A Self-training Approach for Few-Shot Named Entity Recognition

Yudong Qian and Weiguo Zheng(✉)

School of Data Science, Fudan University, Shanghai, China
{20210980103,zhengweiguo}@fudan.edu.cn

Abstract. Named entity recognition (NER) is a basic task in natural language processing and can be used in a wide range of downstream tasks, such as question answering, text summarization, and machine translation. In recent years, deep-learning based methods achieve great performance in the NER task. It often demands a huge amount of data to train models. However, it is very expensive to collect sufficient training data in many real-world applications. Thus, it is important to develop NER systems for few-shot settings. In this paper, we propose a self-training approach for NER that employs the framework of the machine reading comprehension model when lacking training samples. Experimental results on NER benchmarks demonstrate that the proposed method in this paper outperforms the state-of-the-art methods.

Keywords: Named entity recognition · Few-shot learning · Semi supervised learning · Self-training

1 Introduction

Named entity recognition (NER) is an important task of natural language processing, it recognizes the predefined entity types from the input text. Early NER systems, e.g., NetOwl [1], relied on manually-defined rules. Some feature-based supervised learning methods regard the NER task as a multi-classification problem or sequence labeling problem, e.g., CRF [2]. However, traditional NER methods cannot capture the semantic information in the text, so it is difficult to improve the performance of these methods further. As deep learning methods, e.g., BiLSTM + CRF [3], have been widely applied in NER tasks, these methods can capture hidden features and exhibit better generalization ability than traditional methods.

Although deep-learning based methods have achieved great progress in NER tasks, many challenges remain to address, such as the lack of sufficient annotation data in some low-resource fields. Many NER systems have good results in general domain data sets, but they need a large amount of annotation data to train the model, and the acquisition of annotation data usually requires rich domain knowledge, as well as huge labor costs. However, high-quality annotation data is scarce in many practical scenarios. Therefore, it is of great significance to develop NER systems for few-shot settings.

B. Li et al. (Eds.): APWeb-WAIM 2022, LNCS 13422, pp. 183–191, 2023.
https://doi.org/10.1007/978-3-031-25198-6_15

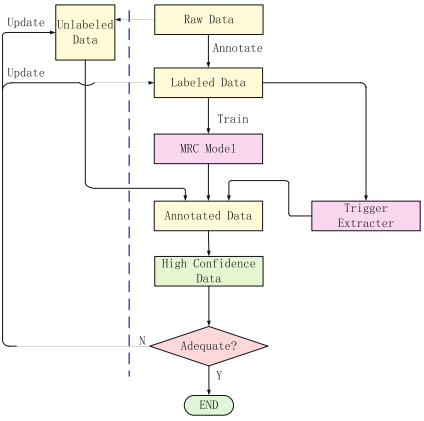

Fig. 1. Framework of our model

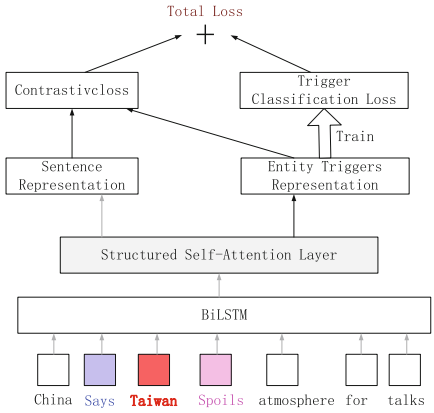

Fig. 2. Trigger representation learning

In this paper, we propose a few-shot NER model based on self-training, taking machine reading comprehension (MRC) as a built-in block. The overall structure of our model is shown in Fig. 1. Specifically, it is mainly composed of three steps: 1) The base model is trained first by using labeled data; 2) Compute the confidence of weak annotation data inferred by the trained model in the former step, and select high-confidence data to expand labeled data; 3) Iterate from step 1 to step 2 until the stop condition is achieved. The introduction of the MRC-based model can encode external knowledge about entities by setting appropriate queries, which benefits the application in few-shot settings. While the framework of self-training is adopted, we use entity triggers to compute the confidence of weak annotation data, which can mine information from different perspectives of labeled data and provide effective filtering rules to filter out noisy data. As self training has proved its effectiveness in few-shot settings, we apply a new confidence measure to the process of self-training and conduct experiments to show the effectiveness of our method.

In summary, the contributions of the paper can be summarized as follows:

- We propose a self-training based framework to recognize named entities in few-shot settings.
- We select machine reading comprehension model as the base model of our self-training framework, and the NER task is regarded as answering the corresponding queries. Besides, we compute confidence of weak labeled data based on entity triggers.
- Extensive experiments are conducted on two benchmarks to confirm the effectiveness of the proposed method.

2 Our Model

2.1 MRC-NER

We first transform the tagging-style annotation NER dataset into MRC-style. Specifically, we generate the query set $Q = \{q_{y_1}, \ldots, q_{y_k}\}$, where q_{y_i} denotes

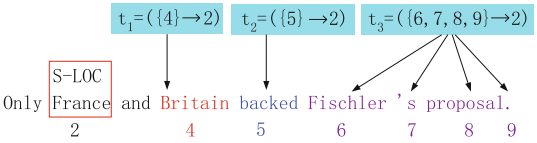

Fig. 3. Example of entity trigger

the query of entity type y_i. Then we can get corresponding answer set $A = \{a_{start_1,end_1}, \ldots, a_{start_p,end_p}\}$ of input S, where $a_{start_p,end_p} = \{w_{start}, \ldots, w_{end}\}$ denotes the corresponding entity mention. Therefore, we can get MRC-style annotation sample $(Question, Answer, Context)$. After transforming tagging-style dataset into MRC-style, we can extract the entity by answering the question of a certain type. Solving NER tasks by the MRC-based model has a key advantage against traditional methods: we can encode prior knowledge about entity categories through the query, and the specific description of similar entity categories can effectively eliminate ambiguity.

For few-shot learning, due to the limited annotation data, it is necessary to import external knowledge. Thus, we choose the MRC-based NER method [5] as the base model and improve its performance through self-training.

2.2 Entity Triggers

Entity Triggers [6] are defined as a set of words that help explain the entity recognition process in a sentence. When we recognize some entity in a sentence, we usually take certain words or phrases in the sentence as the basis for our judgment, even if it is a word we are not familiar with. In short, entity triggers can effectively help us understand the training process of the model and enable the model to summarize the information of entity categories better. This method was proposed by Lin et al. [6], and it achieved good results in few-shot settings by using labeled data with entity triggers. Fig. 3 presents such an example, where t_i denotes an entity and its corresponding trigger.

When it lacks enough annotation data, entity triggers may provide us supplementary information different from the original label information. It can be regarded as supplementary annotations in the case of insufficient annotation data, so as to help the model learn and summarize better from the limited annotation data. Therefore, we select relevant information of entity triggers as an auxiliary to compute the confidence of weak labeled data during self-training process, and it can effectively filter out noisy data and improve the performance of our model.

Trigger Extractor. Although annotating entity triggers manually may have high quality, it needs domain knowledge and high labor costs, which is not practical for NER tasks in few-shot settings. Therefore, we design a model for automatic extraction of triggers based on the AutoTrigger model proposed by Lee et al. [7]. We use SOC (Sampling and Occlusion) [8] algorithm to compute the context-independent importance of phrases, which can be used to extract

triggers. SOC is a technique for model interpretation. The expression of the importance score of the phrase p in input sequence x is:

$$\phi(p,x) = \frac{1}{|S|} \sum_{\widehat{x}_\delta \in S} \left[s\left(x_{-\delta}; \widehat{x}_\delta\right) - s\left(x_{-\{\delta,p\}}; \widehat{x}_\delta; 0_p\right) \right] \tag{1}$$

where $s(x)$ denotes the predict score of the model, $x_{-\delta}$ denotes the sequence after masking a context of length N surrounding the phrase p from input sequence x, $\widehat{x}_\delta$ denotes the sequence of length N obtained according to sampling probability distribution $p(\widehat{x}_\delta | x_{-\delta})$ based on the pre-trained language model, 0_p denotes paddings for phrase p, and S denotes a collection of samples $\widehat{x}_\delta$ from a pre-trained language model. Therefore, the importance score of phrase p can be interpreted as the expectation of difference between predict scores after masking phrase p in all possible context $\widehat{x}_\delta$ of p, which can also eliminate the relationship between the importance score and the context of the phrase.

The process of automatic trigger extraction can be simply described as follows:

1) It first trains a classifier M_t based on annotation data D_L. For the input $x = \left(x^{(1)}, x^{(2)}, \ldots, x^{(n)}\right)$, the classifier M_t uses conditional probability $P(y|x)$ to denote its output, y is the corresponding label sequence. The predict score of target entity e can be expressed as the following formula:

$$s(x,e) = \frac{1}{|e|} \sum_{x^{(j)} \in e} P\left(y^{(j)} | x^{(j)}\right) \tag{2}$$

2) Then generate the candidate trigger set P according to the set of phrase nodes from the constituency parse tree, and calculate the importance score of its target entity for each phrase $p_i \in P$:

$$\phi(p_i,x,e) = \frac{1}{|S|} \sum_{\widehat{x}_\delta \in S} \left[s\left(x_{-\delta}, e; \widehat{x}_\delta\right) - s\left(x_{-\{\delta,p_i\}}, e; \widehat{x}_\delta; 0_{p_i}\right) \right] \tag{3}$$

3) For all candidate triggers $p_i \in P$, select top-K triggers with the highest score after computing the importance score.

2.3 Self-training Framework

Trigger Representation Learning. After extracting entity triggers, we train the model to learn the representation of triggers.

First, for the annotation data with triggers, we obtain the embedding of input sentence S and trigger t according to the method proposed by Lin et al. [9], denoted as g_s and g_t respectively. g_s is the weighted sum of token embeddings in the sentence, and g_t is the weighted sum of embeddings of triggers in the sentence. Then we learn the weight matrix by training in two tasks and obtain the trigger embedding. Fig. 2 shows the framework. For the first task, we learn trigger vectors by using entity types as supervision. The second task aims at

making the trigger vector and sentence matched. The final loss is the weighted sum of the loss of these two tasks.

Confidence. In the iterative process of self-training, how to find and remove noisy data is critical. By selecting reliable weak annotation data, we can improve the quality of expanded labeled data, and then improve and model performance.

Based on trigger vectors learned in last subsection, we compute the distance $d = \|g_x - g_t\|_2$ between trigger t and weak annotation sentence x, and set the threshold λ. For the set of triggers $T_x = \left\{t_x^{(1)}, t_x^{(2)}, \ldots\right\}$ satisfying $d < \lambda$, the corresponding entity type and quantity set is $E_x = \{(e_1, n_1), (e_2, n_2), \ldots, (e_k, n_k)\}$, where e_i denotes the corresponding entity type and n_i denotes the number of triggers belong to this entity type.

For weak annotation data (x, y), the annotation entity type is e_i and its entity type and quantity set is E_x, if the following conditions are satisfied, we will regard this weak annotation data as reliable one:

$$\frac{n_i}{\sum_{j=1}^{k} n_j} \geq \theta_1 \quad or \quad n_i \geq \theta_2 \tag{4}$$

where θ_1 and θ_2 are thresholds. For the reliable weak annotation data obtained after each iteration and the previous labeled data, we define the loss function in the next iteration as follows:

$$L_{ST} = \frac{1}{|D^L|} \sum_{(x,y)\in D^L} L\left(f\left(x\right), y\right) + \frac{\lambda}{|D^U|} \sum_{(x,y)\in D^U} L\left(f\left(x\right), y\right) \tag{5}$$

where $f(\cdot)$ denotes new trained model based on D^L and D^U, and λ_U denotes weight. Self-training is carried out iteratively according to the corresponding steps until reaches the maximum number of iterations or meets stop conditions.

3 Experiments

3.1 Datasets

We use two datasets CoNLL2003 [10] and BC5CDR [11] for experiments. CoNLL2003 is an English general domain dataset, including four named entities: Location, Organization, Person, and Miscellaneous. BC5CDR is an English dataset in the biomedical field, including two named entities: Disease and Chemical. Tagging-style annotation data in two datasets are transformed into corresponding MRC-style annotation data. The queries corresponding to the entity category are obtained from the annotation guide notes.

3.2 Baselines

We select the following models as baselines:

- BiLSTM-CRF [3]: A classical sequence labeling model.

Table 1. Results on CoNLL2003, where P and R denote Precision and Recall, respectively

Per.	BiLSTM-CRF			TMN			TMN+self-training			BERT-tagger			STM		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
1%	41.88	17.62	24.81	71.43	54.27	61.68	74.13	53.53	**62.17**	31.53	30.47	30.99	52.92	62.18	57.18
3%	55.19	46.97	50.75	76.06	74.13	75.08	80.06	75.23	**77.57**	56.01	48.99	52.27	71.18	72.02	71.6
5%	70.26	53.92	61.02	80.38	79.13	79.75	80.45	81.14	**80.79**	59.46	57.74	58.59	76.49	79.52	77.98
7%	71.46	61.26	65.97	82.78	81.31	82.04	82.75	81.99	82.37	65.09	65.23	65.16	84.28	83.75	**84.01**
10%	75.41	70.43	72.83	84.55	82.43	83.48	84.55	82.59	83.56	69.18	71.88	70.5	85.28	85.16	**85.22**
13%	78.03	74.49	76.22	84.79	83.2	83.99	84.51	84.03	84.27	72.01	70.97	71.49	85.02	85.97	**85.49**
15%	79.37	76.15	77.73	85.12	83.47	84.29	86.02	83.29	84.63	73.48	73.19	73.33	84.96	86.33	**85.64**
17%	80.27	77.65	78.94	85.33	84.01	84.66	86.31	83.94	85.11	73.88	75.24	74.55	86.34	86.39	**86.36**
20%	83.11	77.2	80.05	85.5	85.64	85.57	86.32	85.47	85.89	75.26	77.14	76.19	87.23	86.51	**86.87**

- Trigger Matching Network (TMN) [6]: NER model based on manually labeled triggers.
- TMN with Self-training [6]: Self-training is adopted to TMN, and the confidence is computed based on MNLP proposed by Shen et al. [12].
- Bert-Tagger [4]: Sequence labeling model based on BERT.

3.3 Results and Analysis

Table 1 and Table 2 show the results in CoNLL2003 and BC5CDR respectively. It can be observed that, when training data is 1% of the dataset, the F1 value of BilSTM-CRF model is only 24.81%. Few training data leads to poor generalization ability of the model. Although with training samples become more, the model performance has been significantly improved a lot, there is still a big gap between BilSTM-CRF and our model (STM). The performance of Bert-Tagger model is similar to that of BilSTM-CRF. STM performs much better than BilSTM-CRF and Bert-Tagger in the case of few training samples. When compared with two TMN (+self-training) models based on trigger matching, the performance of STM is slightly poor when the training samples are less than 5%. The reason may be that when the sample size is small, the quality of extracted trigger is not high enough, and the query information imported to MRC model cannot be learned well. But when training samples reach 7% or above, the performance will be improved, and it has certain advantages when compared with TMN (+self-training). On the whole, when training samples are less than 20%, STM has a relatively good performance by importing external knowledge and mining information from limited training data. The disadvantage is that when the size of training data is too small (less than 5%), the model can not fully filter out noisy data because of the poor quality of extracted triggers, resulting in poor performance. Therefore, the model can be improved by improving the quality of extracted triggers for few-shot settings, such as transferring existing entity triggers to low-resource field.

The entity definition in the biomedical field is complex, and it's difficult to identify. Therefore, the overall model performance is much lower than that

Table 2. Results on BC5CDR

Percentage	TMN			TMN+self-training			STM		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
1%	59.01	48.78	53.41	59	49.33	**53.73**	48.2	52.03	50.04
3%	66.35	57.24	61.46	65.42	59.23	62.17	60.21	73.44	**66.17**
5%	69.37	63.29	66.19	68.14	66.89	67.51	66.9	71.36	**69.06**
7%	70.29	67.89	69.07	71.46	67.7	69.53	73.17	72.91	**73.04**
10%	72.01	69.35	70.66	69.61	72.84	71.19	75.03	75.06	**75.04**
13%	73.16	70.61	71.86	75.14	69.56	72.24	77.4	76.56	**76.98**
15%	75.04	69.11	71.95	71.38	73.41	72.38	79.37	76.27	**77.79**
17%	74.72	71.01	72.81	74.13	73.64	73.88	77.63	78.22	**77.93**
20%	74.35	72.64	73.48	75.13	73.71	74.41	79.63	79	**79.31**

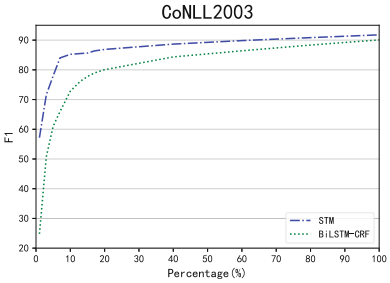

Fig. 4. Effect of varying percentage of training samples on CoNLL2003

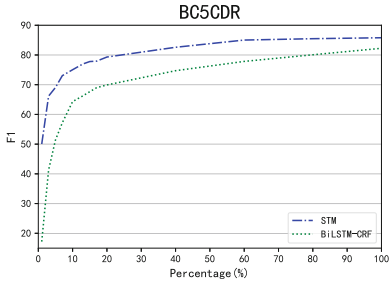

Fig. 5. Effect of varying percentage of training samples on BC5CDR

in CoNLL2003. Compared with the results in CoNLL2003, STM has a more significant advantage in BC5CDR (F1 value is about 4%–5% higher on average). The possible reason is that STM can make full use of the corresponding external knowledge for entities in the biomedical field by setting appropriate queries. In this way, the significant features of the entity category can be extracted, and the noisy data that is easily confused can be filtered out based on triggers, so the advantages are more obvious than in CoNLL2003.

Corresponding line charts are drawn for the performance of STM and BILSTM-CRF in different percentages of training data in CoNLL2003 and BC5CDR respectively, as shown in Fig. 4 and Fig. 5. It can be seen that, less training data, the greater advantage of STM compared with BiLSTM-CRF, the reason is that when the size of training data is small, it is hard for BiLSTM-CRF to learn the important features of corresponding entity category, which leads to poor generalization ability. However, the external knowledge introduced by STM and the information mined from different perspectives of limited training data lead to good generalization ability even if the size of training data is small.

The results of ablation experiments are shown in Table 3. It shows the results of STM, BERT-MRC model without self-training, and self-training based

Table 3. Ablation results of CoNLL2003

Model	Precision	Recall	F1
STM	52.92	62.18	57.18
MRC	48.53	61.76	54.35
STM without Triggers	61.94	44	51.45

MRC model without filtering out noisy data in only 1% training samples of CoNLL2003. After the introduction of entity triggers to filter out noisy data and expand training data with high-quality weak annotation data, the performance of STM (F1 value is 57.18%) improves a lot when compared with that of BERT-MRC model (F1 value is 54.35%). Without the process of filtering out noisy data, STM without Triggers only use weak annotation data to expand training data, although the size of training data has been increased, the quality falls and prediction error of the model will be accumulated, so the model performance falls when compared with F1 value of BERT-MRC model, it is reduced by about 3%. Therefore, it can be concluded that the process of filtering out noisy data by mining trigger information in training data is very important.

4 Conclusion

In this paper, we propose a self-training based NER method to improve the generalization ability of the model in the settings of few-shot. Our model uses MRC-based model as the base model and trains the model under the framework of self-training. The experimental results show that the proposed method outperforms the existing methods.

Acknowledgement. This work was supported by Science and Technology Committee Shanghai Municipality (Grant No. 19ZR1404900) and National Natural Science Foundation of China (Grant No. 61902074).

References

1. Krupka, G.R., Hausman, K.: IsoQuest Inc. 3900 Jermantown Ave., Suite 400 Fairfax, VA 22030 gkrupka@isoquest.com
2. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: ICML (2001)
3. Ma, X., Hovy, E.H.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In: ACL (1) (2016)
4. Devlin, J., Chang, M.W., Lee, K., et al.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1) (2019)
5. Li, X., Feng, J., Meng, Y., et al.: A unified MRC framework for named entity recognition. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 5849–5859 (2020)

6. Lin, B.Y., Lee, D.H., Shen, M., et al.: TriggerNER: learning with entity triggers as explanations for named entity recognition. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8503–8511 (2020)
7. Lee, D.H., Kiran Selvam, R., Sarwar, S.M., et al.: AutoTriggER: named entity recognition with auxiliary trigger extraction. arXiv e-prints arXiv: 2109.04726 (2021)
8. Jin, X., Wei, Z., Du, J., et al.: Towards hierarchical importance attribution: explaining compositional semantics for neural sequence models. In: International Conference on Learning Representations (2019)
9. Lin, Z., Feng, M., dos Santos, C.N., et al.: A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130 (2017)
10. Sang, E.F.T.K., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. Development **922**, 1341 (1837)
11. Li, J., Sun, Y., Johnson, R.J., et al.: BioCreative V CDR task corpus: a resource for chemical disease relation extraction. Database (2016)
12. Shen, Y., Yun, H., Lipton, Z.C., et al.: Deep active learning for named entity recognition. In: International Conference on Learning Representations (2018)