SEPT: Improving Scientific Named Entity Recognition with Span Representation

Tan Yan, Heyan Huang, Xian-Ling Mao
Department of Computer Science and Technology,
Beijing Institute of Technology, China
yantan@bit.edu.com

Abstract

We introduce a new scientific named entity recognizer called SEPT, which stands for **S**pan **E**xtractor with **P**re-trained **T**ransformers. In recent papers, span extractors have been demonstrated to be a powerful model compared with sequence labeling models. However, we discover that with the development of pre-trained language models, the performance of span extractors appears to become similar to sequence labeling models. To keep the advantages of span representation, we modified the model by under-sampling to balance the positive and negative samples and reduce the search space. Furthermore, we simplify the origin network architecture to combine the span extractor with BERT. Experiments demonstrate that even simplified architecture achieves the same performance and SEPT achieves a new state of the art result in scientific named entity recognition even without relation information involved[1].

1 Introduction

With the increasing number of scientific publications in the past decades, improving the performance of automatically information extraction in the papers has been a task of concern. Scientific named entity recognition is the key task of information extraction because the overall performance depends on the result of entity extraction in both pipeline and joint models (Yadav and Bethard, 2018).

Named entity recognition has been regarded as a sequence labeling task in most papers (Ma and Hovy, 2016). Unlike the sequence labeling model, the span-based model treats an entity as a whole span representation while the sequence labeling model predicts labels in each time step independently. Recent papers (Luan et al., 2018; He et al., 2018) have shown the advantages of span-based models. Firstly, it can model overlapping and nested named entities. Besides, by extracting the span representation, it can be shared to train in a multitask framework. In this way, span-based models always outperform the traditional sequence labeling models. For all the advantages of the span-based model, there is one more factor that affects performance. The original span extractor needs to score all spans in a text, which is usually a $O(n^2)$ time complexity. However, the ground truths are only a few spans, which means the input samples are extremely imbalanced.

Due to the scarcity of annotated corpus of scientific papers, the pre-trained language model is an important role in the task. Recent progress such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018), BERT (Devlin et al., 2018) improves the performance of many NLP tasks significantly including named entity recognition. In the scientific domain, SciBERT (Beltagy et al., 2019) leverages a large corpus of scientific text, providing a new resource of the scientific language model. After combining the pre-trained language model with span extractors, we discover that the performance between span-based models and sequence labeling models become similar.

In this paper, we propose an approach to improve span-based scientific named entity recognition. Unlike previous papers, we focus on named entity recognition rather than multitask framework because the multitask framework is natural to help. We work on single-tasking and if we can improve the performance on a single task, the benefits on many tasks are natural.

To balance the positive and negative samples and reduce the search space, we remove the pruner and modify the model by under-sampling.

[1] Code is available at: https://github.com/ethan-yt/sept

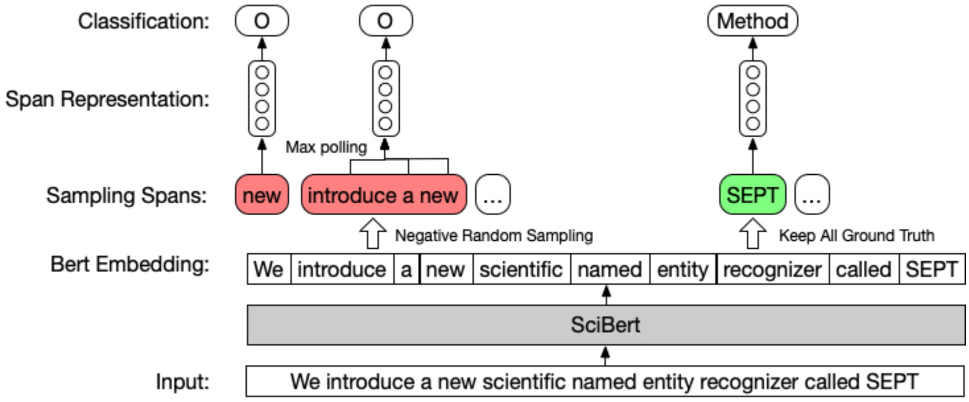

Figure 1: The overview architecture of SEPT. Firstly, feeding the entire abstract into the model and obtain BERT embeddings for each token (word piece). Then, in the training phase, rather than enumerate all spans: (a) For negative spans, we sample them randomly. (b) For ground truths, we keep them all. We use the maximum pooling to obtain the span representation. Finally, each span is classified into different types of entities by an MLP.

Furthermore, because there is a multi-head self-attention mechanism in transformers and they can capture interactions between tokens, we don't need more attention or LSTM network in span extractors. So we simplify the origin network architecture and extract span representation by a simple pooling layer. We call the final scientific named entity recognizer SEPT.

Experiments demonstrate that even simplified architecture achieves the same performance and SEPT achieves a new state of the art result compared to existing transformer-based systems.

2 Related Work

Span-based Models The first Span-based model was proposed by Lee et al. (2017), who apply this model to a coreference resolution task. Later, He et al. (2018); Luan et al. (2018) extend it to various tasks, such as semantic role labeling, named entity recognition and relation extraction. Luan et al. (2018) is the first one to perform a scientific information extraction task by a span-based model and construct a dataset called SCIERC, which is the only computer science-related fine-grained information extraction dataset to our best knowledge. Luan et al. (2019) further introduces a general framework for the information extraction task by adding a dynamic graph network after span extractors.

They use ELMo as word embeddings, then feed these embeddings into a BiLSTM network to capture context features. They enumerate all possible spans, each span representation is obtained by some attention mechanism and concatenating strategy. Then score them and use a pruner to remove spans that have a lower possibility to be a span. Finally, the rest of the spans are classified into different types of entities.

SciBert Due to the scarcity of annotated corpus in the scientific domain, SciBert (Beltagy et al., 2019) is present to improve downstream scientific NLP tasks. SciBert is a pre-trained language model based on BERT but trained on a large scientific corpus.

For named entity recognition task, they feed the final BERT embeddings into a linear classification layer with softmax output. Then they use a conditional random field to guarantee well-formed entities. In their experiments, they get the best result on finetuned SciBert and an in-domain scientific vocabulary.

3 Models

Our model is consists of four parts as illustrated in figure 1: Embedding layer, sampling layer, span extractor, classification layer.

3.1 Embedding layer

We use a pre-trained SciBert as our context encoder. Formally, the input document is represented as a sequence of words $D = \{w_1, w_2, \ldots, w_n\}$, in which n is the length of the document. After feeding into the SciBert model, we obtain the context embeddings $E = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$.

3.2 Sampling layer

In the sampling layer, we sample continuous substrings from the embedding layer, which is also called span. Because we know the exact label of each sample in the training phase, so we can train the model in a particular way. For those negative samples, which means each span does not belong to any entity class, we randomly sampling them rather than enumerate them all. This is a simple but effective way to improve both performance and efficiency. For those ground truth, we keep them all. In this way, we can obtain a balanced span set: $S = S_{neg} \cup S_{pos}$. In which $S_{neg} = \{s'_1, s'_2, \dots, s'_p\}$, $S_{pos} = \{s_1, s_2, \dots, s_q\}$. Both s and s' is consist of $\{\mathbf{e}_i, \dots, \mathbf{e}_j\}$, i and j are the start and end index of the span. p is a hyperparameter: the negative sample number. q is the positive sample number. We further explore the effect of different p in the experiment section.

3.3 Span extractor

Span extractor is responsible to extract a span representation from embeddings. In previous work (Lee et al., 2017), endpoint features, content attention, and span length embedding are concatenated to represent a span. We perform a simple max-pooling to extract span representation because those features are implicitly included in self-attention layers of transformers. Formally, each element in the span vector is:

$$\mathbf{r}^t(s) = \max\{\mathbf{e}_i^t, \dots, \mathbf{e}_j^t\} \tag{1}$$

t is ranged from 1 to embedding length. $\mathbf{e}_i, \dots, \mathbf{e}_j$ are embeddings in the span s. In this way, we obtain a span representation, whose length is the same as word embedding.

3.4 Classification layer

We use an MLP to classify spans into different types of entities based on span representation $\mathbf{r}$. The score of each type l is:

$$\Phi_l(\mathbf{r}) = w_a^\intercal \, \mathrm{MLP}(\mathbf{r}) \tag{2}$$

We then define a set of random variables, where each random variable y_s corresponds to the span s, taking value from the discrete label space $\mathcal{L}$. The random variables y_s are conditionally independent of each other given the input document D:

$$P(Y|D) = \prod_{s \in S} P(y_s|D) \tag{3}$$

$$P(y_s = l|D) = \frac{\exp(\Phi_l(\mathbf{r}(s)))}{\sum_{l' \in \mathcal{L}} \exp(\Phi_{l'}(\mathbf{r}(s)))} \tag{4}$$

For each document D, we minimize the negative log-likelihood for the ground truth Y^*:

$$\mathcal{J}(D) = -\log P(Y^*|D) \tag{5}$$

3.5 Evaluation phase

During the evaluation phase, because we can't peek the ground truth of each span, we can't do negative sampling as described above. To make the evaluation phase effective, we build a pre-trained filter to remove the less possible span in advance. This turns the task into a pipeline: firstly, predict whether the span is an entity, then predict the type. To avoid the cascading error, we select a threshold value to control the recall of this stage. In our best result, we can filter 73.8% negative samples with a 99% recall.

4 Experiments

In our experiment, we aim to explore 4 questions:

1. How does SEPT performance comparing to the existing single task system?
2. How do different numbers of negative samples affect the performance?
3. How a max-pooling extractor performance comparing to the previous method?
4. How does different threshold effect the filter?

Each question corresponds to the subsection below. We document the detailed hyperparameters in the appendix.

4.1 Overall performance

Models	Precision	Recall	F1
BiLSTM(Glove)	0.521	0.506	0.513
BiLSTM(ELMo)	0.577	0.590	0.583
BiLSTM(SciBERT)	0.635	0.684	0.659
SCIIE(ELMo)	0.604	0.624	0.613
SCIIE(SciBERT)	**0.651**	0.677	0.664
SEPT	0.642	**0.716**	**0.677**

Table 1: Overall performance of scientific named entity recognition task. We report micro F1 score following the convention of NER task. All scores are taken from the test set with the corresponding highest development score.

Table 1 shows the overall test results. We run each system on the SCIERC dataset with the same split scheme as the previous work. In BiLSTM model, we use Glove (Pennington et al., 2014), ELMo (Peters et al., 2018) and SciBERT(fine-tuned) (Beltagy et al., 2019) as word embeddings and then concatenate a CRF layer at the end. In SCIIE (Luan et al., 2018), we report single task scores and use ELMo embeddings as the same as they described in their paper. To eliminate the effect of pre-trained embeddings and perform a fair competition, we add a SciBERT layer in SCIIE and fine-tune model parameters like other BERT-based models.

We discover that performance improvement is mainly supported by the pre-trained external resources, which is very helpful for such a small dataset. In ELMo model, SCIIE achieves almost 3.0% F1 higher than BiLSTM. But in SciBERT, the performance becomes similar, which is only a 0.5% gap.

SEPT still has an advantage comparing to the same transformer-based models, especially in the recall.

4.2 Different negative samples

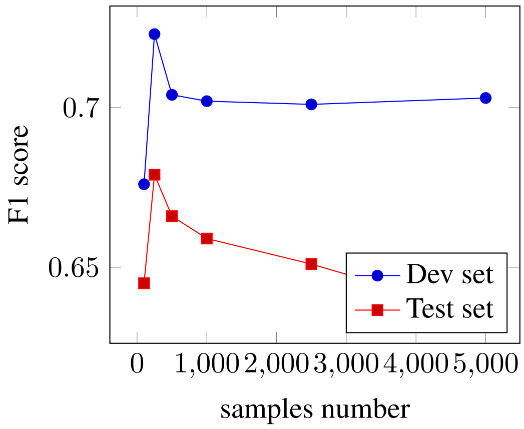

Figure 2: Different negative samples affect F1 scores in different datasets.

As shown in figure 2, we get the best F1 score on around 250 negative samples. This experiment shows that with the number of negative samples increasing, the performance becomes worse.

4.3 Ablation study: Span extractor

In this experiment, we want to explore how different parts of span extractor behave when a span extractor applied to transformers in an ablating study.

As shown in table 2, we discovered that explicit features are no longer needed in this situation. Bert model is powerful enough to gain these features and defining these features manually will bring side effects.

Models	Precision	Recall	F1
SEPT	0.642	0.716	0.677
SEPT(+length)	0.617	0.720	0.664
SEPT(+endpoint)	0.633	0.719	0.674
SEPT(+attention)	0.628	0.726	0.674

Table 2: Ablation study on different part of span extractor.

4.4 Threshold of filter

In the evaluation phase, we want a filter with a high recall rather than a high precision. Because a high recall means we won't remove so many truth spans. Moreover, we want a high filtration rate to obtain a few remaining samples.

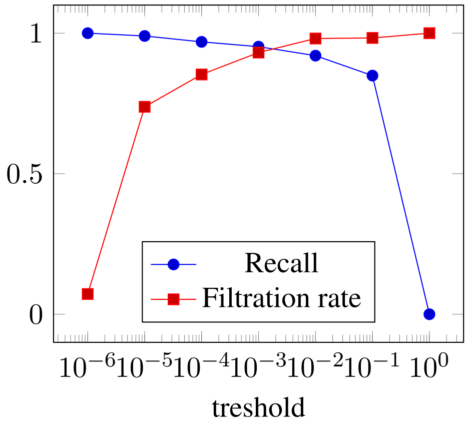

Figure 3: Recall and Filteration rate on different threshold.

As shown in figure 3, there is a positive correlation between threshold and filter rate, and a negative correlation between threshold and recall. We can pick an appropriate value like 10^{-5}, to get a higher filtration rate relatively with less positive sample loss (high recall). We can filter 73.8% negative samples with a 99% recall. That makes the error almost negligible for a pipeline framework.

5 Conclution

We presented a new scientific named entity recognizer SEPT that modified the model by undersampling to balance the positive and negative samples and reduce the search space.

In future work, we are investigating whether the SEPT model can be jointly trained with relation and other metadata from papers.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: Pretrained language model for scientific text. In *EMNLP*.

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*.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. *arXiv preprint arXiv:1805.04787*.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *arXiv preprint arXiv:1808.09602*.

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*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Vikas Yadav and Steven Bethard. 2018. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158.