Less is More: A Prototypical Framework for Efficient Few-Shot Named Entity Recognition

Yue Zhang[1,2(✉)] and Hui Fang[1,2]

[1] Department of Electrical and Computer Engineering, University of Delaware, Newark, USA
{zhangyue,hfang}@udel.edu
[2] Center for Plastics Innovation, University of Delaware, Newark, USA

Abstract. Few-shot named entity recognition (NER) aims to leverage a small number of labeled examples to extract novel-class named entities from unstructured text. Although existing few-shot NER methods, such as ESD and DecomposedMetaNER, are effective, they are quite complex and not efficient, which makes them unsuitable for real-world applications when the prediction time is a critical factor. In this paper, we propose a simple span-based prototypical framework that follows the metric-based meta-learning paradigm and does not require time-consuming fine-tuning. In addition, the BERT encoding process in our model can be pre-computed and cached, making the final inference process even faster. Experiment results show that, compared with the state-of-the-art models, the proposed framework can achieve comparable effectiveness with much better efficiency.

Keywords: Few-shot NER · Metric-learning · Efficiency

1 Introduction

Named Entity Recognition (NER) is an important natural language understanding task, which aims to extract certain types of named entities (e.g., locations) from unstructured text. Most neural NER models follow the supervised learning paradigm and require a large amount of annotated data for training. These models have impressive performance in extracting existing entity types. However, in practice, we want a NER model to rapidly adapt to novel entity types so that we can test the prototype NER systems and get feedback for future improvement. As a result, few-shot NER [5,7], which can learn to extract novel types of entities based on a few training examples for each class, has gotten a lot of attention recently.

The focus of existing few-shot NER research is mainly on optimizing accuracy over very little training data. However, for real applications, improving accuracy will not help if the benefits it brings become outweighed by the inconvenience caused by increasing prediction latency. As it is well known that the more training data you have, the better performance you get, using few-shot models may

E. Métais et al. (Eds.): NLDB 2023, LNCS 13913, pp. 33–46, 2023.
https://doi.org/10.1007/978-3-031-35320-8_3

become less attractive if the time taken to generate predictions is too long. In such cases, people may just annotate more data and use a traditional supervised NER instead. For example, in scientific literature mining, researchers often want to extract entities from thousands of research articles to compose a knowledge base and mine patterns [13,29]. However, based on our preliminary results, existing few-shot NER methods can only finish predicting several articles per second, making the entire waiting time uncomfortably long. In this case, domain experts may give up using few-shot models as a prototype development tool. Therefore for practical applications, we need to focus on optimizing not only accuracy but also prediction latency.

The state-of-the-art few-shot NER methods often utilize pre-trained language models (e.g. BERT [4]) to get the prior knowledge of human language and train their model on existing entities to learn NER specific prior knowledge. DecomposedMetaNER [17] and CONTaiNER [3], additionally followed the transfer learning paradigm and fine-tuned their models on a few examples of novel entities for better adaptation. However, this fine-tuning process forces the whole prediction process to be carried out online since the model's parameters may change as the user provides different examples. As a result, the state-of-the-art few-shot NER models are not very efficient, making them impractical to be used in applications requiring shorter prediction time.

On the contrary, metric-based meta-learning models can accelerate the prediction process through pre-computing and caching part of the model's computation result. These models project samples into an embedding space, where similar and dissimilar samples can be easily discriminated by non-parametric methods. For example, for each class, prototypical network [20] takes the mean vector of all embedded same class examples as the prototype representation for that class. It then classifies each test example to the class of its nearest prototype based on Euclidean distance. In this case, the similarity part needs to be computed online, but the time-consuming encoding part can be done offline. Clearly, this approach seems to shed a light towards efficient few-shot NER methods.

In this work, we propose a **S**imple **S**pan-base **P**rototypical framework (SSP) for few-shot NER. SSP follows the metric-based meta-learning paradigm and does not require fine-tuning. Specifically, our model leverages Layer Normalization [1] to transform the span representation, which ensures the representation of nice geometric properties and is beneficial for the subsequent non-parametric classification process. We also incorporate the attention mechanism into the metric-based meta-learning process to dynamically focus on closely related examples and better handle outliers in the labeled examples. Additionally, we pre-train SSP on supervised NER tasks before training it on meta-learning tasks in order to enhance its span representation capability. The simple structure of our model, combined with pre-computing the BERT representation, makes our inference speed significantly faster than existing models. The contributions of our work can be summarized as follows: (1) We conduct a comparative study to compare the efficiency of existing state-of-the-art (SoTA) methods and find their prediction efficiency is not satisfying. (2) We propose a lightweight few-shot NER model, which is as accurate as previous SoTA models but much more efficient.

2 Related Work

Meta-learning has become a popular method for tackling the few-shot learning problem. Especially, metric-based meta-learning methods have become the mainstream methods for few-shot image classification [20–22] and have also been widely adapted to other NLP tasks such as relation extraction [9]. Our model follows the same nearest class prototype classification idea and generalizes the mean pooling to attention pooling. Compared with ESD [23], which uses four types of attention, our model has a much simpler attention module and is equally effective.

Early few-shot NER methods are typically based on token-level classification [7,12]. Such token-level models tend to make single-token errors, where one of the words inside a multi-word entity is misclassified due to the lack of training examples. StructShot [27], CONTaiNER [3] and L-TapNet+CDT [11] added a label-agnostic conditional random field (CRF) for modeling label dependency between tokens. But a lot of valuable similarity information between tokens has already been lost before CRF because the CRF only takes a single scalar to represent the label distribution. In practice, the transition matrix needs to be learned from existing entities, making it prone to the domain-shift problem if the span length distribution changes greatly. Alternatively, Proto+Reptile [15] used a neural network to predict the transition probability. Recently, span-based few-shot NER methods, such as ESD [23], DecomposedMetaNER [17], and SpanNER [25], have been proposed to explicitly model phrasal representation and ensure the integrity of the phrase. Our model also uses span embedding to model phrases directly.

Moreover, prompt-learning [2] and language modeling [16] were also adapted to solve the few-shot NER problem. However, they needed either a large validation set to choose the template [2] or external lexical annotation to select label word [16]. Similarly, another work [12] uses noisy-supervised pre-training for few-shot NER, but an additional large-scale noisy NER dataset is not always available. So these works does not align well with the meta-learning set up of our work. So we do not compare our model against these methods and leave the exploration of using external resources for future work.

3 Task Formulation

NER aims to identify each named entity correctly and classify it into the corresponding categories. We follow the idea of previous meta-learning studies [5,11] and formalize the few-shot NER problem as solving a list of N-way K-shot NER tasks.

Each task, namely the episode, mimics a real-world few-shot learning challenge. The set of all known examples in an episode is denoted as the **support set S**, and the set of all unknown examples is denoted as the **query set Q**. **N** denotes the number of entity types in the task, and **K** denotes the number of annotated examples for each entity type in the support set. For each task, the

model encounters a set of novel entities and needs to extract entities in Q only using the examples in S. Figure 1 shows an example of a 2-way 2-shot task.

This process needs to be repeated multiple times for the model to learn how to adapt to a new task quickly with a few examples. Therefore, in practice, we randomly sample the annotated sentences to construct episodes according to the N-way K-shot constraints. And the testing set comes from a domain that is different from the training set and has non-intersecting label spaces in order to truly test the model's capability of generalizing to unseen types of entities.

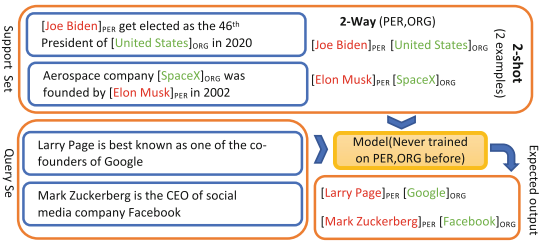

Fig. 1. A 2-way 2-shot task for Person(PER) and Organization(ORG) entity.

4 Simple Span-Based Prototypical (SSP) Framework

We propose a simple span-based prototypical (SSP) framework to tackle the few-shot NER problem. The proposed model consists of four parts: (1) span representation and normalization, where we model and normalize the span embedding; (2) metric based meta-learning with attention, where we generate class centers using an attention mechanism and classify spans accordingly; (3) whole classification pre-training, where we pre-train our model on supervised NER tasks before meta-learning; and (4) post-processing, where we resolve overlapping conflicts in span predictions.

Span Representation and Normalization. Our method formulates the few-shot NER as a span classification problem instead of a token classification problem since token-based models tend to recognize only part of multi-token entities and produce many false positive predictions. For every possible span (i.e. continuous piece of text) in the sentence, our model classifies it into either one of the predefined categories or the special category **O** (i.e. ordinary spans). For instance, given the sentence "Aerospace company SpaceX was founded by Elon Musk in 2002", "SpaceX" and "Elon Musk" will be classified as Person and Organization span while "founded" and "in 2002" will be identified as type O spans. Spans that are partially overlapped with ground truth, like "Aerospace company SpaceX", are not treated as correct spans during evaluation and will be used to construct the prototype for type O spans.

We take BERT [4] as our backbone sentence encoder and follow the standard head-tail concatenation way to compose the span embedding [8,14,18,23]. The last output layer of BERT is used as the contextual representation for tokens in the sentence. We take the first subword of the span's starting and ending words to form its initial span representation $[\mathbf{h_{start}}, \mathbf{h_{end}}]$. Here $\mathbf{h}$ denotes the BERT encoder's output for each subword. To incorporate the length information into the span representation, following the same methodology proposed in the previous study [8], we add word length embedding $\mathbf{WL}$ and subword length embedding $\mathbf{TL}$ to the initial span representation.

After that, the span representation is projected into the desired dimension with linear transformation and normalized using **Layer Normalization** (LN) [1]

$$\widetilde{\mathbf{s}} = \mathrm{LN}\left(\mathbf{W}\left([\mathbf{h_{start}}, \mathbf{h_{end}}] + \mathbf{WL} + \mathbf{TL}\right)\right) \in \mathbb{R}^D \tag{1}$$

Here $\widetilde{\mathbf{s}}$ denotes the output of our span embedding module, D denotes the span embedding size, and it is set to 768 in our model. Layer Normalization aims to first zero-center and standardize the embedding, and then re-scale and re-center the embedding with parameter γ *and* $\beta \in \mathbb{R}^D$.

$$LN(\widetilde{\mathbf{s}}) = \left(\frac{\widetilde{\mathbf{s}} - mean(\widetilde{\mathbf{s}})}{\sqrt{var(\widetilde{\mathbf{s}})}}\right) * \gamma + \beta \tag{2}$$

An inspection of the trained weights reveals that $\gamma \approx \mathbf{1}, \beta \approx \mathbf{0}$. Thus, the mean of each normalized embedding is around $\mathbf{0}$ and the L2 norm is around $\sqrt{D}$. The Euclidean distance metric that we use to calculate embedding similarity could then be further deduced to

$$|\widetilde{\mathbf{s}}_\mathbf{1} - \widetilde{\mathbf{s}}_\mathbf{2}|_2^2 = |\widetilde{\mathbf{s}}_\mathbf{1}|_2^2 + |\widetilde{\mathbf{s}}_\mathbf{1}|_2^2 - 2\widetilde{\mathbf{s}}_\mathbf{1} \cdot \widetilde{\mathbf{s}}_\mathbf{2} = 2D - 2D\cos\left(\widetilde{\mathbf{s}}_\mathbf{1}, \widetilde{\mathbf{s}}_\mathbf{2}\right) \propto \cos\left(\widetilde{\mathbf{s}}_\mathbf{1}, \widetilde{\mathbf{s}}_\mathbf{2}\right), \tag{3}$$

which is proportional to a cosine distance with a $2D$ scale factor. In this way, we can see that our span normalization method is a special case for the widely adapted scaled cosine similarity. This also explains why we need to use a higher-than-usual(for supervised span-based NER, a typical value is 256 [14] or 150 [28]) embedding size as a theoretical analysis shows increasing size help reduce false positive rate [19] for cosine distance based representation learning. Our experiment analysis later shows both normalization and embedding size plays an important role in the model's good performance.

Metric-Based Meta-Learning with Attention. Given an N-way K-shot task $(\mathbf{S}, \mathbf{Q})$ with entity label space $\mathbf{Y}$, our model first enumerates every possible span in each sentence of $\mathbf{S}$ and $\mathbf{Q}$ and encodes them into corresponding span representation collection $\mathbf{R_S}$ and $\mathbf{R_Q}$. Similar to prototypical network [20], our model predicts the span label y_m based on the distance between the span representation $\widetilde{\mathbf{s}}_\mathbf{m}$ and each class center $\widetilde{\mathbf{c}}_\mathbf{k}$. More specifically, we assign y_m to the label of class center $\widetilde{\mathbf{c}}_\mathbf{k}$, in which it is nearest to $\widetilde{\mathbf{s}}_\mathbf{m}$ based on the squared Euclidean distance metric.

$$y_m^* = \underset{k \in \mathbf{Y} \cup \{O\}}{argmin} \, |\widetilde{\mathbf{s}}_\mathbf{m} - \widetilde{\mathbf{c}}_\mathbf{k}|_2^2 \tag{4}$$

We denote the set of support span representations having label k as $\mathbf{Z_k} = \{\widetilde{\mathbf{s}}_\mathbf{n} : \widetilde{\mathbf{s}}_\mathbf{n} \in \mathbf{R_S}\ and\ y_n = k\}$. The original prototypical network computes the mean pooling of $\mathbf{Z_k}$ as the class center $\widetilde{\mathbf{c}}_\mathbf{k}$, which can be viewed as a special case of attention mechanism where the weight is fixed to $\mathbf{1}$. Like ESD [23] and HATT [9], we use **query-support attention** (QSA) to let the class center representation bias towards similar support span representations and be more robust to outliers.

$$\widetilde{\mathbf{c}}_\mathbf{k}^\mathbf{m} = \sum_i \frac{e^{\widetilde{\mathbf{s}}_\mathbf{m} \cdot \mathbf{Z}_\mathbf{k}^\mathbf{i}}}{\sum_i e^{\widetilde{\mathbf{s}}_\mathbf{m} \cdot \mathbf{Z}_\mathbf{k}^\mathbf{i}}} \mathbf{Z}_\mathbf{k}^\mathbf{i} \tag{5}$$

Here $\cdot$ denotes the dot product operation, $\mathbf{Z}_\mathbf{k}^\mathbf{i}$ is the i_{th} member of $\mathbf{Z_k}$, $\widetilde{\mathbf{c}}_\mathbf{k}^\mathbf{m}$ is the type k class center for query span $\widetilde{\mathbf{s}}_\mathbf{m}$. Figure 2 summarizes the entire span representation and meta-learning process.

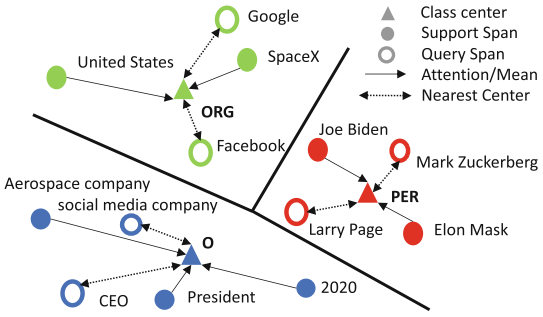

Fig. 2. The span-based prototypical meta-learning framework

The model parameters are updated using gradient descent, and the negative log-likelihood is minimized over a batch of randomly sampled N-way K-shot tasks:

$$-\log \sum_{(\mathbf{S},\mathbf{Q})} \sum_{(\widetilde{\mathbf{s}}_\mathbf{m}, y_m) \in \mathbf{Q}} \frac{e^{|\widetilde{\mathbf{s}}_\mathbf{m} - \widetilde{\mathbf{c}}_\mathbf{k}^\mathbf{m}|_2^2}}{\sum_{k \in \mathbf{Y} \cup \{O\}} e^{|\widetilde{\mathbf{s}}_\mathbf{m} - \widetilde{\mathbf{c}}_\mathbf{k}|_2^2}}. \tag{6}$$

Whole Classification Pre-training. Instead of training our span encoder directly on N-way K-shot episodes, we find pre-training the encoder on the same training set in a supervised NER manner helps the model learn better span representation. Compared with training on N-way K-shot tasks, where the model only optimizes loss on a subset of entity types each time, whole classification pre-training optimizes for the whole label space.

In classification pre-training, the class center $\widetilde{\mathbf{c}}_\mathbf{k}$ is not generated based on the support set but becomes a learnable parameter. We also use the standard dot product instead of squared Euclidean distance as our distance metric. And the loss function is also computed on the sentence level instead of the task level.

Post-processing. The raw output of our model cannot be used directly because there might be overlapping conflicts inside span predictions. For example, two overlapped spans might be both predicted as Non-*O* entities, which is not allowed for flat NER. We use a greedy pruning algorithm, which adapts the Non-maximum Suppression algorithm for NER scenario [28], to decode a set of valid span predictions from the raw output. The algorithm sorts all Non-*O* span predictions based on their confidence scores and then iterates over them, keeping the highest-scoring span and suppressing any overlapping span with a lower score.

5 Experiments

5.1 Experiment Setup

Datasets: *Few-NERD* [5] is a large-scale few-shot NER dataset with a hierarchy of 8 coarse-grained entity types (e.g., Person, Organization) and 66 fine-grained entity types (e.g., Person-Politician, Person-Actor, Organization-Company). There are two different train/valid/test data splitting settings: (1) *Few-NERD Intra*: data are split based on the coarse-grained label of entities (e.g., the train split has all Person entities while the test split has all Organization entities) (2) *Few-NERD Inter*: data are split based on the fine-grained label of entities and the hierarchy relationship is ignored. (e.g. the train split has all Person-Politician entities while the test split has all Person-Actor, this is not allowed in *Few-NERD Intra* because of sharing the same coarse-grained label "Person").

As there is no other few-shot NER dataset available, we pick *SNIPS* as our second benchmark to evaluate if our methods can be generalized to other structure prediction tasks. *SNIPS* is an intent detection and slot-filling dataset with crowdsourced queries expressing seven kinds of user intents. The slot-filling task and NER task both need to make structure predictions on text, but slot-filling is more specific to dialog circumstances. We use the few-shot slot-filling dataset sampled by Hou [11]. It is constructed using the "leaving-one-out" cross-validation strategy. Each time one user intent is used for sampling testing episodes, another intent for validation, and the remaining five intents for training.

Implementation Details: We use *bert-base-cased* as our backbone encoder because some entities, like Music and Movie, would be hard to identify if converted to lowercase. But we also have a variant using *bert-base-uncased* since our baselines use uncased BERT. The *Few-NERD Inter/Intra* training set each has 36/35 fine-grained entity types, which is much bigger than the number of entity types in sampled 5-way/10-way episodes, meaning only a subset of entity types is used in each training step. Therefore, we apply the whole class pre-training for this dataset. In *SNIPS*, we only use uncased BERT since all its sentences are in lowercase. We also do not use whole class pre-training for *SNIPS* since the

episodes sampled from each domain are already constructed with their entire label space. We make our codes public at https://github.com/nsndimt/SSP.

Baselines: We compare our model against three state-of-the-art few-shot NER methods and also re-implement three token-based few-shot NER methods originally implemented by *Few-NERD* [5]. In our experiments, we found the three token-based methods (i.e., **ProtoBERT** [20], **NNShot** and **StructShot** [27]) have shown comparable performance to more complicated methods, especially after careful re-implementation and hyperparameter tuning. We find that, in their original implementation, the dropout regularization(with dropout probability set to 0.5) that is directly applied to the final token embedding significantly decreases their performance on *Few-NERD*. In our re-implementation, we set the dropout probability to 0. **CONTaiNER** [3], **ESD** [23], and **DecomposedMetaNER** [17] are three recently proposed few-shot NER models and should represent the state-of-the-art methods on *Few-NERD* dataset. Among them, **CONTaiNER** is token-based while the other two are span-based Also, **ESD** [23] reach state-of-the-art performance on *SNIPS*. Additionally, we include **L-TapNet+CDT** [11] as the state-of-the-art token-based approach for *SNIPS*.

5.2 Efficiency Comparison

We conduct experiments to evaluate the efficiency of our model and the three state-of-art models on *Few-NERD*. The results are shown in the left plot of Fig. 3 (Note. we cannot measure CONTaiNER's prediction time in the 10-shot scenario since it causes Out-of-Memory error on GPU). It is clear the proposed SSP model is much faster than three state-of-art baselines.

To better understand which system components take more time in each model, we also break down the prediction time into three categories, i.e., fine-tuning, encoding, and inference, and report the results in the right part of Fig. 3.

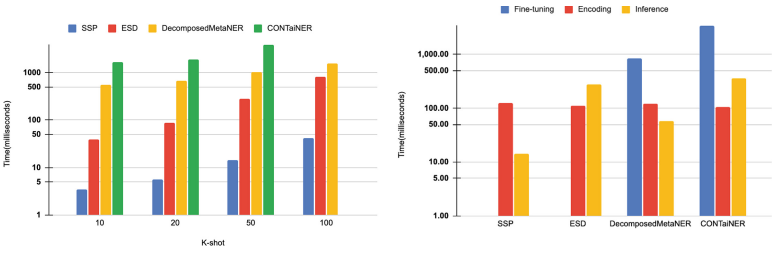

Fig. 3. Left: Prediction time on a single 10-way **K**-shot episode; Right: Prediction time breakdown on a single 10-way 5-shot episode; All tested on a 3090 GPU

The fine-tuning time measures the amount of time CONTaiNER and DecomposedMetaNER are fine-tuned on the support set while SSP and ESD do not have this step. The encoding time consists of both the BERT encoding time and the span/token representation extraction time. The inference time includes the

class center/nearest neighbor construction time and distance calculation time. For ESD and our model, the encoding time can be saved by pre-computing the representation offline. And only the inference time is needed for deploying the model. But for CONTaiNER and DecomposedMetaNER, since fine-tuning would update model parameters, the total prediction time would be the sum of fine-tuning, encoding, and inference time. As shown in Fig. 3, the fine-tuning time is the largest source of latency for models that have it. All models have similar encoding latency because they all use BERT. And our model has the lowest inference time thanks to its simplified structure. Fine-tuning is slow since it runs BERT forward and backward process multiple times while encoding only runs the forward process once. Our model has the lowest inference time since we avoid adding complicated attention mechanisms (used in ESD [23]) or CRF Viterbi decoding (used in CONTaiNER [3] and DecomposedMetaNER [17]) to our model.

5.3 Effectiveness Comparison

Table 1. Performance (F1 percent) on *Few-NERD*. †means we report the result in the original paper; ‡ means we run testing by use using provided checkpoint; cased and uncased denotes whether the backbone BERT is cased. Note: all baselines use uncased

Models	Intra				Inter			
	1 Shot		5 Shot		1 Shot		5 Shot	
	5 way	10 way	5 way	10 way	5 way	10 way	5 way	10 way
ProtoBERT	38.03 ± 0.29	31.43 ± 0.37	53.13 ± 1.03	46.07 ± 0.69	58.08 ± 0.26	52.00 ± 0.59	65.29 ± 0.51	60.54 ± 0.55
NNShot	37.89 ± 0.83	31.56 ± 0.39	50.90 ± 0.48	43.76 ± 0.41	55.62 ± 0.46	50.22 ± 0.34	63.50 ± 0.23	59.70 ± 0.21
StructShot	42.25 ± 0.55	35.25 ± 0.61	51.13 ± 0.26	44.98 ± 0.32	58.81 ± 0.34	53.62 ± 0.46	64.20 ± 0.26	60.22 ± 0.39
CONTaiNER‡	38.48	31.76	53.58	47.10	49.75	44.68	61.74	57.17
ESD†	36.08 ± 1.60	30.00 ± 0.70	52.14 ± 1.50	42.15 ± 2.60	59.29 ± 1.25	52.16 ± 0.79	69.06 ± 0.80	64.00 ± 0.43
DecomposedMetaNER†	**49.48 ± 0.85**	**42.84 ± 0.46**	62.92 ± 0.57	57.31 ± 0.25	64.75 ± 0.35	58.65 ± 0.43	71.49 ± 0.47	68.11 ± 0.05
SSP (uncased)	45.30 ± 0.53	38.34 ± 0.34	63.91 ± 0.18	57.99 ± 0.23	64.38 ± 0.11	58.88 ± 0.18	73.75 ± 0.08	70.56 ± 0.06
SSP (cased)	47.50 ± 0.36	39.79 ± 0.19	**66.16 ± 0.18**	**59.66 ± 0.14**	**65.98 ± 0.20**	**59.93 ± 0.18**	**75.09 ± 0.16**	**71.61 ± 0.12**

Previous experiment results show the SSP model is more efficient. We also conduct experiments to evaluate its effectiveness. Table 1 shows effectiveness comparison on the *Few-NERD* data. The SSP model consistently outperforms all baseline models in the 1-shot and 5-shot setting of *Few-NERD Inter* and the 5-shot setting of *Few-NERD Intra*. In the 1-shot setting of *Few-NERD Intra*, SSP performs slightly worse than DecomposedMetaNER, but performs slightly better than DecomposedMetaNER in the 1-shot setting of *Few-NERD Inter*. DecomposedMetaNER is a two-stage pipeline consisting of separate span detection and classification model. Therefore, we additionally calculate the span detection F1 score of our cased model(i.e. span type can be wrong, as long as it is not classified as O). It turns out that our model's span detection F1 score is 8–10% lower than DecomposedMetaNER in the 1-shot setting of *Few-NERD Intra*. This may indicate having separate span detection and classification is a possible improvement direction for our model.

Moreover, the three re-implemented token-based baselines can outperform more complicated SoTA methods in certain settings, highlighting the importance of properly implementing and tuning the baseline. Our re-implementation avoids applying dropout regularization directly on the token embedding. Moreover, we find if we apply dropout regularization(even with a drop probability of 0.1) to the span embedding in our SSP model, the performance also drops noticeably. This may indicate that the token/span embedding is highly correlated between different dimensions and applying dropout regularization would break such correlation. We find that implementation decisions are important, and simple models such as SSP, when implemented in the correct way, can achieve superior performance than existing complicated models.

Table 2. Performance (F1 percent) of baselines and our methods on *SNIPS*. †denotes the result reported in their paper. We report all seven cross-validation results, each time testing one user intent: Weather (We), Music (Mu), PlayList (Pl), Book (Bo), Search Screen (Se), Restaurant (Re), and Creative Work (Cr).

Models	We	Mu	Pl	Bo	Se	Re	Cr	Avg
ProtoBERT	78.58 ± 1.04	67.27 ± 0.32	79.07 ± 1.18	**90.30 ± 1.08**	82.43 ± 0.52	76.74 ± 1.31	73.91 ± 2.67	78.33
NNShot	80.18 ± 0.74	68.93 ± 1.20	74.24 ± 1.64	84.49 ± 1.17	83.24 ± 1.10	79.50 ± 0.52	73.51 ± 3.61	77.73
StructShot	83.26 ± 1.63	74.27 ± 0.68	77.94 ± 0.98	86.26 ± 1.16	85.89 ± 0.87	81.92 ± 0.30	72.83 ± 4.13	80.34
L-TapNet+CDT†	71.64 ± 3.62	67.16 ± 2.97	75.88 ± 1.51	84.38 ± 2.81	82.58 ± 2.12	70.05 ± 1.61	73.41 ± 2.61	75.01
ESD †	84.50 ± 1.06	66.61 ± 2.00	79.69 ± 1.35	82.57 ± 1.37	82.22 ± 0.81	80.44 ± 0.80	**81.13 ± 1.84**	79.59
SSP	**85.70 ± 2.56**	**74.28 ± 1.85**	**84.15 ± 0.48**	87.23 ± 0.73	**88.65 ± 0.73**	**82.83 ± 0.51**	78.83 ± 0.60	**83.09**

Table 2 reports the experiment result on *SNIPS*. On average, our model can outperform all our baselines in the 5-shot setting. This demonstrates our model's potential to be adapted as a few-shot slot-filling model. This is promising for scientific text mining because the slot-filling method has been successfully adapted to extract solid oxide fuel cell information [6] and chemical reaction information [10].

5.4 Additional Analysis

Span vs Token Representations: We conduct a detailed analysis to explore why span-based methods can outperform token-based ones. ProtoBERT is used to represent the token-based models. A simple version of our model that does not use whole class pre-training or Query-Support Attention, and uses uncased BERT is used to represent the span-based models for a fair comparison. All comparisons are carried out on the 5-way 5-shot setting of Few-NERD Inter.

First, we break the evaluation metric by different span lengths, as shown in Fig. 4 Left. Compared with the token-based model, the span-based model has significantly higher precision for single-word entities while having a slight advantage in both the precision and recall for multi-word entities. We hypothesize that the token-based model breaks multi-word entities into small spans more frequently and therefore causes a lot of false-positive single-word entities.

Therefore, we dig deeper and concentrate on two groups of entities: single-word prediction and multi-word ground truth. The analyzing result is demonstrated in Fig. 4 Mid and Right. For single-word prediction, we classified the error into three cases: (1) "Inside", denoting it is inside a ground truth entity that has the same label as the prediction (2) "Misclassified", denoting it has a wrong label; (3) "Outside", denoting it is not part of any ground truth entities. Clearly, the span-based method makes much fewer Inside errors. For multi-word ground truth, we classified the error into three cases: (1) "All O", denoting all the words inside the ground truth are misclassified as O; (2) "Partial O", denoting part of the words are misclassified as O; (3) Other, denoting the rest occasions. Here, we can see that the token base method makes more "Partial O" errors, indicating that the token base method breaks a lot of multi-word entities by misclassifying one of its tokens as O.

Ablation Studies: We also conduct ablation studies to explore the contribution of different components in our SSP model. Due to the huge number of possible variants, we split our ablation studies into two parts. We start with a simplified version and gradually add or modify some components until it is the same as the SSP model reported in Table 1. Both studies are carried out on *Few-NERD* and we report the averaged F1 score in percentage across all eight settings (i.e $5/10\,way \times 1/5\,shot \times Inter/Intra\,split$).

The first part focuses on studying the effect of span representation size and different representation normalization techniques as mentioned in Sect. 4. We do not use attention or pre-training in our model and we use uncased BERT in order to control the number of variables and also make SSP comparable to other span-based methods. We introduce the following variants of SSP, each with a different normalization strategy: (1) *No-Norm*, which removes the Layer Normalization layer; (2) *L2-Norm*, in which we not only remove Layer Normalization but also replaces the Euclidean distance with cosine distance $Tcosine(\widetilde{\mathbf{s}}_\mathbf{m} \cdot \widetilde{\mathbf{c}}_\mathbf{k})$; Here T is a temperature parameter and $T = 40$; (3) *LayerNorm**, which is a variation of LayerNorm by removing the re-scaling and re-centering part. The results of these three variations together with the original SSP, when the span embedding dimension is 768, are reported at the top part in the left plot of Fig. 5. The results

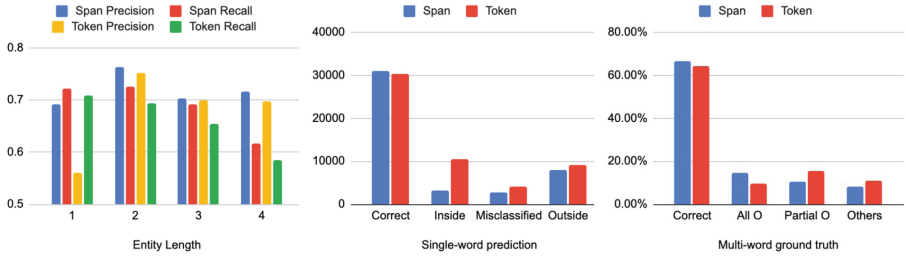

Fig. 4. Left: Evaluation metric breakdown by entity length; Mid: Single-word prediction breakdown Right: Multi-word ground truth breakdown

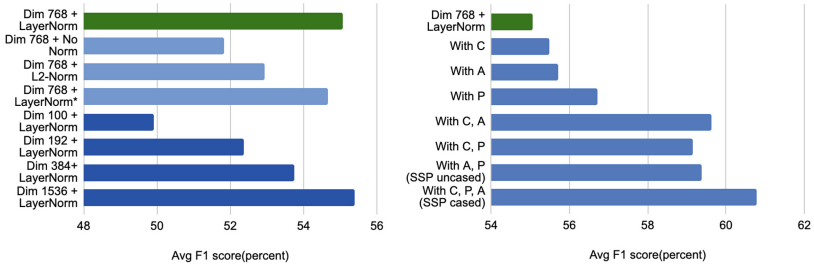

Fig. 5. Left: Ablations on Layer Normalization and representation dimension; Right: Ablations on different model components, where C denotes cased BERT, A denotes Query Support attention, and P denotes Whole class pre-training. The last two configurations is the same as SSP (cased/uncased) reported in Table 1

of SSP with the original LN but with different span embedding dimensions (i.e., 100, 192, 384, 1538) are shown at the lower part in the same plot.

The plot shows that having Layer Normalization on span representation greatly improves our model's performance over *No-Norm*. And the LayerNorm* result confirms our observation on trained γ and β, indicating that the affine projection which is applied after the re-scaling and re-centering operation makes little difference. Moreover, the L2-Norm result proves that just re-scaling is not enough, and re-centering also plays an important role here. Moreover, we can see that the span representation capacity (aka the embedding dimensions) also makes a huge difference. The bigger the span representation, the better the few-shot learning performance. This discovery is also in line with other studies in metric-learning [24]. We choose to fix the dimension to 768 for all other experiments since this is also the hidden size of *bert-base* and should make our model comparable to other token-based models(including DecomposedMetaNER which use the average of in-span token embedding as span representation).

In the second part of ablation studies, we start with the "Dim 768 + LayerNorm" configuration in the first part and gradually add or change some model components, including (1) **C** - Cased BERT encoder which replaces the uncased BERT (Sect. 5.1); (2) **A** - Query support Attention which replaces the mean pooling in the class center construction process (Sect. 4); (3) **P** - Whole class pre-training which pre-trains the model before mete-learning (Sect. 4). With the addition of **A** and **P**, our model variant is identical to the "SSP(uncased)" configuration reported in Table 1. When we further added the **C** component, our model is equivalent to "SSP(cased)". The right plot in Fig. 5 summarizes the result of our second part ablation study, which shows that all of the component modifications are necessary as they are complementary to each other.

6 Conclusion and Future Work

In this work, we present a simple span-based prototypical framework (SSP) for few-shot NER. Compared with the token-based models, SSP makes fewer

single-token errors and can better extract multi-word entities. We discovered that techniques such as layer normalization, query-support attention, and whole-class pre-training are beneficial for boosting model performance. Additionally, experimental results indicate that certain implementation details, such as dropout and span representation size, require careful consideration and tuning. Experiments on *Few-NERD* and *SNIPS* datasets show that SSP is significantly faster than existing state-of-the-art methods with comparable effectiveness.

Our work sheds a light on how to make few-shot NER suitable for domain applications, where there exists a large corpus to be analyzed within a limited amount of prediction time (e.g. scientific text mining [10,13,26,29]). Also, the problem we find can help construct future few-shot NER benchmarks to consider more real word influence factors. Additionally, our model can be combined with active learning to help accelerate annotation. Two potential use cases include (1) prioritizing the annotation of difficult examples and (2) saving search time by filtering rare entities out of a big corpus.

Acknowledgements. We greatly thanks all reviewers for their constructive comments. The research was supported as part of the Center for Plastics Innovation, an Energy Frontier Research Center, funded by the U.S. Department of Energy (DOE), Office of Science, Basic Energy Sciences (BES), under Award Number DE-SC0021166. The research was also supported in part through the use of DARWIN computing system: DARWIN - A Resource for Computational and Data-intensive Research at the University of Delaware and in the Delaware Region, Rudolf Eigenmann, Benjamin E. Bagozzi, Arthi Jayaraman, William Totten, and Cathy H. Wu, University of Delaware, 2021, URL: https://udspace.udel.edu/handle/19716/29071.

References

1. Ba, J., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv e-print archive (2016)
2. Cui, L., Wu, Y., Liu, J., Yang, S., Zhang, Y.: Template-based named entity recognition using BART. In: ACL-IJCNLP (2021)
3. Das, S.S.S., Katiyar, A., Passonneau, R., Zhang, R.: CONTaiNER: few-shot named entity recognition via contrastive learning. In: ACL (2022)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL (2019)
5. Ding, N., et al.: Few-NERD: a few-shot named entity recognition dataset. In: ACL-IJCNLP (2021)
6. Friedrich, A., et al.: The SOFC-exp corpus and neural approaches to information extraction in the materials science domain. In: ACL (2020)
7. Fritzler, A., Logacheva, V., Kretov, M.: Few-shot classification in named entity recognition task. In: SAC (2019)
8. Fu, J., Huang, X., Liu, P.: SpanNER: named entity re-/recognition as span prediction. In: ACL-IJCNLP (2021)
9. Gao, T., Han, X., Liu, Z., Sun, M.: Hybrid attention-based prototypical networks for noisy few-shot relation classification. In: AAAI (2019)
10. Guo, J., et al.: Automated chemical reaction extraction from scientific literature. J. Chem. Inf. Model. **62**(9), 2035–2045 (2021)

11. Hou, Y., et al.: Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In: ACL (2020)
12. Huang, J., et al.: Few-shot named entity recognition: an empirical baseline study. In: EMNLP (2021)
13. Kononova, O., et al.: Text-mined dataset of inorganic materials synthesis recipes. Sci. Data **6**(1), 1–11 (2019)
14. Li, Y., Liu, L., Shi, S.: Empirical analysis of unlabeled entity problem in named entity recognition. In: ICLR (2021)
15. de Lichy, C., Glaude, H., Campbell, W.: Meta-learning for few-shot named entity recognition. In: 1st Workshop on Meta Learning and Its Applications to Natural Language Processing (2021)
16. Ma, R., Zhou, X., Gui, T., Tan, Y.C., Zhang, Q., Huang, X.: Template-free prompt tuning for few-shot NER. In: NAACL (2022)
17. Ma, T., Jiang, H., Wu, Q., Zhao, T., Lin, C.Y.: Decomposed meta-learning for few-shot named entity recognition. In: ACL (2022)
18. Ouchi, H., et al.: Instance-based learning of span representations: a case study through named entity recognition. In: ACL (2020)
19. Reimers, N., Gurevych, I.: The curse of dense low-dimensional information retrieval for large index sizes. In: ACL (2021)
20. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: NIPS (2017)
21. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H.S., Hospedales, T.M.: Learning to compare: relation network for few-shot learning. In: CVPR (2018)
22. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NISP (2016)
23. Wang, P., et al.: An enhanced span-based decomposition method for few-shot sequence labeling. In: NAACL (2022)
24. Wang, X., Han, X., Huang, W., Dong, D., Scott, M.R.: Multi-similarity loss with general pair weighting for deep metric learning. In: CVPR (2019)
25. Wang, Y., Chu, H., Zhang, C., Gao, J.: Learning from language description: low-shot named entity recognition via decomposed framework. In: EMNLP (2021)
26. Weston, L., et al.: Named entity recognition and normalization applied to large-scale information extraction from the materials science literature. J. Chem. Inf. Model. **59**(9), 3692–3702 (2019)
27. Yang, Y., Katiyar, A.: Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In: EMNLP (2020)
28. Yu, J., Bohnet, B., Poesio, M.: Named entity recognition as dependency parsing. In: ACL (2020)
29. Zhang, Y., Wang, C., Soukaseum, M., Vlachos, D.G., Fang, H.: Unleashing the power of knowledge extraction from scientific literature in catalysis. J. Chem. Inf. Model. **62**(14), 3316–3330 (2022)