * f_e.params

    # update dictionary
    # enqueue the current minibatch
    enqueue(span_pair_queue, span_pair_k)
    enqueue(context_queue, context_k)

    # dequeue the earliest minibatch
    dequeue(span_pair_queue)
    dequeue(context_queue)
```

Figure 6: Pseudocode of Contrastive Span Pair Objective (CSPO) in a PyTorch-like style.