SRCB at SemEval-2023 Task 2: A System of Complex Named Entity Recognition with External Knowledge

Yuming Zhang, Hongyu Li, Yongwei Zhang, Shanshan Jiang, Bin Dong
Ricoh Software Research Center (Beijing) Co., Ltd.
{Yuming.Zhang1, Hongyu.Li, Yongwei.Zhang, Shanshan.Jiang, Bin.Dong}@cn.ricoh.com

Abstract

The MultiCoNER II shared task aims at detecting semantically ambiguous and complex named entities in short and low-context settings for multiple languages. The lack of context makes the recognition of ambiguous named entities challenging. To alleviate this issue, our team **SRCB** proposes an external knowledge based system, where we utilize 3 different types of external knowledge retrieved in different ways. Given an original text, our system retrieves the possible labels and the descriptions for each potential entity detected by a mention detection model. And we also retrieve a related document as extra context from Wikipedia for each original text. We concatenate the original text with the external knowledge as the input of NER models. The informative contextual representations with external knowledge significantly improve the NER performance in both Chinese and English tracks. Our system win the 3rd place in the Chinese track and the 6th place in the English track.

1 Introduction

The task of Multilingual Complex Named Entity Recognition (MultiCoNER) (Malmasi et al., 2022b; Fetahu et al., 2023b) aims to deal with the complex named entity recognition problem. Unlike the ordinary entities in most Named Entity Recognition (NER) tasks, these complex entities can be composed of any form of a language. Compared with the 6 categories defined in MultiCoNER (Malmasi et al., 2022a), MultiCoNER II (Fetahu et al., 2023a) furtherly defines 33 fine-grained categories for the complex entities and provides them with short and uncased texts (in short and low-context settings). Moreover, at the stage of test, some spelling mistakes are added into the test set, causing data disturbance. Recognizing complex named entities in such settings is challenging for NER systems.

In practice, for a professional annotator, a solution to deal with these complex entities is using external knowledge. The external knowledge includes professional knowledge of their own, information presented by search engines and contents in databases. Retrieving such related external knowledge can help to determine whether a fragment of sentence is an entity or not and eliminate the ambiguity of complex entities (Wang et al., 2019). Meanwhile, (Wang et al., 2022) proposed a general knowledge-based NER system retrieve the related documents of the input sentence as external knowledge which is proved to be very effective in short and low-context named entity recognition. Therefore, we believe that introducing external knowledge to NER systems is still a straight-forward and effective way to improve the performance of complex entity recognition.

In this paper, we propose an external knowledge based system, which is composed of a knowledge retrieval module and an NER module. In the knowledge retrieval module, given an original text, we try to detect potential entities with a high-recall mention detection model and project the detected entities to the corresponding Wikidata entities to collect entity types and entity descriptions from Wikidata. Besides, we also retrieve the most related document as extra context for the given text. We refer to the collected entity types, the entity descriptions and the retrieved extra context as **Prompt**, **Description** and **Context** respectively. And then in the NER module, during training and predicting, we concatenate the original texts with these external knowledge as the input of a range of NER models, which use different pretrained language models or different model structures. Finally, we use the model ensemble method of voting, which shows a certain improvement in both Chinese and English track.

2 Related Work

NER (Sundheim, 1995) is one of the most famous basic tasks in natural language processing, of which

Proceedings of the The 17th International Workshop on Semantic Evaluation (SemEval-2023), pages 671–678

the aim is to recognize entities from texts and classify them into artificially defined categories. Nowadays' NER methods can achieve good performance on some famous NER datasets (e.g., CoNLL 2002, CoNLL 2003, OntoNotes), such as the most popular model architecture of a pretrained language model like BERT(Devlin et al., 2019) with an additional Conditional Random Field (CRF, (Lafferty et al., 2001)) layer. However, all these methods suffer severe performance degradation when faced with MultiCoNER datasets due to the lack of contextual information and the complexity of entities, as the contextual features play a very significant role in NER (Lê, 2019). Introducing external knowledge has been proved to be effective in solving such problems. (Wang et al., 2021) uses Google search engine to retrieve external contexts for the input texts and achieves good performance. Besides introducing external knowledge from different sources such as Wikipedia, search engines and professional databases, some research proposes to utilize external knowledge in different ways. In order to help utilize word-level representation for the Chinese language, which is not naturally segmented, (Zhang and Yang, 2018; Ma et al., 2020; Li et al., 2020a) propose modifications based on the structures of LSTM or Transformers to introduce the external linguistic knowledge of lexicon. While (Wang et al., 2021, 2022) uses a simple and straightforward way of simply concatenating the input text with external knowledge of related documents as input to improve the contextual representation, which significantly improves the NER performance.

3 System Description

In this section, we introduce our external knowledge based NER system. Given an original text of n tokens $\boldsymbol{x} = \{x_1, x_2, ..., x_n\}$, the knowledge retrieval module first use a mention detection model to detect potential entities within the text as many as possible. Then it takes the mentions of potential entities or the original text as the query to retrieve three types of external knowledge from different sources: **Prompt** which is the possible labels of the potential entities retrieved from a knowledge base constructed from Wikidata; **Description** which is the concise descriptions for the potential entities retrieved from MediaWiki API; **Context** which is the related document of the original text retrieved from Wikipedia. The system concatenates the original text with these three types of external knowledge into a new readable input text, and feed it into the NER module. Although the models we implemented in our NER module differ in the output format, the outputs of these models are finally converted to $\hat{\boldsymbol{y}} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_n}\}$

3.1 Knowledge Retrieval Module

External knowledge can improve the performance on named entity recognition tasks effectively, especially for those which require recognizing complex entities in short and low-context settings. Wikipedia and Wikidata has been proved to be high-quality resources of external knowledge. Therefore, we retrieved three types of external knowledge from Wikipedia or Wikidata, and feed the concatenation of them and the original input texts into the NER module.

3.1.1 Knowledge Base Construction

Wikidata is a large-scale multilingual knowledge graph that covers over 101 million real-world entities, and provides the properties of the entities and their relations with other entities. Some of the properties directly indicate the types of the entities, e.g., (World Trade Organization-instance of-organization) indicates that World Trade Organization is one organization, and (Joseph Stalin-occupation-politician) indicates that Joseph Stalin is a politician. For those entities of which the properties don't indicate their types directly, we can derive their types by the relations of *subclass of* and *instance of* recursively, as shown in Figure 1. We aim to project each potential entity extracted by mention detection models to its true Wikidata entity, so that we can use the types of the Wikidata entity as its possible types.

Based on this, we construct a knowledge base contains the mentions of Wikidata entities and the types of them, using the Wikidata dump of version (2022-6-20) downloaded from Wikimedia[1]. Firstly, in order to project a string of potential entity to its true Wikidata entity, we collect the mentions of the Wikidata entities. The mentions include the title, the label[2] and the aliases, which respectively collected from the corresponding field of *title*, *label* and *aliases*. Secondly, we label each Wikidata entity with the 33 entity type labels defined by MultiCoNER II dataset by the following steps: (1)

[1] https://dumps.wikimedia.org/

[2] A field of Wikidata entitiy mentions. In order to distinguish this with entity type labels, we later refer to this as "entity label" in this section.

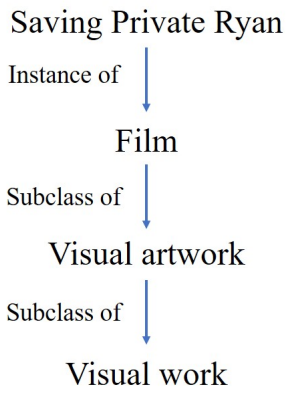

Figure 1: An example that shows the recursive way of deriving the types of Wikidata entities.

For each label, we transfer it into a word sequence in natural language. Then we collect a set of the synonyms of this word sequence including itself as the mentions of the root entities for this label[3]. (2) For each entity type label, we search for the Wikidata entities of which the title or entity label exactly matches at least one of the mentions of its root entities as root entities. (3) As the first step, for each label, we add its root entities to an empty entity collection. Then repeat the process that adding all Wikidata entities that have the direct relation of *instance of*, *subclass of* or *occupation* with the entities added to the collection at last step until no more entity is added to the collection. Then we label the entities of the collection except the root entities with the corresponding label of the root entities.

3.1.2 Mention Detection Model

The only way to project the potential entities in the original text to the corresponding Wikidata entities is to match them through entity mentions. In order to extract the mentions of the potential entities from the original text, we train a mention detection model, which can recognize entities regardless of their labels.

For this task, we use 3 methods: sequence tagging, span pointer (Li et al., 2020b) and global pointer (Su et al., 2022). The method of global pointer obtains the best performance out of the three in both Chinese and English track as Table 1 shows, so we finally choose the method of global pointer to build our mention detection models. This

	Chinese	English
Sequence tagging	90.82	86.13
Span pointer	89.45	86.01
Global pointer	**98.91**	**98.10**

Table 1: Mention Detection Model performance

method is to train a classifier to classify all possible spans within the original texts as positive (real entities) or negative. In the training phrase, we use the span of real entities as positive samples, while use k[4] spans which are not real entities randomly selected from the original texts as negative samples. In the predicting phase, we collect all possible spans within a given text, and use the classifier to classify each span as positive or negative.

It is worth mentioning that we need to recall real entities as many as possible, so we use the score of recall as the metrics of our mention detection models. The reason is that a missing prediction influences much more than an incorrect prediction. Specifically, for those spans within the original text that are incorrectly predicted as entities, it could be filtered by our knowledge base during matching of entity mentions or could be ignored by our named entity recognition module. However, a missing prediction means the loss of external knowledge for a real entity, which may cause a missing prediction or incorrect prediction that leads to degradation of the final performance of our system.

3.1.3 Knowledge Retrieval

Prompt We use ElasticSearch (ES)[5] to search the best matches of Wikidata entities for each potential entity string extracted by the mention detection models. For each of such potential entity strings, we query the *title*, *entity label* and *aliases* field of our database to get the top-k[6] results. Title and entity label are considered to be more formal than aliases, so we add a boost on *title* and *entity label* fields as three times large as *aliases* field. Finally, we collect all labels of retrieved top-k entities as the possible labels of the potential entity string. We refer to the retrieved possible labels as **Prompt**.

Description Usually, there is a concise English description for a Wikidata entity on its Wikidata page, which can provide more additional information that helps the recognition of the entity type.

[3]Here, a root type entity refers to the Wikidata entity corresponding with one entity type label. Note that one entity type label may be corresponding with several root entities.

[4]k=3 in our system

[5]https://www.elastic.co/

[6]k=3 in our case.

We use MediaWiki API[7] to access the descriptions from Wikidata for each candidate entity that matches at least one Wikidata entity in our knowledge base. Moreover, for the Chinese track, we used Google Translate[8] to convert the English descriptions into Chinese. We refer to the retrieved descriptions as **Description**.

Context We use almost the same way introduced by (Wang et al., 2022) to get external relevant context from Wikipedia[9], but we simplify most of the processes. We use the original text as the query to retrieve the most similar sentence from Wikipedia passages, and use the paragraph which the most similar sentence belongs to as the final retrieved context. We only retrieve contexts for English track due to time limitation and we assume the context retrieved for the translated Chinese input texts would not be so effective as that of English track. We refer to the retrieved contexts as **Context**.

3.2 Named Entity Recognition Module

In the NER module, we implemented different models in the Chinese and English track. All these models receive the concatenation of the original text and the three types of external knowledge as input. One thing to be mentioned is that we process this concatenation into a readable token sequence to ensure the external knowledge can be understood by our models. Figure 2 shows an example of the processed concatenation. In this figure, we place **Context** after the original text because it's the most similar with the original text. For **Prompt** and **Description**, we construct a template of "**<Mention>** could be **<Prompt** or **Description>**", where **Mention** is a potential entity string that detected by the mention detection model, and get a readable sequence of natural language by filling the corresponding contents. We also use the special tokens[10] of different pretrained models to indicate which component the following tokens belong to.

In the Chinese track, we mainly use three structures that regard NER as a sequence tagging task: a) Normal, through pretrained language model, obtaining the representation of each word in the original text. And then classify each representation. b) EntLM (Ma et al., 2022), in this structure, we use spatial mapping to map all entity categories to embeddings with the same size as the hidden layer. The goal of the task is no longer to fit the one-hot vectors of the entity categories, but to fit the embeddings of the entity categories, which is very similar to the Mask Language Model (MLM) task in the pretraining for BERT-like models. The calculation is as follow:

$$P_{pred} = E_i * E_{label} \ / \ d$$

where E_i is the embedding of a token in the original text, obtained from the pretrained language model $BERT$. E_{label} refers to the projected label embeddings, d is their dimension.
c) LEAR (Yang et al., 2021), in this structure, we first manually describe each entity category with a series of annotations $(ann_1, ann_2, ..., ann_n)$ and integrate the original text $X = (x_1, x_2, ..., x_n)$ with these entity category annotations by attention mechanism, which introduces the meaning of the entity categories to the models. The calculation is as follow:

$$Q = w_1 * E_X$$
$$K = w_2 * E_{ann}$$
$$V = w_3 * E_{ann}$$
$$attention = softmax(QK/dK) * V$$

where E_X, E_{ann} denote the embeddings of original text and label annotation, and w is the weight. Then introduce the *attention* into the model structure to recognize the entities.

In addition to the sequence tagging methods, we also regard the NER task as a machine reading comprehension (MRC, (Li et al., 2020b)) task. Given a query built from a certain label, the aim of the models is to extract the correct spans of entities that belongs to the label from the original text. In our case, we build the query as "What are the xxx (i.e., LOC -> location) entities in this sentence?" and feed it as long as the original text to Ernie (Sun et al., 2020) to get the token embeddings E_X. Then two binary classifiers are used to determine whether a token is the beginning or the end of an entity respectively from its token embedding . The calculation is as follows:

$$P_{start} = softmax(W_{start} E_X)$$
$$P_{end} = softmax(W_{end} E_X)$$

where W_{start}, W_{start} are weights.

In a flat NER dataset, the heuristic of matching the start index with its nearest end index works for

[7] https://www.wikidata.org/w/api.php
[8] https://translate.google.com
[9] We used the Wikipedia dump of version (2022-06-20)
[10] **[SEP]** for models based on BERT-like pretrained models

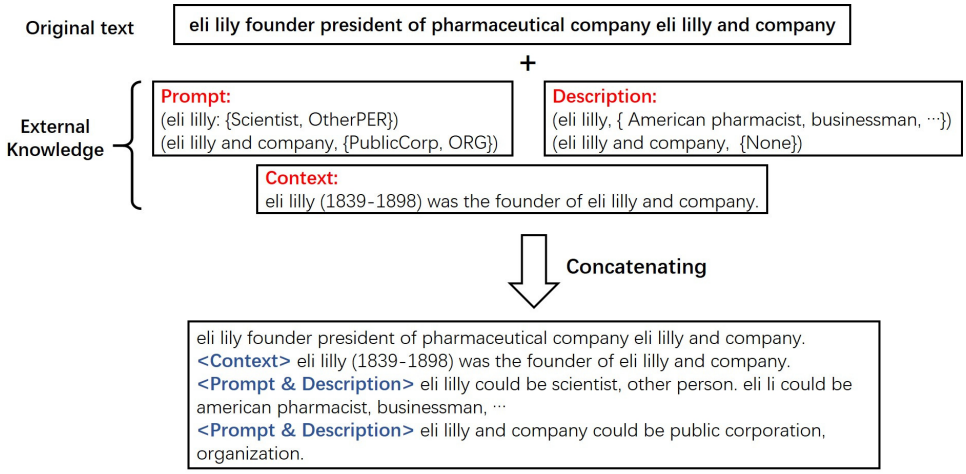

Figure 2: An example that illustrates the input of the NER module

the determination of entity spans, while does not work for nested ones since entities could overlap with each other. So in addition to the modeling of P_{start} and P_{end}, this method also applies sigmoid function to score how much a span that starts from a predicted start index i_{start} and ends at a predicted end index i_{end} is likely to be an entity or not as follows:

$$P_{i_{start},j_{end}} = sigmoid(W * concat(E_{i_{start}}, E_{j_{end}}))$$

where $E_{i_{start}}$, $E_{j_{end}}$ are the representations of the start and end indexes to be matched, and W represents the weights to be learned. This allows each predicted start index to match with multiple end indexes at the same time.

Let $P_{start,end}$ denotes the probability of match between start indexes and end indexes. During training, the loss is calculated as follows:

$$\zeta_{start} = crossentropy(P_{start}, Y_{start})$$

$$\zeta_{end} = crossentropy(P_{end}, Y_{end})$$

$$\zeta_{span} = crossentropy(P_{start,end}, Y_{start,end})$$

$$\zeta = \alpha\zeta_{start} + \beta\zeta_{end} + \gamma\zeta_{span}$$

where α, β, γ are hyperparameters.

In the English track, we use MRC method as well as Universal Information Extraction (UIE) model(Lu et al., 2022). UIE models are pretrained to be more specialized in information extraction (IE) tasks including NER. UIE used a unified text-to-structure generation framework, which project IE tasks such as NER to record generation tasks. The original outputs of UIE are token sequences of entities without the concrete offsets within the original input text. We use the results of mention detection model to help align these token sequences with spans of the original text, which shows a smaller error rate compared with the alignment strategy proposed in the paper.

We trained a range of models with different model structures or different pretrained language models. At the test stage, we mainly used the model ensemble methods of **majority voting** and **random voting** to improve performance. Majority voting means all models are used as the candidate models in ensemble. And as the final result, it will pick the label which the most number of models agree with for each prediction. While random voting means each time we randomly select a random number of models as one candidate combination and choose the combination that reaches the best evaluation result through multiple experiments. This works because not all models can contribute to the true value, and sometimes the inconsistency between models drives the result away from the true value.

4 Experiments

4.1 Data Introduction

In the Chinese track, there are 9,759 samples in the training set and 506 samples in the development set, and the maximum length of text is 106. In the English track, there are 16,778 samples in the training set and 871 samples in the development set, and the maximum length of text is 69. The maximum length of data is not very long in both Chinese and English tracks, which means the texts Concatenating external is feasible considering the maximum length and predicting cost for our mod-

Language	Methods	F1
Chinese	baseline	72.94
	Sequence tagging	88.72
	MRC	89.25
English	baseline	67.42
	Sequence tagging	77.25
	MRC	78.49
	UIE	81.68

Table 2: The results of different methods on the development set. The baseline models are sequence tagging models trained without using external knowledge.

els. Besides, the label distribution of the training set and the development set is very close.

4.2 Training Details

For the sequence tagging models, we compared different model structures and different Chinese pre-trained models downloaded from Huggingface[11], such as BERT[12], RoBERTa[13] and ERNIE[14]. We set a learning rate of 1e-5 in the encoder part based on the above pre-trained models, and a learning rate of 5e-5 in the decoder part (fully connected layers). Besides, we set the batch size to 4, 8, 16, 32 and so on.

For MRC, the pre-trained model has greatly improved the results, so here we try different pre-trained models as candidates for model ensemble. We trained each model for 30 epochs with the learning rates of 1e-5 and the max sequence length of 512.

For the UIE models, we trained each model for 50 epochs with the learning rates of 1e-4, 3e-4, 5e-5 and the batch size of 64. Finally, we keep the checkpoints which reach the best macro F1 score on the development set.

5 Results

At the test stage, there are some spelling mistakes and typos added to the test set, which causes the knowledge retrieval module fails to retrieve external knowledge for some of the potential entities due to the matching problem.To deal with this problem, we use the predictions of the models which are trained without using external knowledge to supplement the final predictions on the tokens which are predicted as 'O' by the ensemble models.

We won the 3rd place in the Chinese track and the 6th place in the English track. For the test results of the Chinese and the English track, our system obtains overall macro F1 of 75.86 and 75.62 respectively. Although our system shows good performance on the clean subset, our system also suffers a great performance degradation on the noisy subset compared with that on the clean set, which shows the impact of spelling mistakes and typos is huge for our system.

6 Analysis

To evaluate the quality and coverage of **Prompt** for train, validate and test set retrieved from our knowledge base, we define a special recall which counts a TP if the correct label is one of the retrieved labels of a real entity. The evaluation results of the training set and the development set in the Chinese track are 92.27 and 93.40. For the English track, the numbers are 86.45 and 84.65. From the evaluation results in the Chinese track and the English track, we can observe that the performance in the English track falls significantly compared with that in the Chinese track. This is caused by the fact that there are more potential entity string fail to match any of the Wikidata entities due to the absence of certain mention of it. And due to the noise of spelling mistakes and typos in the test set, we observe a decrease in the number of potential entities for which at least 1 label has been retrieved. **Description** also suffers a decrease in the number of potential entities for which at least 1 description has been retrieved. We didn't prepare fuzzy match for both English and Chinese language due to the limit of the time. Less **Prompt** and **Description** cause performance degradation of final NER performance on the test set, compared with that on the development set. Fortunately, we applied fuzzy match to the retrieval of **Context**, so that it is not influenced much as we observed.

The reason for performance degradation on the test set is not only caused by the quality reduction of external knowledge. We expected that the spelling mistakes and typos only appear in the entity tokens, while they also appear in the tokens other than entity tokens. This makes a negative impact on the contextual representation, which causes the performance of NER models drop regardless of

[11] https://huggingface.co/

[12] https://huggingface.co/yechen/bert-large-chinese

[13] https://huggingface.co/hfl/chinese-roberta-wwm-ext-large

[14] https://huggingface.co/nghuyong/ernie-3.0-xbase-zh

the external knowledge.

Comparing to majority voting, the method of random voting obtain 1.06 improvement on the development set and obtains 0.26 improvement on the test set. This method is not so effective on the test set as it does on the development set, and we assume the reason is that fitting the development set by random voting may lead to the loss of generalization ability on the test set.

7 Conclusion

In conclusion, we describe our external knowledge based system which utilizes 3 different types of external knowledge for the MultiCoNER II task. And we use different model structures, methods and pretained language models for the NER task. Our models benefits from the external knowledge and obtain great improvement on the development set. However, they also show the weakness of relying too much on the external knowledge from the performance degradation due to the quality decline of external knowledge retrieval on the test set. The result of model ensemble shows the methods of **random voting** is effective both on the development set and the test set. For future work, we plan to improve the performance of our knowledge retrieval module when faced with noise of spelling mistakes and typos, and explore different ways of external knowledge utilization that involve the modification of model structures. Moreover, we will concentrate more on the importance of contextual information for NER models, in order to reduce the dependence on external knowledge.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Besnik Fetahu, Zhiyu Chen, Sudipta Kar, Oleg Rokhlenko, and Shervin Malmasi. 2023a. MultiCoNER v2: a Large Multilingual dataset for Fine-grained and Noisy Named Entity Recognition.

Besnik Fetahu, Sudipta Kar, Zhiyu Chen, Oleg Rokhlenko, and Shervin Malmasi. 2023b. SemEval-2023 Task 2: Fine-grained Multilingual Named Entity Recognition (MultiCoNER 2). In *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*.

Xiaonan Li, Hang Yan, Xipeng Qiu, and Xuan-Jing Huang. 2020a. Flat: Chinese ner using flat-lattice transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6836–6842.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

TA Lê. 2019. A deep neural network model for the task of named entity recognition.

Ruotian Ma, Minlong Peng, Qi Zhang, Zhongyu Wei, and Xuan-Jing Huang. 2020. Simplify the usage of lexicon in chinese ner. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5951–5960.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. Template-free prompt tuning for few-shot NER. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5721–5732, Seattle, United States. Association for Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. MultiCoNER: A large-scale multilingual dataset for complex named entity recognition. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3798–3809, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. SemEval-2022 task 11: Multilingual complex named entity recognition (MultiCoNER). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1412–1437, Seattle, United States. Association for Computational Linguistics.

Jianlin Su, Ahmed Murtadha, Shengfeng Pan, Jing Hou, Jun Sun, Wanwei Huang, Bo Wen, and Yunfeng Liu. 2022. Global pointer: Novel efficient span-based approach for named entity recognition. *arXiv preprint arXiv:2208.03054*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8968–8975.

Beth M. Sundheim. 1995. Named entity task definition, version 2.1.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. Improving named entity recognition by external context retrieving and cooperative learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1800–1812, Online. Association for Computational Linguistics.

Xinyu Wang, Yongliang Shen, Jiong Cai, Tao Wang, Xiaobin Wang, Pengjun Xie, Fei Huang, Weiming Lu, Yueting Zhuang, Kewei Tu, et al. 2022. Damonlp at semeval-2022 task 11: A knowledge-based system for multilingual named entity recognition. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1457–1468.

Zihan Wang, Jingbo Shang, Liyuan Liu, Lihao Lu, Jiacheng Liu, and Jiawei Han. 2019. CrossWeigh: Training named entity tagger from imperfect annotations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5154–5163, Hong Kong, China. Association for Computational Linguistics.

Pan Yang, Xin Cong, Zhenyu Sun, and Xingwu Liu. 2021. Enhanced language representation with label knowledge for span extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4623–4635, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yue Zhang and Jie Yang. 2018. Chinese NER using lattice LSTM. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1554–1564, Melbourne, Australia. Association for Computational Linguistics.