An Empirical Study of Span Modeling in Science NER

Xiaorui Jiang

Original citation & hyperlink:
Jiang, X., An Empirical Study of Span Modeling in Science NER. In *International Conference on Theory and Practice of Digital Libraries* (pp. 41-48). Springer, Cham. (2021)
Published as part of the series Lecture Notes in Computer Science, Vol 12866.

DOI 10.1007/978-3-030-86324-1_4
ISSN 0302-9743

Publisher: Springer

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-030-86324-1_4

An Empirical Study of Enhanced Entity Representations for Span-Based Scientific Named Entity Recognition

Xiaorui Jiang[1][0000-0003-4255-5445]

[1] Coventry University, Priory St, Coventry, CV1 5FB, UK
xiaorui.jiang@coventry.ac.uk

Abstract. Span-based models have become popular for a wide range of NLP problems. Quite a few deep learning based methods for representing token spans have been proposed in past research. Recent experimental studies mainly focused on such representations' performances in different end tasks and only tested a rather limited number of span modeling options. This paper expands this line of empirical investigation by switching the emphasis to span representation methods with a focus on scientific named entity recognition (science NER). In this study, many classical span representation methods and their combinations are tested on two challenging recent scientific information extraction datasets. Furthermore, different context encoding methods are also evaluated, as well as their combinations with span encoding. I find that some classical span encodings, such as the concatenation and difference/sum of span endpoints, prove to be effective in most cases, either used alone or to enhance other span encoding methods. Indeed, more comprehensive entity representations prove to improve over the previous state of the art of science NER. In contrast to span encoding, little regularity can be observed in different context representations' performances. In addition, the choice of (the combination of) span and context representations seem to also vary with datasets that are of different domains, annotations, and nature.

Keywords: Scientific Named Entity Recognition, Span-Based Model, Span Representation, Context Representation, SciBERT

1 Representation Learning in Span-based Science NER

1.1 Span Modeling in NLP

The biggest reason for this paper to focus on span-based NER models is that span modelling has been recognized as a general methodology for an extremely wide range of, if not the full range of, natural language processing problems. The rich application context of the concept of *span* has resulted in an even richer set of modeling options. Specific to NER, one advantage of span-based model is its capability of dealing with nested entities, which happen fairly frequently in many application domains, e.g. Peking University (ORG, i.e. organization) containing another entity Peking (LOC, i.e. location),

As a matter of fact, many natural language processing tasks require the recognition of span – contiguous sequence of tokens. For example, keyphrase extraction identifies

the starts and ends of salient spans of tokens [1], while NER also determines the type of each recognized span [2-9]. Semantic role labeling can also be formulated as a span selection problem by scoring possible argument spans for each semantic role label of a verb [10]. Recently extractive question answering such as SQUAD [11-12] requires extracting, for each question, a contiguous token span as the answer. In addition, recognized spans also serve as the foundation of more complex NLP tasks involving relations between multiple spans, such as coreference resolution (as recognizing coreference relations between spans of entity mentions) [13], relation extraction (as linking and classifying relations between several entity spans) [14], constituency parsing (as grouping contiguous sequences of tokens into syntactic constituents and contiguous constituents to higher level constituents) [15], aspect-based sentiment analysis (as linking aspect expressions and opinion expressions which are both contiguous spans of tokens) [16], and etc. A recent paper about using span-relation modeling as a general NLP methodology discusses the approaches for these different tasks in detail [17]. Therefore, it is useful and important to investigate the promising choices of span modeling.

1.2 Span Based NER

This paper emphasizes on investigating the performances of different span-based entity representation methods. There are generally two technical routes to model an NER problem: as a sequential tagging problem or as a span classification problem.

In sequential tagging, entity type labels are prepended with prefixes to indicate the start (B-), inside (I-), end (E-) or outside (O-) of potential entities, and a machine learning model chooses for each token an appropriate label from the expanded label set [1-3]. More recent models typically add a CRF (Conditional Random Field) layer before classification to account the inter-token constraints on the valid labels to choose [2-3]. One disadvantage of sequential NER is that it needs further postprocessing to deal with nested entities, which however is often suboptimal. In addition, each token is encoded by a deep neural network and then sent to classification without taking into consideration the encodings of other tokens in a candidate span (a.k.a. *entity proposal*).

Span-based methods naturally solve these two issues. Candidate entity spans are enumerated, and the tokens in each candidate span are further processed to a feature vector, called *span representation*, for the entity proposal [4]. There can be many design options for encoding a candidate entity span. Typically, the two endpoints and a summary generated for the span tokens were widely adopted span representations in previous studies [4-6]. There are also rich design options for other aspects in span-based models, such as further encoding candidate spans [7], generating a span's *context representation*[1] as part of the entity representation [6-7], and performing entity boundary detection in a similar way to sequential tagging to constrain candidate span enumeration [7-8] etc. This paper focuses on the design options for span representation and context representation to encode entity proposals.

[1] Span/context representation is used interchangeably with span/context encoding in the paper.

1.3 Representation Probing

Parallel to this work are the studies of probing different aspects of representation learning for NER. First, different token embeddings can be employed. A comparative study of the performances of different word embeddings in NER was given in [18]. Typically, earlier works based on BiLSTM also incorporated character-level features encoded by stacking a separate CNN or BiLSTM layer on the character sequences of tokens, the parameters of which are learned through training [1-3]. In pretrained language models like FLAIR embeddings [19], character-level information has already been incorporated into the contextual word embeddings, thus the separate character-level feature extraction layer can be omitted. Word-piece models like BERT (and the SciBERT variant [20] used by this paper) can be seen as having implicitly encoded character-level information because rare words, which are often named entities especially in scientific domains, are tokenized into sequences of characters. Out of this reason, this paper does not consider character embeddings, as in BERT NER [3].

There are also studies about probing different and supplementary domain data to enhance science NER. In [21], it was found that a domain-independent classifier trained on 10 different scientific domains performed consistently better than all domain-specific classifiers that were trained on individual domains. This effect was very prominent for highly multidisciplinary domains such as material sciences. A recent work evaluated various span representations on six different NLP tasks including a general domain NER task [22]. While the authors put emphasis on which span representation might be appropriate for which end task, they only tested a very limited number of rather simple span representations. Neither combinations of span representations nor their combination with context representations were considered. Different from the above, this paper pays specific attention to science NER which is more challenging and less evaluated by the NLP community. More specifically, I will test a much wider range of span representations and a large number of combinations of span and context representations by fine-tuning SciBERT. Different from this study, the only relevant work on science NER that I am aware of focused on comparing different deep learning architectures [23].

2 Entity Representation Learning Methods

Notations are first formalized. Subsequent sections will present the details of the tested options of entity representation. A sentence is a sequence of tokens $t_1, \ldots, t_N$. Special tokens [CLS] and [SEP] are prepended and appended, respectively, to the token sequence. Pretrained SciBERT embeddings $\mathbf{e}$ as inputs will be sent to BERT layers and each token will be encoded to a token representation vector $\mathbf{h}$, output by the final layer.

For each span $s = [i, j]$ as an entity proposal, its entity representation has two parts, the span representation $\mathbf{s}_{i...j}$ with regard to $t_i, \ldots, t_j$, and the optional context representations with regard to $t_1, \ldots, t_{i-1}$, and $t_{j+1}, \ldots, t_N$. In this study, left and right contexts are encoded separately into $\mathbf{c}^l$ and $\mathbf{c}^r$. Usually, span length is also encoded into a learnable vector $\mathbf{w}$. Thus, each entity proposal is encoded into $\mathbf{f} = [\mathbf{s}; \mathbf{c}^l; \mathbf{c}^r; \mathbf{w}]$.

Finally, feature vector $\mathbf{f}$ is sent into a linear classifier. Note that, there can be a non-linear feature projection on $\mathbf{f}$ before classification or the final classifier can be a

stronger nonlinear classifier such as an MLP (Multi-Layer Perceptron). This paper deliberately uses a weak single-layer feed-forward network (FFN, i.e., a fully connected linear projection followed by Tahn nonlinear activation) for classification, hoping to focus on the impact of entity representation options and to force the model to learn a good enough representation. Although this treatment sacrifices a bit the comparability to previous studies, it can give stronger evidence for the superiority of (a combination of) certain span and context encodings if good result is achieved.

2.1 Span Representation

This paper classifies the large number of span representation proposals into three categories. Different from [22] where only one category was used in the probing experiments, various combinations of the span representation methods belong to the three categories have been proposed in the literature and will be tested in this paper.

The `SPAN_ENDS` category considers concatenating the two span endpoints $[\mathbf{h}_i, \mathbf{h}_j]$ (CONCAT). This is the most common way of encoding a candidate entity span and was used in most span-based models for a wide range of applications. In a number of complex models, only `SPAN_ENDS` is adopted for NE classification [9, 24]. Therefore, there are two setups for `SPAN_ENDS`: CONCAT or NONE.

The `SPAN_SUMM` category to summarize the tokens in a span into a fixed dimensional feature vector $\hat{\mathbf{h}}$. Like in [22], I found that average pooling or mean pooling does not perform well. Being a much less popular method too, average pooling or mean pooling is not considered in this paper. The first option is *max pooling* (MAX_POOL) which takes, for each dimension, the maximum value over time of a sequence of token encodings, here the maximum over $\mathbf{h}_i$, …, $\mathbf{h}_j$, as the span summary vector. It was recognized as a surprisingly effective feature extraction method and was widely adopted in the literature [7, 25-26]. *Self attentive summary* (SEFL_ATT) is another span summary method to be tested [27]. It first calculates attention weights to each span token by Eq. (1) and then computes the weighted sum of span token representations by Eq. (2), where FFN encodes $\mathbf{h}_t$ using a single-layer FFN and softmax normalizes the attention weights. Note that, a simpler form of self attention was also tested where α_t is replaced by $\alpha_t = softmax(\mathbf{w}^a \cdot \mathbf{h}_t)$ as in [22]. But no significant difference was detected, thus only results of Eq. (1) will be reported. Therefore, there are three setups for `SPAN_SUMM`: MAX_POOL, SELF_ATT and NONE.

$$\alpha_t = \text{softmax}\big(\mathbf{w}^a \cdot \text{FFN}(\mathbf{h}_t)\big) \quad \text{for } t = i, \dots, j \tag{1}$$

$$\hat{\mathbf{h}} = \text{SELF\_ATT}([\mathbf{h}_i, \dots, \mathbf{h}_j]) = \sum_{t=i}^{j} \alpha_t \cdot \mathbf{h}_t \tag{2}$$

The `END_COMB` category accounts for combing the two endpoints. This paper considers a common way called DIFF_SUM which concatenates the difference and sum of the two endpoints: $\tilde{\mathbf{h}} = [\mathbf{h}_i - \mathbf{h}_j;\ \mathbf{h}_i + \mathbf{h}_j]$ [28-29]. Thus, `END_COMB` has two setups: DIFF_SUM or NONE. There is an interesting alternative, the Hadamard product of span ends, i.e. $\hat{\mathbf{h}} = \mathbf{h}_i \circ \mathbf{h}_j$ [6]. A similar combination called *coherence term* originally used for question answering [30] was also tested in [22]. They are left to future work.

2.2 Context Representation

Due to computational expensiveness of fine-tuning, this paper reports on the following four choices of context representation (CONTEXT): (1) NONE for span-only representation; (2) SELF_ATT as a local context representation applied to both the left and right contexts [31]; (3) MAX_POOL as a local context representation too; (4) [CLS] as a global context representation, a method special to BERT/SciBERT-based models [7]. Context representation alone is not considered; it must go with one span representation. Nevertheless, there are other options for context encoding, including span-aware attention [8] and its location-weighted variant [32]. There was a special context representation method only used in [29], $[\mathbf{h}_i - \mathbf{h}_{j+1};\ \mathbf{h}_j - \mathbf{h}_{i-1}]$ (I call it OFFSET_DIFF). The accompanying SPAN_END option is special too $[\mathbf{h}_{i-1}, \mathbf{h}_{j+1}]$ (I call it OFFSET). These rare options are left to future work.

2.3 Probing Objectives

I am mainly interested in the answers for four probing questions about science NER.

*Q*1: Which SPAN_SUMMARY method is better for science NER?

*Q*2: Whether SPAN_ENDS is the most useful feature for span representation?

*Q*3: Whether SPAN_COMB helps to improve science NER performance?

*Q*4: Whether and how CONTEXT can be promising for science NER?

Especially, this paper wants to investigate whether there is a universally better-than-others option or combination of modeling options for entity representation, which consistently performs better across all (or at least most) other settings and different datasets.

3 Scientific NER Experiments

3.1 Datasets and Experimental Setup

Probing experiments are done on two recent challenging scientific information extraction datasets popular among the NLP community. The first is SciERC [6] – 2687 sentences from 500 abstracts of AI papers annotated with entity, relation and coreference information with a 1861/275/551 train/dev/test split. Only the 8089 entity annotations are used. The six entity types are Problem, Method, Material, Metric, OtherScientificTerm and Generic (term). The second is the dataset published with SemEval 2017 Shared Task 10 [33] – 3432 sentences from 500 abstracts of computer science, materials sciences, and physics papers with a 2293/371/768 train/dev/test split, named as ScienceIE2017. The raw data was preprocessed by tokenizing using Stanford's Stanza [34] and removing entity annotations which start in the middle of a meaningful word. Both datasets are transformed into the JSON format used by SpERT [7].

The codes are written in PyTorch using HuggingFace's Transformers library [35] by modifying SpERT [7], i.e., by adding the span and context representations and turning off the relation extraction component. Because this paper uses SciBERT, the dimensionalities of $\mathbf{h}$, $\mathbf{c}^l$ and $\mathbf{c}^r$ are all 768. The dimensionality of $\mathbf{s}$ depends on choices of span and context encoding. The length of $\mathbf{w}$ is typically set to 25 (choice of this paper,

not tuned) or 30. To make results comparable, most hyperparameter choices follow the SpERT paper, except that 40 epochs are run for model selection and a larger batch size $B = 16$ is chosen to speed up training. The attention hidden size, i.e., the size of the FFN output in Eq. (1) is set to half of SciBERT hidden size; it is not tuned. As in [22], micro F1 is averaged over five runs, with standard deviation underscripted to save space.

3.2 Results

First, the performances of each individual `SPAN_SUMM`, `SPAN_END` and `END_COMB` option without context representation ($Q1$) are compared. Table 1 summarizes the results. For succinctness, Table 1 is also used to answer $Q2$ (partly), so the CONCAT column is put next to NONE column. The slightly shaded blanks in Table 1 lists the four simplest span representations (c.f. [22]): MAX_POOL, SEL_ATT, CONCAT and DIFF_SUM. In consistency with [22], MAX_POOL is a well-performing single feature for span-only representation and SELF_ATT is the worst. As said before, AVG_POOL and SUM_POOL were also tested at earlier stages, but they systematically performed worse than MAP_POOL, in consistency with results in [7] and [22], so were not reported and further experimented on. CONCAT is also a good feature. DIFF_SUM alone performs reasonably well too but underperforms MAX_POOL and CONCAT. On the contrary, SELF_ATT received a drastic performance drop from the other three on both datasets. Using variants of SELF_ATT does not change the picture. However, much better performance will be achievable when SELF_ATT is combined with other span representations, like CONCAT in Table 1, and context representations (Line 5 and Line 8 in Table 2). It is guessed that (1) SELF_ATT provides a complementary feature to span representation, and (2) Science NER is much more difficult than general domain NER, signaled by the much lower F1 (around 71% F1 by full SpERT on SciERC, and only 46% on ScienceIE2017 [6])[2]. Results from relevant recent studies are also attached for reference. In [21], character-level embeddings were concatenated to SciBERT token embeddings before being sent into the encoder, while other parts were exactly the same as the original BERT NER model [4]. SciIE [6] and SpERT [7] are both methods trained on a multi-tasked setting using relation extraction as supplementary tasks to boost NER performance, which is beyond the scope of this paper.

[2] I also intended to inject entity representation into to a recent idea of "NER as MRC" [24] by additionally encoding entity type or type description into the SciBERT model, but I got astonishingly poor results using their original source codes on science NER datasets. It seems that annotation guidelines of Science NER datasets are not as well-defined and useful for improving NER performance. Most NER datasets annotate only (or mostly) noun phrases. Entities in general domains are easier to identify, like GPE (GeoPolitical Entity), PER (Person) or LOC (Location) etc. They take simpler syntactic roles in sentences. In biomedical NER, entities are often gene and disease names, sometimes chemical compounds. Entity types in science NER are more complex. Science NER annotates Process too, which is usually a verb phrase. Problem can be as simple as a noun phrase but can also be as complex as a short clause; similar for Method. Material for cross-domain dataset ScienceIE2017 can mean a tool, a corpus or a chemical compound etc. In single-domain dataset SciERC, Materials can still mean different things.

Table 2 focuses on the impact of SPAN_END as it is the most widely adopted, and often times the only, feature for the downstream entity classifier. The upward arrow (in red) or downward arrow (in green) on each row indicate performance increase or decrease if SPAN_END is set to CONCAT. Compared to SciIE, DYGIE++ further added coreference resolution into the multi-tasking framework [34]. From Table 2, adding CONCAT to span representation (see the two SPAN_END= CONCAT columns) improves NER performance for most model settings, except CONTEXT=SELF_ATT on SciERC (compared to the two corresponding SPAN_END=NONE columns). Although there is no evidence to say that combing CONCAT with other SPAN_SUMM options universally improves NER performance, it seems that CONCAT is indeed a promising feature, especially when results in Table 1 are also considered. This reasonably explains the fact that almost all span-based models for all kinds of NLP tasks, not only NER, use span endpoint concatenation as their basic feature. Nevertheless, more experiments on other datasets are required to derive a more evidenced conclusive statement on the role of CONCAT.

Now look at the impact of DIFF_SUM on NER performance in Table 3. Often times, adding the difference and sum of the two span endpoints to the feature vector will also result in a performance boost. This is especially true when the span representation is "weak", for example when span endpoints are not used. In the 10 rows with SPAN_END =NONE, significantly better NER performances are achieved on both datasets with END_COMB=DIFF_SUM than the weaker models without considering END_COMB. However, it seems that DIFF_SUM does not go perfectly well with CONCAT: mixed behavior exhibits when the two are combined; no clear trend can be observed from the results on the two datasets. On the other hand, it is observed that, for both datasets, the best performing models (in boldface in Table 3) appear in the END_COMB=DIFF_SUM category. The best micro F1 value 47.13 on ScienceIE2017 is significantly better than existing results, including the best participant result 0.44 from SemEval 2017 shared Task 10. SciBERT NER [21] is a sequential labeling method; this partially justifies that comprehensive span representation, in combination with an appropriate context representation, indeed has significant importance. In fact, most span-only variants also perform rather well (Table 1; Line 1-4 in Table 3). This F1 score is also much better than the BiLSTM based multi-tasking model [6]; this justifies the benefit of in-domain pre-trained language model like SciBERT and the benefit of the pre-training/fine-tuning paradigm that is nowadays prevalent in state-of-the-art NLP solutions.

Similar mixed behavior can be found in the choice of context representation. No single context representation or single combination of span and context representation is universally superior to other choices. The performance also seems to vary with datasets of different nature. However, there is a prominent phenomenon, if allowed to say so, that CONCAT alone does not seem to get along well with CONTEXT, with an exception for [CLS] on ScienceIE2017. Often, MAX_POOL is more agreeable with context representations (Line 5-8 in Table 3). [CLS] is a surprisingly good CONTEXT representation method when combined with CONCAT for SPAN_END and MAX_POOL (and SELF_ATT) for SPAN_SUMM (Line 10 and Line18 in Table 3).

3.3 Summary and Discussions

To summarize, only a few questions in Sect. 2.3 have a relative clear answer. For most questions, the answers, if any, seem to be dependent on the nature of dataset.

For *Q*1: SELF_ATT is the weakest single feature for span representation, while MAX_POOL and CONCAT are two promising features, followed by `END_COMB=` DIFF_SUM alone (Table 1). However, SELF_ATT seems to complement other span encodings quite well (Line 13-16 in Table 3).

For *Q*2: Most time, CONCAT is a promising feature, either as a span-only representation or together with context representations (Table 1-2). However, the behavior of its impact becomes more random when combined with DIFF_SUM (see the two downward arrows in Table 2 and Line 9-12 in Table 3).

For *Q*3: Often times, DIFF_SUM is a promising feature except when CONCAT is combined with MAX_POOL (either for `SPAN_SUMM` or for `CONTEXT`). For the latter, no clear trend can be observed.

For *Q*4: To conclude on `CONTEXT` is the hardest thing. Little regularity can be observed among different combinations of modeling options.

4 Conclusions

This paper investigates the performances of various span and context representation methods for span-based science NER, used either individually or in combination, following the SciBERT fine-tuning paradigm. For span representation methods, some interesting phenomena exist. Span endpoint concatenation and endpoint combination by difference/sum are two promising span-only features for entity representation, which agrees with [22], but self attentive summary performs significantly worse. However, the latter provides complementary information usable to improve performance in in more complex combinations of modeling options. Comprehensive entity representations do improve science NER performance; they outperform SciBERT NER by a large margin. Furthermore, in most cases these two features also add additional NER power when combined with context encodings and other span encodings. This is justified by the fact that the best performing models on both datasets both rely on an appropriate combination of span and entity representations. As to which context representation and which span/context combination are better, there is no clear trend observed. However, it seems that when span endpoints are not used, context representations consistently improve performance. This reasonably explains the reality that span endpoint concatenation the most prevalent feature for entity representation. As a complement to [22] this study conjectures that the best combination of modeling options may also depend on the nature of the dataset. Future work includes experimenting with other span endpoint combinations [6, 30] and span context representations mentioned in Sect. 2.1-2.2 and demonstrating the effectiveness of stronger entity representation in state-of-the-art NER approaches [9, 23], including the one discussed in footnote 2 [24].

Table 1. Performances of Individual Span Representation Methods and the Impact of SPAN_END.

SPAN_SUMM	CONTEXT	END_COMB	**SciERC:** SPAN END (micro F1 dev/test)			**ScienceIE2017:** SPAN END (micro F1 dev/test)		
			NONE	CONCAT		NONE	CONCAT	
MAX_POOL	NONE	NONE	$71.295_{1.209}$ / $68.954_{0.568}$	$71.381_{0.627}$ / $69.099_{1.269}$	↑	$57.713_{1.209}$ / $45.118_{0.568}$	$57.667_{0.501}$ / **46.194**$_{0.902}$	↑
SELF_ATT	NONE	NONE	$62.177_{0.615}$ / $60.613_{0.561}$	$71.212_{0.352}$ / $68.552_{1.062}$	↑	$49.810_{0.508}$ / $39.625_{0.681}$	$71.212_{0.352}$ / $46.585_{0.353}$	↑
NONE	NONE	NONE	N/A	$70.757_{0.853}$ / $68.458_{0.779}$		N/A	$55.890_{0.648}$ / $45.833_{0.897}$	
NONE	NONE	NONE	$71.245_{0.593}$ / $68.186_{1.025}$	--		$56.358_{0.576}$ / $45.711_{0.440}$	--	
[21]: BERT NER + char embed (SciBERT)			-----$_{??}$ / $65.6_{1.0}$			-----$_{??}$ / $43.8_{1.0}$		
[6]: SciIE (BiLSTM + Glove; +RE/COREF)			$68.1_{??}$ / $64.2_{??}$			-----$_{??}$ / $46.0_{??}$		
[7]: SpERT (SciBERT; +RE)			-----$_{??}$ / 70.33			N/A		

Note: The shaded blanks are four simplest span representations in [22]. It is unclear whether the average results or the best run were reported in [6, 21, 34]. "-----" and underscripted "??" indicate unreported figure and standard deviation respectively.

Table 2. Performances of Combinations of Span and Context Representations and the Impact of SPAN_END.

	SPAN_SUMM	CONTEXT	**SciERC:** SPAN_END (micro F1 dev/test)			**ScienceIE2017:** SPAN_END (micro F1 dev/test)		
			NONE	CONCAT		NONE	CONCAT	
1	MAX_POOL	[CLS]	$71.266_{0.412}$ / $68.661_{0.859}$	$71.516_{0.707}$ / **69.269**$_{0.880}$	↑	$58.040_{0.701}$ / $46.206_{0.504}$	$57.753_{0.656}$ / $46.007_{0.845}$	↓
2	SELF_ATT	[CLS]	$62.426_{0.789}$ / $60.942_{0.778}$	$71.876_{0.618}$ / $69.118_{0.852}$	↑	$49.996_{0.703}$ / $40.415_{0.709}$	$57.844_{0.550}$ / $46.113_{1.129}$	↑
3	NONE	[CLS]	N/A	$70.976_{0.727}$ / $67.609_{0.562}$		N/A	$56.008_{0.837}$ / $45.322_{1.009}$	
4	MAX_POOL	SELF_ATT	$71.363_{0.606}$ / $\underline{68.935}_{1.254}$	$71.291_{0.651}$ / $68.687_{0.864}$	↓	$58.387_{0.510}$ / $45.636_{0.478}$	$58.152_{1.028}$ / $46.423_{0.666}$	↑
5	SELF_ATT	SELF_ATT	$70.979_{0.502}$ / $69.109_{0.833}$	$72.180_{0.853}$ / $68.729_{1.239}$	↓	$57.312_{0.468}$ / $45.582_{0.902}$	$57.454_{0.877}$ / $45.992_{0.653}$	↑
6	NONE	SELF_ATT	N/A	$71.051_{0.673}$ / $68.505_{0.474}$		N/A	$56.134_{0.988}$ / $45.373_{0.749}$	
7	MAX_POOL	MAX_POOL	$71.383_{0.592}$ / $68.778_{0.495}$	$71.784_{0.727}$ / $69.032_{0.747}$	↑	$57.817_{0.576}$ / $45.784_{0.646}$	$58.131_{0.515}$ / $46.400_{0.688}$	↑
8	SELF_ATT	MAX_POOL	$71.058_{0.871}$ / $68.564_{0.849}$	$71.008_{0.579}$ / $68.976_{0.612}$	↑	$56.856_{0.124}$ / $45.230_{0.753}$	$57.758_{0.806}$ / **46.748**$_{0.822}$	↑
9	NONE	MAX_POOL	N/A	$71.644_{0.701}$ / $68.452_{0.435}$		N/A	$56.143_{0.832}$ / $45.503_{0.750}$	
[21]: BERT NER + char embed (SciBERT)			-----$_{??}$ / $65.6_{1.0}$			-----$_{??}$ / $43.8_{1.0}$		
[6]: SciIE (BiLSTM + Glove; +RE/COREF)			$68.1_{??}$ / $64.2_{??}$			-----$_{??}$ / $46.0_{??}$		
[36]: DYGIE++ (BERT; + RE/COREF)			-----$_{??}$ / 67.5			N/A		
[7]: SpERT (SciBERT; +RE)			-----$_{??}$ / 70.33			N/A		

Note: the setting of SPAN_SUMM=MAX_POOL, CONTEX=[CLS] and SPAN_END=NONE replicates an NER-only version of SpERT.

Table 3. Performances of Combinations of Span and Context Representations and the Impact of END_COMB.

	SPAN_END	SPAN_SUMM	CONTEXT	**SciERC:** END_COMB (micro F1 dev/test)			**ScienceIE2017:** END_COMB (micro F1 dev/test)		
				NONE	DIFF_SUM		NONE	DIF_SUM	
1	CONCAT	NONE	NONE	$70.757_{0.853}$ / $68.458_{0.779}$	$70.816_{0.519}$ / $68.202_{0.646}$	↓	$55.890_{0.508}$ / $45.833_{0.681}$	$55.870_{0.352}$ / $45.318_{0.353}$	↓
2	CONCAT	NONE	[CLS]	$70.976_{0.727}$ / $67.609_{0.562}$	$71.331_{0.635}$ / $68.094_{0.577}$	↑	$56.008_{0.837}$ / $45.322_{1.009}$	$56.626_{0.503}$ / $46.624_{0.160}$	↑
3	CONCAT	NONE	SELF_ATT	$71.051_{0.673}$ / $68.505_{0.474}$	$71.179_{0.388}$ / $68.480_{0.933}$	↓	$56.134_{0.988}$ / $45.373_{0.749}$	$56.714_{0.576}$ / $45.852_{0.276}$	↑
4	CONCAT	NONE	MAX_POOL	$71.644_{0.701}$ / $68.452_{0.435}$	$71.422_{0.482}$ / $69.038_{0.465}$	↑	$56.143_{0.832}$ / $45.503_{0.750}$	$56.465_{0.367}$ / $45.976_{0.622}$	↑
5	NONE	MAX_POOL	NONE	$71.295_{1.209}$ / $68.954_{0.568}$	$71.720_{0.522}$ / $69.085_{0.971}$	↑	$57.713_{1.209}$ / $45.118_{0.568}$	$57.556_{0.501}$ / $45.984_{0.902}$	↑
6	NONE	MAX_POOL	[CLS]	$71.266_{0.412}$ / $68.661_{0.859}$*	$71.846_{0.865}$ / $68.952_{0.355}$	↑	$58.040_{0.701}$ / $46.206_{0.504}$	$57.889_{0.656}$ / $46.305_{0.845}$	↑
7	NONE	MAX_POOL	SELF_ATT	$71.363_{0.606}$ / $68.935_{1.254}$	$71.408_{0.679}$ / $69.002_{0.714}$	↑	$58.387_{0.510}$ / $45.636_{0.478}$	$58.149_{0.923}$ / $46.740_{0.191}$	↑
8	NONE	MAX_POOL	MAX_POOL	$71.383_{0.592}$ / $68.778_{0.495}$	$71.456_{0.669}$ / $69.533_{0.945}$	↑	$57.817_{0.576}$ / $45.784_{0.646}$	$57.786_{0.865}$ / $46.183_{0.522}$	↑
9	CONCAT	MAX_POOL	NONE	$71.381_{0.627}$ / $69.099_{1.269}$	$72.192_{0.767}$ / $69.413_{0.351}$	↑	$57.667_{0.501}$ / $46.194_{0.902}$	$58.042_{0.620}$ / $45.978_{0.816}$	↓
10	CONCAT	MAX_POOL	[CLS]	$71.516_{0.707}$ / **$69.269_{0.880}$**	$71.992_{0.739}$ / $69.266_{0.833}$	↓	$57.753_{0.656}$ / $46.007_{0.845}$	$57.879_{1.187}$ / $46.934_{1.737}$	↑
11	CONCAT	MAX_POOL	SELF_ATT	$71.291_{0.651}$ / $68.687_{0.864}$	$71.873_{0.749}$ / $68.648_{0.700}$	↓	$58.152_{1.028}$ / $46.423_{0.666}$	$58.315_{0.378}$ / $46.641_{0.686}$	↑
12	CONCAT	MAX_POOL	MAX_POOL	$71.784_{0.727}$ / $69.032_{0.747}$	$71.768_{0.418}$ / $68.581_{1.119}$	↓	$58.131_{0.515}$ / $46.400_{0.688}$	$57.870_{0.529}$ / $45.945_{0.535}$	↓
13	NONE	SELF_ATT	NONE	$62.177_{0.615}$ / $60.613_{0.561}$	$71.475_{0.831}$ / $68.875_{1.090}$	↑	$49.810_{0.508}$ / $39.625_{0.681}$	$58.401_{0.501}$ / $46.358_{0.488}$	↑
14	NONE	SELF_ATT	[CLS]	$62.426_{0.789}$ / $60.942_{0.778}$	$71.813_{0.369}$ / $69.202_{0.528}$	↑	$49.996_{0.703}$ / $40.415_{0.709}$	$57.846_{0.616}$ / $46.177_{1.032}$	↑
15	NONE	SELF_ATT	SELF_ATT	$70.979_{0.502}$ / $69.109_{0.833}$	$71.946_{0.819}$ / $69.385_{0.681}$	↑	$57.312_{0.468}$ / $45.582_{0.902}$	$58.117_{0.605}$ / $46.639_{0.681}$	↑
16	NONE	SELF_ATT	MAX_POOL	$71.058_{0.871}$ / $68.564_{0.849}$	$71.470_{0.754}$ / $69.086_{0.396}$	↑	$56.856_{0.124}$ / $45.230_{0.753}$	$58.358_{0.576}$ / **$47.134_{0.916}$**	↑
17	CONCAT	SELF_ATT	NONE	$71.212_{0.668}$ / $68.522_{0.328}$	$71.209_{0.490}$ / $68.873_{0.811}$	↑	$58.432_{0.386}$ / $46.585_{0.353}$	$58.119_{0.841}$ / $46.602_{0.923}$	↑
18	CONCAT	SELF_ATT	[CLS]	$71.876_{0.618}$ / $69.118_{0.852}$	$71.621_{0.659}$ / **$69.780_{0.645}$**	↑	$57.844_{0.550}$ / $46.113_{1.129}$	$57.812_{0.831}$ / $46.071_{1.090}$	↓
19	CONCAT	SELF_ATT	SELF_ATT	$72.180_{0.853}$ / $68.729_{1.239}$	$72.109_{1.175}$ / $69.045_{1.462}$	↑	$57.454_{0.877}$ / $45.992_{0.653}$	$58.376_{0.542}$ / $46.434_{0.421}$	↑
20	CONCAT	SELF_ATT	MAX_POOL	$71.008_{0.579}$ / $68.976_{0.612}$	$71.799_{0.503}$ / $68.673_{1.192}$	↓	$57.758_{0.806}$ / **$46.748_{0.822}$**	$58.202_{0.428}$ / $46.626_{0.673}$	↓
[21]: BERT NER + char embed (SciBERT)				-----?? / $65.6_{1.0}$			-----?? / $43.8_{1.0}$		
[6]: SciIE (BiLSTM + Glove; +RE/COREF)				$68.1_{??}$ / $64.2_{??}$			-----?? / $46.0_{??}$		
[36]: DYGIE++ (BERT; + RE/COREF)				-----?? / 67.5			N/A		
[7]: SpERT (SciBERT; +RE)				-----?? / 70.33			N/A		

* This is the NER-only version of SpERT.

References

1. Augenstein, I., Søgaard, A.: Multi-Task Learning of Keyphrase Boundary Classification. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 341–346. Association for Computational Linguistics, Stroudsburg, PA, USA (2017).
2. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K, Dyer, C.: Neural Architectures for Named Entity Recognition. In: Proceedings of the 2016 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 260–270. Association for Computational Linguistics, Stroudsburg, PA, USA (2016).
3. Luan, Y., Ostendorf, M., Hajishirzi, H.: Scientific Information Extraction with Semi-supervised Neural Tagging. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2641–2651. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
4. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Stroudsburg, PA, USA (2019).
5. Sohrab, M. G., Miwa, M.: Deep Exhaustive Model for Nested Named Entity Recognition. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2843–2849. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
6. Luan, Y., He, L., Ostendorf, M., Hajishirzi, H.: Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3219–3232. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
7. Eberts, M., Ulges, A.: Span-based Joint Entity and Relation Extraction with Transformer Pre-training. In: 24th European Conference on Artificial Intelligence - ECAI 2020. European Association for Artificial Intelligence (2020).
8. Xia, C., Zhang, C., Yang, T., Li, Y., Du, N., Wu, X., Fan, W., Ma, F., Philip, Y.: Multi-grained Named Entity Recognition. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1430–1440. Association for Computational Linguistics, Stroudsburg, PA, USA (2019).
9. Tan, C., Qiu, W., Chen, M., Wang, R., Huang, F.: Boundary Enhanced Neural Span Classification for Nested Named Entity Recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, 34(05), 9016–9023. Association for the Advancement of Artificial Intelligence, Palo Alto, California, USA (2020).
10. Ouchi, H., Shido, H., Matsumoto, Y.: A Span Selection Model for Semantic Role Labeling. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 1630–1642. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
11. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P: SQuAD: 100,000+ Questions for Machine Comprehension of Text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2383–2392. Association for Computational Linguistics, Stroudsburg, PA, USA (2016).

12. Rajpurkar, P., Robn, J., Liang, P.: Know What You Don't Know: Unanswerable Questions for SQuAD. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 784–789. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
13. Lee, K., He, L., Lewis, M., Zettlemoyer, L.: End-to-End Neural Coreference Resolution. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 188–197. Association for Computational Linguistics, Stroudsburg, PA, USA (2017).
14. Guo, J., Che, W. Liu, T., Xu, J.: A Unified Architecture for Semantic Role Labeling and Relation Classification. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 1264–1274. Association for Computational Linguistics, Stroudsburg, PA, USA (2016).
15. Swayamdipta, S., Thomson, S., Lee, K., Zettlemoyer, L., Dyer, C., Smith N. A.: Syntactic Scaffolds for Semantic Structures. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3772–3782. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
16. Zhao, H.., Huang, L., Zhang, R., Lu, Q., Xue, H.: SpanMlt: A Span-based Multi-Task Learning Framework for Pair-wise Aspect and Opinion Terms Extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 3239–3248. Association for Computational Linguistics, Stroudsburg, PA, USA (2020).
17. Jiang, Z., Xu, W., Araki, J., Neubig, G.: Generalizing Natural Language Analysis through Span-relation Representations. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 2120–2133. Association for Computational Linguistics, Stroudsburg, PA, USA (2020).
18. Taillé, B., Guigue, V., Gallinari, P.: Contextualized Embeddings in Named-Entity Recognition: An Empirical Study on Generalization. Jose J. et al. (eds) Advances in Information Retrieval. ECIR 2020. Lecture Notes in Computer Science, vol 12035, pp. 383–391. Springer, Heidelberg (2020).
19. Akbik, A., Blythe, D., Vollgraf, R.: Contextual String Embeddings for Sequence Labeling. In: Proceedings of the 27th International Conference on Computational Linguistics, pp. 341–346. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
20. Beltagy, I., Lo, K., Cohan, A.: SciBERT: A Pretrained Language Model for Scientific Text. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3615–3620. Association for Computational Linguistics, Stroudsburg, PA, USA (2019).
21. Brack, A., D'Souza, J., Hoppe, A., Auer, S., Ewerth, R.: Domain-Independent Extraction of Scientific Concepts from Research Articles. Jose J. et al. (eds) Advances in Information Retrieval. ECIR 2020. Lecture Notes in Computer Science, vol 12035, pp. 251–266. Springer, Heidelberg (2020).
22. Toshniwal, S., Shi, H., Shi, B., Gao, L., Livescu, K., Gimpel, K.: A Cross-Task Analysis of Text Span Representations. In: Proceedings of the 5th Workshop on Representation Learning for NLP, pp. 166–176. Association for Computational Linguistics, Stroudsburg, PA, USA (2020)
23. Hu, Z., Yin, H., Xu, G., Zhai, Y., Pan, D., Liang, Y.: An Empirical Study on Joint Entities-relations Extraction of Chinese Text Based on BERT. In: Proceedings of the 2020 12th International Conference on Machine Learning and Computing, pp. 473 478. ACM, New York, NY, USA (2020).

24. Li, X., Feng, J., Meng, Y., Han, Q., Wu, F., Li, J.: A Unified MRC Framework for Named Entity Recognition. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5849–5859. Association for Computational Linguistics, Stroudsburg, PA, USA (2020).
25. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research, 12. Pp. 2493–2537 (2011).
26. Hashimoto, K., Xiong,C., Tsuruoka, Y., Socher, R.: A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1923–1933. Association for Computational Linguistics, Stroudsburg, PA, USA (2017).
27. Lin, Z., Feng, M., Xu, M., Xiang, B., Zhou, B., Bengio, Y.: A Structured Self-attentive Sentence Embedding. In: Proceedings of the 5th International Conference on Learning Representations. Available at: https://openreview.net/references/pdf?id=SyLIO1tte.
28. Ouchi, H., Shindo, H., Matsumoto, Y.: A Span Selection Model for Semantic Role Labeling. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 1630–1642. Association for Computational Linguistics, Stroudsburg, PA, USA (2018).
29. Zhan, J., Zhao, H.: Span Model for Open Information Extraction on Accurate Corpus. In: Proceedings of the AAAI Conference on Artificial Intelligence, 34(05), pp. 9523–9530. Available at: https://doi.org/10.1609/aaai.v34i05.6497.
30. Seo, M., Lee, J., Kwiatkowski,T., Parikh, A., Farhadi, A., Hajishirzi, H.: Real-Time Open-Domain Question Answering with Dense-Sparse Phrase Index. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4430–4441. Association for Computational Linguistics, Stroudsburg, PA, USA (2019).
31. Liu, S., Sun, Y., Li, B., Wang, W., Zhao, X.: HAMNER: Headword Amplified Multi-Span Distantly Supervised Method for Domain Specific Named Entity Recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, 34(05), pp. 8401–8408. Available at: https://doi.org/10.1609/aaai.v34i05.6358.
32. Zhou, Y., Huang, L., Guo, T., Han, J., Hu, S.: A Span-based Joint Model for Opinion Target Extraction and Target Sentiment Classification. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19), pp. 5485–5491. Doi: https://doi.org/10.24963/ijcai.2019/762.
33. Augenstein, I., Das, M., Riedal, S., Vikraman, L., McCallum, A.: SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp. 546–555. Association for Computational Linguistics, Stroudsburg, PA, USA (2017).
34. Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 101–108. Association for Computational Linguistics, Stroudsburg, PA, USA (2020).
35. Wolf, T., Debut, L., Sahn, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., Rush, A.: Transformers: State-of-the-Art Natural Language Processing. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45. Association for Computational Linguistics, Stroudsburg, PA, USA (2020).

36. Wadden, D., Wennberg, U., Luan, Y., Hajishirzi, H.: Entity, Relation, and Event Extraction with Contextualized Span Representations. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 5784–5789. Association for Computational Linguistics, Stroudsburg, PA, USA (2019).