Few-Shot Named Entity Recognition with Biaffine Span Representation

Anonymous ACL submission

Abstract

While Named Entity Recognition (NER) is a widely studied task, making inferences of entities with only a few labeled data (i.e., few-shot NER) has been challenging. Correspondingly, the N-way K-shot NER task is proposed to recognize entities in the given N categories with only K labeled samples for each category. Existing methods treat this task as a sequence labeling problem, while this paper regards it as an entity span classification problem and designs a Biaffine Span Representation (BSR) method to learn contextual span dependency representation to fit into the classification algorithm. The BSR applies a biaffine pooling module to establish the dependencies of each word on the whole sentence and to reduce the dimension of word features, thus, the span representation could gain contextual dependency information to help improve recognition accuracies. Experimental study on four standard NER datasets shows that our proposed BSR method outperforms pre-trained language models and existing N-way K-shot NER algorithms in two types of adaptations (i.e., Intra-Domain Cross-Type Adaptation and Cross-Domain Cross-Type Adaptation). Notably, F_1 value has increased by an average of 13.77% and 18.30% on the 5-way 1-shot task and the 5-way 5-shot task, respectively.

1 Introduction

As a fundamental task in the Natural Language Processing (NLP) area, Named Entity Recognition (NER) is mainly about extracting the boundaries and categories of entities in the given sentences, which can provide helpful information for down-stream tasks like text classification (Lee et al., 2018), event detection (Popescu et al., 2011), question answering (Lee et al., 2007), and more. The current state-of-the-art models (Yamada et al., 2020; Straková et al., 2019; Sohrab and Miwa, 2018; Yu et al., 2020) have made progress in entity classification by applying pre-trained models and neural network architectures. These well-performed deep learning models require extensive and high-quality manually labeled data. However, these data might not be readily available due to factors including the lack of domain knowledge from human annotators, high cost of annotating the large-scale data, or the restriction of privacy and security (Lu et al., 2020). Therefore, the Few-Shot learning task is proposed to recognize unlabeled instances (query set) according to only a few labeled samples (support set) (Lu et al., 2020).

To tackle the Few-Shot NER problem, various types of methods have already been designed including prototype-based methods, noisy supervised pre-training methods, self-training methods, and meta-learning methods (Hofer et al., 2018; Fritzler et al., 2019; Huang et al., 2020). These methods usually apply a transfer learning strategy to learn information from other rich-resource domains to overcome the shortage of samples in the target domain, which brings a label-discrepancy problem. Recently, Li formulated a N-way K-shot NER task to recognize entities in the given N categories with only K labeled entities for each category (Li et al., 2020a). This task requires the same size of label categories for the source domain and the target domain to explore the establishment of shared models during transfer learning. Li tackled this task in the sequence labeling framework, which needs to label each word according to various labeling schema (Li et al., 2012) including "BIO" (Beginning, Inside, Outside) and "BIOES" (Begin, Inside, Outside, End, Single).

In this paper, we reformulate the N-way K-shot NER task as an entity span classification problem to simplify the task. Rather than labeling each word with entity types and entity boundaries, we just need to determine which type each span belongs to. To improve the classification accuracy, we note that the dependency between words within an entity span and words outside this entity span can help

to recognize entities. Take the sentence "Parkinson's disease is a brain disorder" as an example, "Parkinson's disease" has a dependency on "brain disorder" and can help to recognize span "brain disorder" as a "Disease" entity. To characterize this dependency during the entity span representation learning, we design a Biaffine Span Representation (BSR) method by applying a biaffine pooling module to establish the dependencies of each word on the whole sentence and reduce the dimension of word features. Thus, the span representation could gain contextual dependency information to help improve recognition accuracies. After obtaining the representation, we classify entity spans into entity categories by a FeedForward Neural Network (FFNN) classifier. Experiments on four NER datasets show significant improvements of our BSR method over pre-trained language models and existing N-way K-shot learning algorithms in terms of F_1 value.

The rest of this paper is organized as follows. Section 2 states the N-way K-shot problem in the NER task and explains existing Few-Shot learning methods and entity annotation frameworks in NER. Section 3 details our BSR model for the N-way K-shot NER task. Following the experimental study in Section 4, we conclude this paper in Section 5.

2 Background

This section first introduces the Few-Shot Named Entity Recognition (NER) problem and the N-way K-shot scenario. Then, we discuss current Few-Shot learning methods in NER. Finally, we detail two frameworks to generate entity label sequences in NER, namely, the sequence-labeling-based framework and the entity-span-based framework.

2.1 Problem Statement

The Few-Shot NER problem aims to build a model $\mathcal{F} : \mathcal{S} \rightarrow \mathcal{Y}$ with sentences $\mathcal{S}$ as the input and entity label sequences $\mathcal{Y}$ as the output, where the input sentences $\mathcal{S}$ only contain a few instances of entities. The N-way K-shot NER task puts constraints on the general Few-Shot NER by limiting the size of entity type set Y to N for the input sentences $\mathcal{S}$. Therefore, it just needs to fine-tune the model learned on other rich-resource domains when applying the transfer learning strategy. The formal descriptions are as follows.

Suppose $\mathcal{S}_i$ and $\mathcal{S}_j$ are the sentences in the source domain i and the target domain j, respectively. The sizes of entity types in $\mathcal{S}_i$ and $\mathcal{S}_j$ are both fixed as N. $\mathcal{S}_j$ is further divided into a support set $\mathcal{S}_j^{spt}$ and a query set $\mathcal{S}_j^{qry}$. The N-way K-shot NER task first trains a model on $\mathcal{D}_i = \{\mathcal{S}_i, \mathcal{Y}_i\}$, where $\mathcal{Y}_i$ is corresponding label sequences of $\mathcal{S}_i$. Then it makes adaptations on $\mathcal{S}_j$, i.e., it first fine-tunes the model on $\mathcal{D}_j^{spt} = \{\mathcal{S}_j^{spt}, \mathcal{Y}_j^{spt}\}$ and then predicts the label sequences for $\mathcal{D}_j^{qry} = \{\mathcal{S}_j^{qry}\}$, where $\mathcal{Y}_j^{spt}$ is the corresponding label sequences of $\mathcal{S}_j^{spt}$ and each entity type in $\mathcal{S}_j^{spt}$ only contains K entities. We call $\mathcal{D}_j^{spt}$ as the N-way K-shot setting. There are two fundamental rules for datasets in domain i and j:

- The entity types in $\mathcal{D}_i$ are different from types in $\mathcal{D}_j : Y_i \cap Y_j = \varnothing$.
- The entity types in $\mathcal{D}^{spt}$ are the same as types in $\mathcal{D}^{qry}$, but sentences appear in $\mathcal{D}^{spt}$ will NEVER appear in $\mathcal{D}^{qry} : Y^{spt} = Y^{qry}$, $\mathcal{S}^{spt} \cap \mathcal{S}^{qry} = \varnothing$.

2.2 Few-Shot NER methods

The N-way K-shot NER task is a newly presented Few-Shot NER setting. There are only a few works on it. Li first investigated it and built a model that can be effectively adapted to new tasks by updating a small set of low-dimensional parameters (Li et al., 2020a). As Few-Shot NER methods could also be adjusted on this new task, we also reviewed related works on these methods.

Many Few-Shot NER methods are based the transfer learning strategy, which aims at training models from source domains and making adaptations on the target domain; that is, training the model $\mathcal{F} : \mathcal{S}_j \rightarrow \mathcal{Y}_j$ that maps sentences $\mathcal{S}$ to labels $\mathcal{Y}$ in target domain j based on the model learned in source domain i. The target domain j only contains a few labeled sentences, while the source domain i can have plenty of labeled data (Yang et al., 2021; Ding et al., 2021). These methods face the challenge that the model may need to be trained from scratch on the target domain j and the information of label dependencies learned in source domains might be unsuitable for the target domain (Hou et al., 2019).

Meta-learning-based methods are another type of methods, which aim at training models by a few samples and applying them to new tasks directly, without retraining from scratch. Meta-learning can be categorized as metric-based methods that

need to calculate the distance between training samples' centers and test samples (Snell et al., 2017), memory-based methods that need to store and maintain input representations in an external memory (Florez and Mueller, 2019), and optimization-based methods that need to split tasks into masses meta-tasks (Finn et al., 2017; Li et al., 2020a).

Besides, some other Few-Shot learning strategies are also designed. Li regarded NER as a logical rule extraction problem. They utilized a weakly supervised named entity tag as a seed rule and designed a dynamic label selection method to generate new rules and build new tags (Li et al., 2021). Jiang combined the weakly labeled data with the strongly labeled data during training and suppressed the extensive noise data to make inferences (Jiang et al., 2021). Tong learned the semantics hid in the "O" label to improve results (Tong et al., 2021). Ma used prompt-based models to deal with Few-Shot NER tasks (Ma et al., 2021).

2.3 Sequence Labeling and Entity Span

NER tasks require to obtain the label sequences for entities. Above section reviews related models for Few-Shot NER. After training the models, we should also transfer the model outputs into label sequences. Currently, there are mainly two widely used frameworks for generating label sequences, namely, the sequence-labeling-based framework and the entity-span-based framework. When applying different frameworks to formulate the problem, the NER models should also be adjusted accordingly.

The sequence-labeling-based framework can be defined as: for the input sentence s with n tokens, denoted by $s = \{w_1, \ldots, w_n\}$, assign each token w_i a label l_i which belongs to the pre-defined entity type set Y and generate a label sequence $y = \{l_1, \ldots, l_n\} \in Y$. To locate the boundaries of entities in s, many sequence labeling schema have been proposed to add extra tags for labels in y (Lin et al., 2020; Kruengkrai et al., 2020; Tabassum et al., 2020). For example, in the "BIO" sequence labeling schema, words in non-entities will be labeled as "O", and the label of the word in the start location of an entity will be added with a "B-" whereas other locations in this entity will be added with a "I-". To be specific, the example sentence "Parkinson's disease is a $[\text{brain disorder}]_{\text{Disease}}$." will be annotated as "O, O, O, O, B-Disease, I-Disease". Therefore, methods in this framework need to label both entity types and entity boundaries. Besides, as the labels are added word by word, these methods are hard to handle nested NER (some entities are within an entity) tasks, for example, it is uneasy to label the "$[\text{brain}]_{\text{BodyPart}}$" entity within the "$[\text{brain disorder}]_{\text{Disease}}$" entity in the example sentence. The extracted outer entities ("brain disorder") will also affect the performance of the recognition of inner entities ("brain").

The entity-span-based framework splits sentences into several entity spans, where each span is a span of tokens $\{w_p, \ldots, w_q\}$ $(1 \leqslant p \leqslant q \leqslant n)$, and then calculates classification results $\hat{y} \in Y$ for each entity span. For example, the entity span "brain disorder" in the example sentence will be labeled as "Disease", the entity span "brain " will be labeled as "BodyPart" and the entity span "disease is a" will be labeled as "O". In this way, the entity span formulation converts NER tasks into multi-class classification problems rather than determining entity types and entity boundaries like sequence-labeling-based methods (Joshi et al., 2020; Yamada et al., 2020; Yu et al., 2020). Comparatively speaking, entity-span-based methods simplify the task and may achieve better performances, although they may increase the time complexity. This is also the reason we try to use entity span to formulate the N-way K-shot NER task.

3 Methodology

In this section, we justify the intuition of the BSR model and explain details about the BSR model. Afterwards, we introduce the training and adapting process of the BSR model.

3.1 Biaffine Span Representation

Following the entity span framework, we formulate the N-way K-shot NER task as an entity span classification problem. After inputting sentences, we learn the span representation and then train a classifier to determine entity types. Therefore, the span representation plays an essential role.

As mentioned in the introduction section, we note that the dependency between words within an entity span and words outside this entity span can help to recognize entities, just as "Parkinson's disease" can help to recognize span "brain disorder" as a "Disease" entity for the sentence "Parkinson's disease is a brain disorder". To characterize these dependencies, we apply a biaffine pooling module, an idea similar as Yu's work (Yu et al., 2020), how-

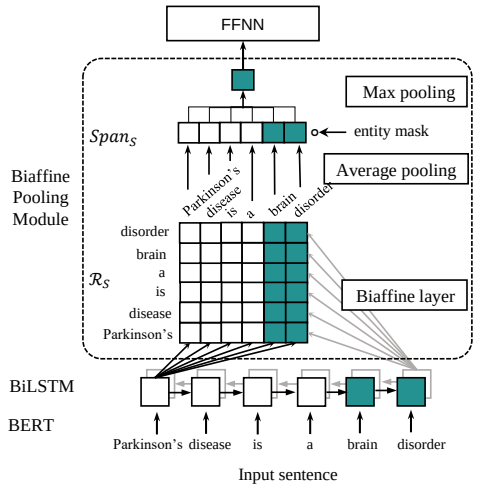

Figure 1: The whole architecture of our BSR model. Words in the sentence will first be embedded by BERT and the hidden features will then be learned by a BiLSTM layer. To get the dependency representation between an entity span (words in blue) and their surrounding words, we deploy a biaffine pooling module, which consists of a biaffine layer and a pooling layer. Finally the span representation is fit into a FFNN classifier.

ever, they only established the dependency between the head and the tail for each entity span to make classification for general NER tasks. Besides, we also perform dimension reduction in the biaffine pooling module to improve the generalization of our BSR model for N-way K-shot learning. Figure 1 shows an overview of the whole architecture.

We first use $\text{BERT}_{\text{base}}$ (Devlin et al., 2018) to get word embedding vector w_i for each word in sentence s and create entity mask $\mathcal{M}_s$ for each entity span. Here, $w_i \in \mathbb{R}^d$ where d is the result dimension of $\text{BERT}_{\text{base}}$ and $\mathcal{M}_s = \{0,1\}^l$ where l denotes the number of words in the sentence. After that, a BiLSTM layer is applied to learn sentence representations. Let $\mathcal{H}_f(i)$ and $\mathcal{H}_b(i)$ respectively denote forward and backward results obtained by the BiLSTM layer for w_i in s.

We then need to learn the contextual dependency representation for each span, which is accomplished by a biaffine pooling module in our BSR model. In this module, a biaffine layer is first used to compute a $l_f \times l_b \times r$ sentence representation tensor $\mathcal{R}_s$ for each sentence s:

$$\mathcal{R}_s = \mathcal{H}_f{}^{\intercal}\, \mathbf{U}_s^1\, \mathcal{H}_b + (\mathcal{H}_f \oplus \mathcal{H}_b)^{\intercal}\, \mathbf{U}_s^2 + \mathbf{b} \quad (1)$$

where r is a low-dimensional space, $l_f \in \mathbb{R}^l$ is the forward word representation, $l_b \in \mathbb{R}^l$ is the backward word representation, $\mathbf{U}_S^1$ is a $h \times r \times h$ tensor with h as the dimension of $\mathcal{H}_f$ or $\mathcal{H}_b$, $\mathbf{U}_S^2$ is a $2h \times r$ matrix, and b is the bias.

In Figure 1, $\mathcal{R}_s$ is represented as a $l \times l$ matrix and each element in the matrix is a dependency feature vector in the low-dimensional space, i.e., $\mathcal{R}_s(i,\ j)$ means the biaffine dependency score of the i^{th} word on the j^{th} word.

After the biaffine layer, we adopt two pooling strategies to obtain the span dependency representation including the average pooling and the max pooling. The average pooling is used to calculate the average dependency scores of each word on the whole sentence, while the max pooling is to extract the max vector within each span to highlight the important features of each word in this span. The max pooling is only performed on words within the same span, therefore, we add a mask operation in the average pooling for convenience. We mask the words that are not in the span to get average span dependency scores $Span_s \in \mathbb{R}^l \times \mathbb{R}^r$ by Hadamard product of entity mask $\mathcal{M}_s$ and average sentence representation $MEAN(\mathcal{R}_s)$:

$$Span_s = MEAN(\mathcal{R}_s) \circ \mathcal{M}_s \quad (2)$$

Finally we utilize a FFNN to classify entity span categories.

3.2 Training and Adapting Algorithm

Algorithm 1 summarizes the procedures of using our BSR model to train and adapt on source domain i and target domain j.

When training the BSR model on the source domain i, named task $\mathcal{T}_i$, we initialize parameters and sample a batch of subsets $\{\mathcal{D}_i^m = \{\mathcal{S}_i^m, \mathcal{Y}_i^m\}\}_{m=1,2,...}$ from $\{\mathcal{S}_i, \mathcal{Y}_i\}$ (line 3 to 6). We iteratively train the model on each $\mathcal{D}_i^m$ by utilizing gradient descent to optimize model parameters with cross-entropy loss (line 7 to 13). To reduce the training time, we shrink training epochs for subsequent subtasks.

For adapting, the trained model is fine-tuned on support dataset $\mathcal{D}_j^{spt}$ from the target domain j, named task $\mathcal{T}_j$ (line 17 to 20). Finally, the model is applied to make inference on unlabeled dataset $\mathcal{D}_j^{qry}$ (line 21).

4 Experiments

In this section, we present our experimental study on the proposed BSR model for the N-way K-shot NER task. We first introduce four public

Algorithm 1: Training and Adapting of BSR Model

Source-domain-epoch and target-domain-epoch mean the training epoch in source domain and target domain, respectively. Gradient descent ∇ is applied to calculate cross-entropy loss $\mathcal{L}$ and optimize model parameter θ.

Training():
begin
  Set learning rate α for task $\mathcal{T}_i$;
  Initialize BSR model parameter θ;
  Initialize Source-domain-epoch;
  Sample batch of subtasks $T = \{\mathcal{D}_i^m\}_{m=1,2,...}$;
  for $\mathcal{D}_i^m$ *in* T **do**
    for e *in source-domain-epoch* **do**
      $\theta_e = \theta_{e-1} - \alpha \nabla_{\theta_{e-1}} \mathcal{L}_{\mathcal{D}_i^m}(\hat{\mathcal{Y}}_i^m, \mathcal{Y}_i^m)$;
    end
    decrease source-domain-epoch;
  end
  return θ;
end
Adapting():
begin
  set learning rate β for task $\mathcal{T}_j$;
  for e *in target domain-epoch* **do**
    $\theta_e = \theta_{e-1} - \beta \nabla_{\theta_{e-1}} \mathcal{L}_{\mathcal{D}_j^{spt}}(\hat{\mathcal{Y}}_j^{spt}, \mathcal{Y}_j^{spt})$;
  end
  Evaluate $\mathcal{D}_j^{qry}$ using the model $BSR(\theta)$;
  return evaluation performance;
end

Table 1: Datasets used in experiments

Dataset	Domain	Types	Sentences
OntoNotes	Various	18	179,062
GENIA	Medical	36	18,546
BioNLP13CG	Medical	16	6,018
ManyEnt-53	Various	53	54,021

datasets and several baseline models including pre-trained language models and typical few-shot learning models. Next, we summarize hyper-parameters in the BSR model. And finally, we split the N-way K-shot NER task into Intra-Domain Cross-Type Adaptation and Cross-Domain Cross-Type Adaptation, and report the experimental results of each adaptation.

4.1 Datasets, Baselines, and Evaluation Metric

We use four standard NER datasets for evaluation including OntoNotes (Pradhan et al., 2013), GENIA (Kim et al., 2003), BioNLP13CG (Nédellec et al., 2013), and ManyEnt-53 (Eberts et al., 2020) as shown in Table 1. We compare the BSR model with the following:

- Pre-trained Models: We utilize Flair (Akbik et al., 2018), ELMo (Peters et al., 2018), and BERT (Vaswani et al., 2017) models to get word embeddings and build a CRF layer to predict the labels of entities.

- ProtoNet (Snell et al., 2017): This is a metric-based meta-learning method for the Few-Shot NER task. It focuses on computing distances between the center points of training samples (prototypes) and test samples. The model is afterwards optimized by calculating cross entropy loss between test samples and prototypes.

- MAML (Finn et al., 2017): This optimization-based meta-learning method uses a large-scale dataset in the source domain i to optimize model parameters. It can be quickly adapted to new tasks in the target domain j over a few gradient steps. During the training procedure, the dataset in domain i is separated into several few-shot subtasks. The loss of each subtask is recorded, and the total loss of all subtasks can optimize the model.

- FEWNER (Li et al., 2020a): This is an optimization-based meta-learning method that focuses on the N-way K-shot NER task. It defines two kinds of model parameters: context parameters φ and sequence labeling parameters θ. In training, the model first updates φ by loss of each subtask in the source domain i and updates θ according to φ. In testing, it maintains θ and updates φ according to the Few-Shot task in the target domain j, and finally makes inference for the task without labels in domain j. The FEWNER achieves SoTA results in several tasks.

We utilize $F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$ value as the evaluation metric to balance the influence of $Precision$ value and $Recall$ value.

4.2 Experiment Settings

To encode word into vectors, we use $\text{BERT}_{\text{base}}$ which has 12 heads of attention layers and 768 word-embedding dimensions. The BiLSTM layer has one hidden layer whose hidden size is 200. The initial states h_0 and c_0 are initialized randomly. The dimension in the Biaffine layer is 100. We normalize the results after the Biaffine layer and utilize a dropout of 0.2. Different from meta-learning-based

Table 2: Intra-Domain Cross-Type Adaptation performance on four datasets (%)

Methods	GENIA 5-way		OntoNotes 5-way		BioNLP13CG 5-way		ManyEnt-53 5-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Flair	9.77	11.44	13.08	17.62	7.26	17.82	11.68	12.93
ELMo	15.21	19.18	13.28	15.53	7.73	21.56	12.85	16.15
BERT	12.02	14.93	11.55	13.14	11.06	24.22	11.52	15.94
ProtoNet	12.34	15.03	12.01	19.33	13.31	21.93	13.78	23.91
MAML	13.73	16.46	10.51	15.10	14.02	25.27	15.17	21.43
FEWNER	23.24	29.19	22.84	39.67	22.55	36.89	25.87	38.01
BSR	**25.65**	**42.56**	**26.99**	**54.59**	**27.33**	**52.69**	**30.21**	**49.00**

models (Li et al., 2020a) which need masses of subtasks to learn cross domain information, our transfer-learning-based BSR model requires less subtasks whereas each subtask contains more sentences. To be specific, the number of subtasks is 3 and each subtask contains no more than 5000 sentences. In the training procedure, learning rate α for task $\mathcal{T}_i$ is 0.00005 and we initialize source-domain-epochs to 20 and decrease it by step 4 for each subsequent subtask. In the adapting procedure, learning rate β for task $\mathcal{T}_j$ is the same as α and the target domain-epochs is 30. In the final evaluation procedure, we transfer entity span classification results to labeling sequences and calculate F_1 value.

4.3 Experimental Results

We evaluate our BSR model on two types of adaptations including Intra-Domain Cross-Type Adaptation and Cross-Domain Cross-Type Adaptation.

For Intra-Domain Cross-Type Adaptation, we first randomly sample N entity types (N-way) in each dataset and select sentences using a sampling method (referring to Appendix). Secondly, we mask all entities whose types are in N-way to constitute a source domain dataset $\mathcal{D}_i$. Afterwards, we further mask all entities whose types are not in N-way to generate a target domain dataset $\mathcal{D}_j$ to ensure rule $Y_i \cap Y_j = \varnothing$. Although the source domain dataset $\mathcal{D}_i$ and the target domain dataset $\mathcal{D}_j$ come from the same original dataset, their entity types are different (Intra-Domain Cross-Type).

For Cross-Domain Cross-Type Adaptation, we use the fully labeled dataset in domain i as the source domain dataset $\mathcal{D}_i$, and fully labeled dataset in domain j as the target domain dataset $\mathcal{D}_j$. $\mathcal{D}_i$ and $\mathcal{D}_j$ come from different datasets and their entity types are also different (Cross-Domain Cross-Type).

4.3.1 Intra-Domain Cross-Type Adaptation

In this experiment, for each target domain dataset, we sample 5 types (5-way) randomly. Afterwards, we apply the dataset generation method mentioned in appendix A to generate a Few-Shot support dataset $\mathcal{D}_j^{spt} = \{\mathcal{S}_j^{spt}, \mathcal{Y}_j^{spt}\}$, and finally, we construct a Few-Shot query dataset $\mathcal{D}_j^{qry}$ by randomly sampling 5000 sentences from $\{\mathcal{S}_j - \mathcal{S}_j^{spt}\}$.

Table 2 shows the average F_1 results of Intra-Domain Cross-Type Adaptation over 10 iterations with different random seeds. For the GENIA dataset, the backbone of the word embedding method in baseline models is the combination of Glove and character-level-CNN introduced in the original paper (Li et al., 2020a), whereas for other datasets is the learnable BERT model. Our BSR model achieves the state-of-the-art results and improves F_1 value on FEWNER by 2.41%, 4.15%, 4.78%, 4.34% in 1-shot setting, and 13.37%, 14.92%, 15.80%, 10.99% in 5-shot setting for GENIA, OntoNotes, BioNLP13CG, and ManyEnt datasets, respectively. The results of the average F_1 value in baseline models show that the BERT model performs better than Glove and character-level-CNN in the 5-shot setting when used as the backbone layer. However, the advantage of the BERT model becomes slighter when it comes to 1-shot setting.

All the results of 5-shot are better than those of 1-shot, and BSR model has a prominent improvement from 1-shot to 5-shot in comparison with the baseline models. Results on the GENIA dataset show lower F_1 values than on other datasets. We suppose the reason is due to the imbalance label type distribution. For example, the type "RNA_N/A" appears only 15 times in the whole GENIA dataset whereas the type "protein_molecule" appears 21,632 times. For dataset ManyEnt-53, the least frequent entity type is "military collective" which appears 985 times and the most frequent entity type is "human"

Table 3: Cross-Domain Cross-Type Adaptation performance on three datasets to BioNLP13CG (%)

Methods	GENIA → BioNLP13CG 5-way		OntoNotes → BioNLP13CG 5-way		ManyEnt-53 → BioNLP13CG 5-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
Flair	10.53	12.49	8.37	9.15	8.09	13.95
ELMo	10.39	11.45	10.76	11.85	13.36	19.12
BERT	13.36	15.15	9.15	9.98	15.38	18.87
ProtoNet	14.05	15.38	8.34	8.93	16.26	26.42
MAML	14.98	17.34	9.22	10.57	16.05	24.79
FEWNER	22.46	27.94	13.09	15.46	17.20	25.67
BSR	**35.99**	**49.12**	**38.73**	**52.78**	**32.93**	**50.41**

which appears 21,888 times.

4.3.2 Cross-Domain Cross-Type Adaptation

In this experiment, for each target domain dataset, we sample 5 types (5-way) randomly. Afterwards, we apply the dataset generation method mentioned in appendix A to generate a Few-Shot support dataset $\mathcal{D}_j^{spt} = \{\mathcal{S}_j^{spt}, \mathcal{Y}_j^{spt}\}$, and finally, we construct a Few-Shot query dataset $\mathcal{D}_j^{qry}$ by randomly sampling 5000 sentences from $\{\mathcal{S}_j - \mathcal{S}_j^{spt}\}$. We let BioNLP13CG as the target domain j and other datasets as the source domain i, and design three adaptations: GENIA → BioNLP13CG, OntoNotes → BioNLP13CG, and ManyEnt-53 → BioNLP13CG, respectively.

Table 3 shows the average F_1 value results of Cross-Domain Cross-Type Adaptation during 10 experiments with different random seeds. In GENIA → BioNLP13CG and OntoNotes → BioNLP13CG adaptations, the backbone of word embedding method in baseline models is the combination of Glove and character-level-CNN, from original paper (Li et al., 2020a). And for ManyEnt-53 → BioNLP13CG adaptation the backbone of word embedding method is learnable BERT model. BSR method achieves state-of-the-art results and improves F_1 value on FEWNER by 13.53%, 25.64%, 15.73% in 1-shot setting, and 21.18%, 37.32%, 27.74% in 5-shot setting for these adaptations, respectively. The FEWNER model shows that the inference performance in target domain j is affected by the source domain i: the result F_1 value is higher when source domain i and target domain j are medical in GENIA → BioNLP13CG adaptation. However, the F_1 value is lower when source domain i and target j are different in OntoNotes → BioNLP13CG adaptation. Experiment results show BSR has a more stable result in three adaptations compared with FEWNER: the F_1 value is not influenced by source/target domains and remains at approximately 50%.

4.3.3 Ablation study on Intra-Domain Cross-Type Adaptation

Table 4 reports an ablation analysis on Intra-Domain Cross-Type adaptation using the BioNLP13CG dataset. We evaluate the different sub-components and parameter settings of the proposed BSR model including the biaffine layer and the BiLSTM layer; the different dimensions of BERT model hidden status to get different word embedding vectors; the different numbers of subtasks during the training procedure; the different numbers of initial target-domain-epochs during the training procedure.

Table 4: Ablation study on Intra-Domain Cross-Type Adaptation with dataset BioNLP13CG

	1-shot	5-shot
Origin model	27.33	52.69
w/o BiLSTM layer	1.61 ↓	3.48 ↓
w/o biaffine layer	2.81 ↓	2.42 ↓
BERT hidden status dimension : 128	-7.71 ↓	-14.01 ↓
BERT hidden status dimension : 256	-3.87 ↓	-6.59 ↓
BERT hidden status dimension : 512	-2.97 ↓	-3.23 ↓
Training subtask T num : 1	-3.05 ↓	-5.77 ↓
Training subtask T num : 5	+4.72 ↑	-0.34 ↓
Training subtask T num : 7	-1.91 ↓	-3.52 ↓
Training subtask T num : 9	+0.95 ↑	+1.47 ↑
initial source-domain-epochs : 10	-3.37 ↓	-1.51 ↓
initial source-domain-epochs : 15	+1.13 ↑	-2.21 ↓
initial source-domain-epochs : 25	-2.24 ↓	-2.68 ↓
initial source-domain-epochs : 30	-2.00 ↓	-1.92 ↓

We remove the biaffine layer and compare it with the original model. As shown in Table 4, the model without the biaffine layer could also outperform the sequence labeling baseline models, whereas the F_1 value decreases 2.81% and 2.42% in the 5-way Intra-Domain Cross-Type Adaptation with 1-shot and 5-shot setting, respectively. We also replace the BiLSTM layer with two linear layers and compare it with the original model. The F_1 value of the new

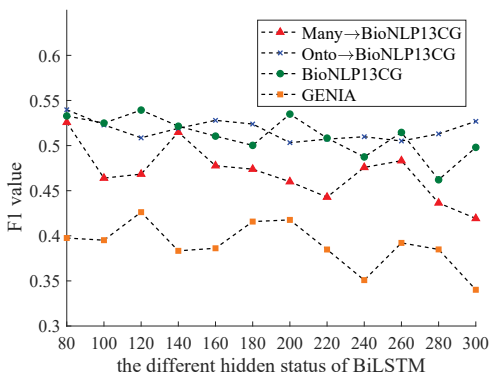

Figure 2: The average F_1 value from 10 times experiments with different random seed for four adaptations with different dimensions of BiLSTM hidden status

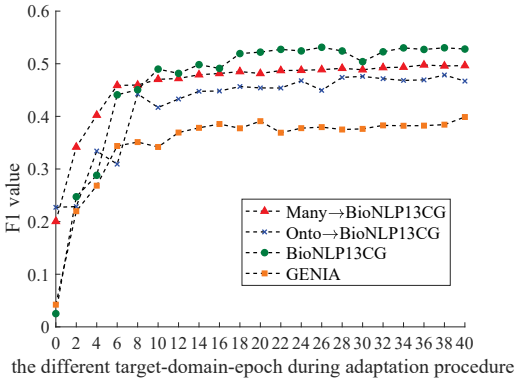

Figure 3: The average F_1 value from 10 times experiments with different random seed for four adaptations with different numbers of target-domain-epochs

model decreases 1.61% and 3.48% in the 5-way Intra-Domain Cross-Type Adaptation with 1-shot and 5-shot setting, respectively.

It is observed that BERT with a lower hidden status dimension (128, 256, and 512) could not outperform the initial setting (768), and the average F_1 value increasingly grows with the increase of hidden status dimensions. We suppose a lower parameter number in BERT would learn less information from the training datasets. In this specific adaptation, we find that when increasing the number of subtasks (e.g., 9 subtasks), the accuracies only increase a little with the cost of prolonging training time. After balancing accuracies and training time, we decide to use a small subtask number. Further, the number of source-domain-epochs could not exert a tremendous influence on results. The model has already fit the subtasks in the source domain with training epochs less than 10.

Figure 2 shows the performance on different BiLSTM hidden status in a 5-way 5-shot setting on four adaptations. The experiment settings are the same as in Section 4.2, except that the hyper-parameter BiLSTM hidden status is adjusted dynamically (from 80 to 300). Moreover, we also decrease the initial source-domain-epochs to 10 for less training time. Every experiment for each adaptation is executed 10 times and the average F_1 value is calculated. The result on GENIA dataset is the worst among all four adaptations and the average F_1 value has a declining trend with the increasing dimension of BiLSTM hidden status.

Figure 3 shows the performance on different numbers of target-domain-epochs in a 5-way 5-shot setting on four adaptations. Experiment settings are the same as in Section 4.2 except that the hyper-parameter target-domain-epochs is dynamically adjusted (from 0 to 40). Results in Figure 3 show that the average F_1 values of four adaptations become almost stable after executing 10 target-domain-epochs, and this supports our assumption that the model requires fewer training epochs during the testing procedure.

5 Conclusion

This paper treats the N-way K-shot NER task as an entity span classification problem and proposes the BSR method to learn the entity span representation for tackling this task. More specifically, we calculate contextual dependency score of entity spans with their surrounding words in the sentence and use FFNN to classify the span representation generated by the biaffine pooling module. We then evaluate BSR with four standard NER datasets based on two kinds of adaptations: Intra-Domain Cross-Type Adaptation and Cross-Domain Cross-Type Adaptation. Experimental results show that the proposed BSR model learns the features through fewer labeled entities and outperforms other pre-trained models and Few-Shot learning baseline models. Our future work is to identify features that cause the model to make wrong predictions.

References

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th international conference on computational linguistics*, pages 1638–1649.

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*.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. 2021. Few-nerd: A few-shot named entity recognition dataset. *arXiv preprint arXiv:2105.07464*.

Markus Eberts, Kevin Pech, and Adrian Ulges. 2020. Manyent: A dataset for few-shot entity typing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5553–5557.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR.

Omar U Florez and Erik Mueller. 2019. Learning to control latent representations for few-shot learning of named entities. *arXiv preprint arXiv:1911.08542*.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. 2019. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000.

Maximilian Hofer, Andrey Kormilitzin, Paul Goldberg, and Alejo Nevado-Holgado. 2018. Few-shot learning for named entity recognition in medical text. *arXiv preprint arXiv:1811.05468*.

Yutai Hou, Zhihan Zhou, Yijia Liu, Ning Wang, Wanxiang Che, Han Liu, and Ting Liu. 2019. Few-shot sequence labeling with label dependency transfer and pair-wise embedding. *arXiv preprint arXiv:1906.08711*.

Jiaxin Huang, Chunyuan Li, Krishan Subudhi, Damien Jose, Shobana Balakrishnan, Weizhu Chen, Baolin Peng, Jianfeng Gao, and Jiawei Han. 2020. Few-shot named entity recognition: A comprehensive study. *arXiv preprint arXiv:2012.14978*.

Haoming Jiang, Danqing Zhang, Tianyu Cao, Bing Yin, and Tuo Zhao. 2021. Named entity recognition with small strongly labeled and large weakly labeled data. *arXiv preprint arXiv:2106.08977*.

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

J-D Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.

Canasai Kruengkrai, Thien Hai Nguyen, Sharifah Mahani Aljunied, and Lidong Bing. 2020. Improving low-resource named entity recognition using joint sentence and token labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5898–5905.

Changki Lee, Yi-Gyu Hwang, and Myung-Gil Jang. 2007. Fine-grained named entity recognition and relation extraction for question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 799–800.

Ho-Suk Lee, Hong-Rae Lee, Jun-U Park, and Yo-Sub Han. 2018. An abusive text detection system based on enhanced abusive and non-abusive word lists. *Decision Support Systems*, 113:22–31.

Jiacheng Li, Haibo Ding, Jingbo Shang, Julian McAuley, and Zhe Feng. 2021. Weakly supervised named entity tagging with learnable logical rules. *arXiv preprint arXiv:2107.02282*.

Jing Li, Billy Chiu, Shanshan Feng, and Hao Wang. 2020a. Few-shot named entity recognition via meta-learning. *IEEE Transactions on Knowledge and Data Engineering*.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020b. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*.

Qi Li, Haibo Li, Heng Ji, Wen Wang, Jing Zheng, and Fei Huang. 2012. Joint bilingual name tagging for parallel corpora. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1727–1731.

Bill Yuchen Lin, Dong-Ho Lee, Ming Shen, Ryan Moreno, Xiao Huang, Prashant Shiralkar, and Xiang Ren. 2020. Triggerner: Learning with entity triggers as explanations for named entity recognition. *arXiv preprint arXiv:2004.07493*.

Jiang Lu, Pinghua Gong, Jieping Ye, and Changshui Zhang. 2020. Learning from very few samples: A survey. *arXiv preprint arXiv:2009.02653*.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Claire Nédellec, Robert Bossy, Jin-Dong Kim, Jung-Jae Kim, Tomoko Ohta, Sampo Pyysalo, and Pierre Zweigenbaum. 2013. Overview of bionlp shared task 2013. In *Proceedings of the BioNLP shared task 2013 workshop*, pages 1–7.

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

Ana-Maria Popescu, Marco Pennacchiotti, and Deepa Paranjpe. 2011. Extracting events and event descriptions from twitter. In *Proceedings of the 20th international conference companion on World wide web*, pages 105–106.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using ontonotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Jake Snell, Kevin Swersky, and Richard S Zemel. 2017. Prototypical networks for few-shot learning. *arXiv preprint arXiv:1703.05175*.

Mohammad Golam Sohrab and Makoto Miwa. 2018. Deep exhaustive model for nested named entity recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2843–2849.

Jana Straková, Milan Straka, and Jan Hajič. 2019. Neural architectures for nested ner through linearization. *arXiv preprint arXiv:1908.06926*.

Jeniya Tabassum, Mounica Maddela, Wei Xu, and Alan Ritter. 2020. Code and named entity recognition in stackoverflow. *arXiv preprint arXiv:2005.01634*.

Meihan Tong, Shuai Wang, Bin Xu, Yixin Cao, Minghui Liu, Lei Hou, and Juanzi Li. 2021. Learning from miscellaneous other-class words for few-shot named entity recognition. *arXiv preprint arXiv:2106.15167*.

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

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. Luke: deep contextualized entity representations with entity-aware self-attention. *arXiv preprint arXiv:2010.01057*.

Guanqun Yang, Shay Dineen, Zhipeng Lin, and Xueqing Liu. 2021. Few-sample named entity recognition for security vulnerability reports by fine-tuning pre-trained language models. In *International Workshop on Deployable Machine Learning for Security Defense*, pages 55–78. Springer.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. *arXiv preprint arXiv:2005.07150*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28:649–657.

Jingwei Zhuo, Yong Cao, Jun Zhu, Bo Zhang, and Zaiqing Nie. 2016. Segment-level sequence modeling using gated recursive semi-markov conditional random fields. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1413–1423.

A N way K shot dataset sampling method

In Few-Shot NER problems, the input sentence might contain several entities with different types and boundaries. Therefore there is a high possibility that the number of entities and categories do not match the N-way and the K-shot setting. For example, the selected sentence may contain the entity whose category does not belong any of the given N entity types, or the selected sentence contains the entity whose type in support dataset has enough K-shot entities. Thus, select K sentences for each type in Y^{spt} randomly to generate support dataset $\mathcal{S}^{spt}$ is not suitable for N-way K-shot NER task.

We propose a sampling method to gain $\mathcal{S}_j^{spt}$ in the following way: we only select sentences that contain the entity type in N-way and mask entities in other types until all N-way have at least K-shot entities. Algorithm 2 shows the detailed sampling method. The result is, $\mathcal{S}_j^{spt}$ will contain several sentences, and at least one type has K entities and others may have more than K entities. For example, when it comes to the 2-way 1-shot configuration, the sentence "Positron emission tomography in a case of $[\text{intracranial hemangiopericytoma}]_{\text{Cancer}}$." will be selected. But the sentence "A significant diminution of $[\text{tumor}]_{\text{Cancer}}$ size and weight was observed in the drug-treated animals." will not be selected because "Cancer" label already meet requirement of 1-shot in previous sentence. But the sentence "The presence of activating $[\text{TSH}-\text{R}]_{\text{Gene\_or\_gene\_product}}$ mutations has also been demonstrated in differentiated $[\text{thyroid carcinomas}]_{\text{Cancer}}$." will be selected because there is a "Gene_or_gene_product" label. Thus, there will be 1-shot of "Gene_or_gene_product" entity and 2-shot of 'Cancer" entity in $\mathcal{S}_j^{spt}$.

Algorithm 2: N-way K-shot Support Set Sampling

STATISTICS(Y_j) calculates the total number of labels for each type in Y_j. NUMBER(y,Y_j) means the total number of the specific label type y in Y_j.

Input: Dataset $\mathcal{D}_j = \{\mathcal{S}_j,\ \mathcal{Y}_j\}$, label type Y_j, K
Output: support set $\mathcal{D}_j^{spt} = \{\mathcal{S}_j^{spt}, \mathcal{Y}_j^{spt}\}$
$\mathcal{S}_j^{spt},\ \mathcal{Y}_j^{spt} \leftarrow \varnothing,\ \varnothing$;
while $\exists\, n \in$ STATISTICS($\mathcal{Y}_j^{spt}$), $n < K$ **do**
    types $\leftarrow l$, if NUMBER($l,\ \mathcal{Y}_j^{spt}$) $< K$, $\forall\, l \in Y_j$
    $s \leftarrow$ a sentence sampled from D_j;
    $y \leftarrow$ the corresponding label sequence;
    if $y \cap$ types $\neq \varnothing$ **then**
        $\mathcal{D}_j^{spt} \cup \{s,\ y\}$;
    end
end
return $\mathcal{D}_j^{spt}$