EiCi: A New Method of Dynamic Embedding Incorporating Contextual Information in Chinese NER

Anonymous ACL submission

Abstract

With the continuous development of deep learning technology, the field of Named Entity Recognition(NER) has made great achievements in recent years. In Chinese NER, making full use of word information is becoming the key to improve model performance. In the previous related work, lexicon was applied to add word information. However, the word vectors generated by that way is static. It means that it cannot accurately describe some polysemous words in a specific context, which will affect the performance of the NER task. This paper presents EiCi to solve this problem. The new method is proposed that, without relying on external pre-trained word vectors, it takes the advantage of the pre-trained language model BERT to extract polysemous word information. In order to further utilize the word information, a sub-module for type recognition is also added to assist the main task of NER. Experiments on two main Chinese NER datasets show EiCi has better performance than the traditional NER models and other NER models that use word information.Source codes of this paper are available on Github[1]

1 Introduction

Named entity recognition (NER) aims to identify the required entities from the input text. It can classify the entity category and position of each input character. The types of these entities are defined according to the domain of the text. Not only can NER be used as a separate task, but it also plays an important role in various natural language processing applications. such as information retrieval(Guo et al., 2009; Petkova and Croft, 2007), question answering(Mollá et al., 2006), text understanding(Zhang et al., 2019; Cheng and Erk, 2020), and knowledge base construction(Etzioni et al., 2005) etc.

Compared with the clear boundaries between words and characters in English, there is uncertainty in Chinese word segmentation. Because of this advantage, the English NER can easily extract the information of words and characters. For example, we can take the whole word "English" as the word information, and use the letters in it as character information. In Chinese NER, we have relatively little use of word information(He and Wang, 2008; Liu et al., 2019).

In addition, pre-trained models, such as Bert(Devlin et al., 2019), have brought long lost vitality to the research of the entire natural language processing field. We usually used Word2Vec(Mikolov et al., 2013) or Glove(Pennington et al., 2014) as the word embedding vector generation method. These methods all learn from a large amount of corpus and then generate a multi-dimensional vector that can represent a character or a word. Obviously, the vectors generated in this way is static and immutable. This brings up the problem of ambiguity. A word may have different meanings in different contexts. In this case, it is not appropriate to use the same vector to represent it. The pre-trained language model can solve this type of problem specifically. It can obtain the context information of the word, accurately capture the specific meaning, and dynamically generate a vector that can match the current context to represent the word.

It has been verified that adding word information to Chinese NER can significantly improve the performance(Zhang and Yang, 2018). In order to make better use of word information, we added a type recognition module to the regular NER task. It is found in experiments that type recognition has a relatively large correlation with word-level information. The entities in the Chinese datasets often have a high coincidence rate with the words obtained after word segmentation. So we added the word-level information to the type recognition

[1] Our code implementation. http://github.com/*/

module and let it promotes the effect of regular NER task.

EiCi, a new method of dynamic embedding incorporating contextual information is proposed. Different from the previous methods, it first obtains the words by segmenting the sentence, and then put them into the pre-trained language model ***Bert*** to get the word vectors. Then concatenate the word vector with the corresponding character vector to obtain a vector containing word information. Finally put them into a sub-module for type recognition. The fused word information can help improve the accuracy of type recognition. This can make the main task of NER perform better. On the one hand, compared to the conventional NER models, we incorporate word information in the input layer of the model; on the other hand, compared to the previous methods of incorporating word information, our method is more efficient and can bring better results.

The main contribution of this work can be summarized as follows:

- We propose a new method of generating word vectors. Without pre-trained vectors, we rely on pre-trained language models to dynamically generate corresponding word vectors during the training process.
- In order to make better use of word information, we use a type recognition module to predict the type of the entities and play an auxiliary role for the main NER task.

We conduct experiments on two public Chinese NER datasets: Resume and MSRA. The results of the experiments show that compared with both conventional NER models and other models that use word information, our model performs better.

2 Related Work

In this section, some related work will be introduced.

2.1 BERT-Based Character Embedding

The Bidirectional Encoder Representations from Transformers (BERT)(Devlin et al., 2019) was proposed in 2018. It is a fine-tuning based pre-trained representation model that uses a Bidirectional Transformers model and multi-head attention mechanism(Vaswani et al., 2017). There are two steps in its framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Before BERT, there were pre-trained language models such as ELMO(Peters et al., 2018) and GPT(Radford et al., 2018), but BERT combined their advantages to achieve better results.

Input embedding is one of the most important aspects of natural language processing. Its main function is to convert our natural language into vectors that the computer can recognize and calculate Therefore, the quality of the word vectors can often affect the final result of NLP tasks. The most famous method of word embedding before is Word2Vec(Mikolov et al., 2013). The way it generates the vectorized representation of a word is Continuous Bag of Words(CBOW) and Skip-gram. The disadvantage of Word2Vec is that it cannot distinguish the different meanings of polysemous words.

By using two unsupervised tasks: Masked Language Model(MLM), which means masking some input tokens at random and predicting the masked tokens to train a deep bidirectional representation, and Next Sentence Prediction(NSP), which is trained to capture the relationship between two sentences, BERT can full account of context when generating embeddings.

2.2 Using Word Information in Chinese NER

In recent years, the use of word information in Chinese NER has gradually become an effective way to improve NER results. The first to significant improvement is Lattice-LSTM(Zhang and Yang, 2018). Lattice-LSTM designs to incorporate word information into the character-based neural NER model. However, in order to model the graph-based input, Lattice-LSTM has a very complicated structure. So its efficiency in training is relatively low. Another work is SoftLexicon(Ma et al., 2020). It is an improvement based on Lattice-LSTM. It mainly solves the problem that the structure of Lattice-LSTM is too complicated. It incorporates word information by adjusting in the character representation layer. It is worth noting that it used the same pre-trained word vectors as Lattice-LSTM. For each character in the input text, it will look for the word containing the corresponding character in the lexicon, and then will find the word in four cat-

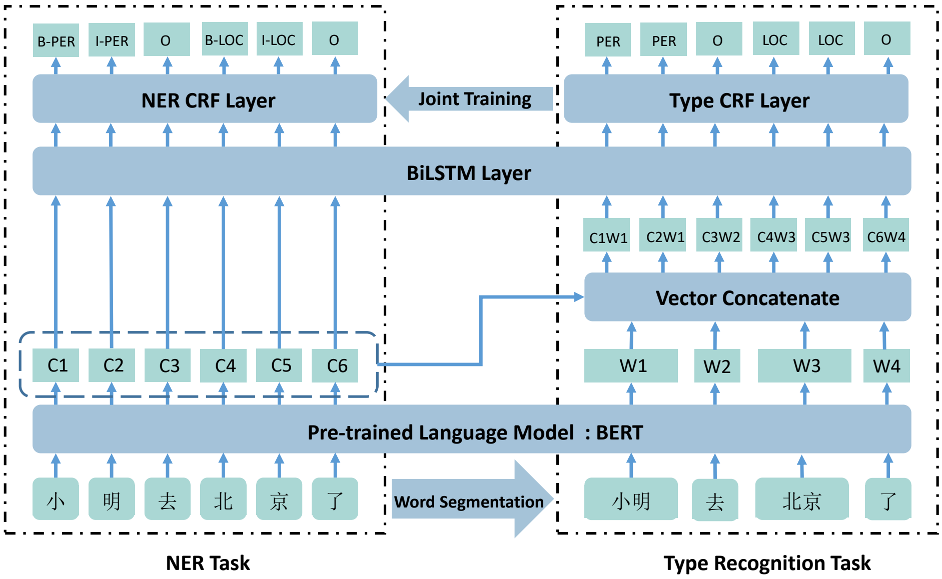

Figure 1: The overall architecture of EiCi.

egories according to the position of the character in the word. Finally, the character vector and the word vector are concatenate into the model for training, and the word information is incorporated in this way. LEBERT(Liu et al., 2021), the latest work on the use of lexicon in Chinese NER. It obtains LEBERT by directly incorporating word information into BERT, and then uses LEBERT as a new pre-trained language model for training.

3 EiCi: Dynamic Embedding Incorporating Contextual Information

In this paper, we proposed a new method to incorporate word information in the input layer. In order to make full use of word information, a submodule is designed for type recognition to assist in improving the results of NER tasks. This new model is called EiCi. EiCi is a multi-task training model(Vandenhende et al., 2020). The overall architecture of the proposed model is shown in Figure 1.

3.1 NER Task

In NER module, we use BERT-BiLSTM-CRF as the main framework. We will introduce these components individually.

3.1.1 Character-Level Representation

For the input sentence:

$$S =< c_1, c_2, \cdots, c_n > . \tag{1}$$

$c_i (1 \leq i \leq n)$ represents the character in position i in the sentence. We need to use a character-level vector $e_i^c (1 \leq i \leq n)$ to represent it. Transform the sentence into the representation as follows:

$$S =< e_1^c, e_2^c, \cdots, e_n^c > . \tag{2}$$

c means it belongs to a character-level vector. We use BERT to obtain character-level vectors.

$$e_i^c = bert(c_i)(1 \leq i \leq n). \tag{3}$$

3.1.2 BiLSTM Encoder Layer

Character-level vectorized representation of the sentence obtained in the previous stage $S =< e_1^c, e_2^c, \cdots, e_n^c >$. We will put into the encoding layer to further extract the inner relationship between the characters. Because NER is a sequence labeling problem, RNN(Huang et al., 2015) is generally used for encoding. However, RNN will bring gradient explosion and gradient disappearance, as well as problem of long-distance dependence, so we choose BiLSTM(Huang et al., 2015) for encoding. We will get the hidden sequences $H =< h_1, h_2, \cdots, h_n >$ of all characters as follows:

$$h_i = [(h_i)_{forward}; (h_i)_{backward}]. \tag{4}$$

$$\begin{aligned} (h_i)_{forward} &= LSTM(e_i^c, (h_{i-1})_{forward}). \\ (h_i)_{backward} &= LSTM(e_i^c, (h_{i-1})_{backward}). \end{aligned} \tag{5}$$

In order to better extract the relevant information of the context, we adopt a bidirectional LSTM model. It includes a forward LSTM and a backward LSTM. The hidden layer output of each character is composed of two parts. Compared with traditional RNN, LSTM has more complicated information transfer mechanism. When traditional RNN transfers information in a sequence, it will input all the information of the previous state, while LSTM preforms selective input and selective forgetting. The above h_i refers to the hidden layer output of the current character, and the h_{i-1} refers to the selective reserved content of the previous state.

3.1.3 CRF Layer

Conditional random fields(CRF)(Lafferty et al., 2001) can usually be used as a decoding layer for sequence labeling tasks. In previous work, LSTM and CRF were used as a combination to complete the task of entity recognition(Chiu and Nichols, 2016; Lample et al., 2016). Its principle can be described as follows:

$$p(y|s;\theta) = \frac{\prod_{t=1}^{n} \phi_t(y_{t-1}, y_t|s)}{\sum_{y' \in \mathcal{Y}_s} \prod_{t=1}^{n} \phi_t(y'_{t-1}, y'_t|s)}. \quad (6)$$

The advantage of CRF is that it can consider the rationality of the sequence labeling results output by the model. For example, the label that appears after "O" will definitely not be "I-LOC", because "I-LOC" must exist after "B-LOC". Here $\mathcal{Y}_s$ denotes all possible label sequences of s, and

$$\phi_t(y^{'}, y|s) = \exp(\mathrm{w}_{\mathrm{y}^{'},\mathrm{y}}^{\mathrm{T}} \mathrm{h}_\mathrm{t} + \mathrm{b}_{\mathrm{y}^{'},\mathrm{y}}) \quad (7)$$

where $w_{y^{'},y}$ and $b_{y^{'},y}$ are trainable parameters corresponding to the label pair $(y^{'}, y)$, and θ denotes model parameters. For label inference, it searches for the label sequence y^* with the highest conditional probability given the input sequence s:

$$y^* = \arg_y \max p(y|s;\theta). \quad (8)$$

which can be efficiently solved using the Viterbi algorithm(Forney, 1973).

3.2 Type Recognition Task

In order to make better use of the word information, a type recognition module is designed to complete the type classification task of input tokens. This sub-module helps to improve the effectiveness of the NER task.

3.2.1 Representation with Word Information

In this part, we will first introduce the concept of Chinese word segmentation, and then introduce the method of adding word information in the presentation layer. Finally, the reason for adding word information is discussed.

Chinese Word Segmentation

Chinese Word Segmentation(CWS) is a fundamental task for Chinese natural language processing. It aims at identifying word boundaries in a sentence composed of continuous Chinese characters. Generally, previous studies model the CWS task as a character-based sequence labeling task(Xue, 2003; Chen et al., 2015). Recently, pre-trained models such as BERT have been introduced in CWS tasks, which could provide prior semantic knowledge and boost the performance of CWS systems. In recent years, Chinese word segmentation models that use contextual information(Tian et al., 2020) and global character association(Tian et al., 2021) have achieved great performance on various datasets.

In this paper, we use the word segmentation tools such as ***jieba*** to segment the words contained in the sentence. All the words used are directly derived from the text input itself. This also strengthens the relationship between the obtained word vectors and the sentence. For example, in the following text sequence $S =< c_1, c_2, c_3, c_4, c_5, c_6 >$. After using the word segmentation tool, we got $S_w =< w_1(c_1, c_2), w_2(c_3), w_3(c_4, c_5), w_4(c_6) >$. $w_1(c_1, c_2)$ means the word w_1 is composed of c_1 and c_2.

Word-Level Representation

The words obtained from the word segmentation results will be put into the pre-trained language model BERT to generate the corresponding word vectors. Convert the sentence into a sequence of vectors in units of words as follows:

$$S_w =< e_1^w, e_2^w, \cdots, e_n^w > . \quad (9)$$

$$e_n^w = bert(w_n). \quad (10)$$

w means it belongs to a word-level vector.

Incorporating Word Information

The next step is to concatenate the obtained word-level vector with the origin character-level vector. The words and characters often have a one-to-many relationship, we concatenate the word-level vector and each character-level vector of the characters that make up it. Eventually, we get the following sequence:

$$S_{cw} =< (cw)_1, (cw)_2, \cdots, (cw)_6 > \tag{11}$$

$$\begin{aligned}
(cw)_1 &= c_1 \oplus w_1 \\
(cw)_2 &= c_2 \oplus w_1 \\
(cw)_3 &= c_3 \oplus w_2 \\
(cw)_4 &= c_4 \oplus w_3 \\
(cw)_5 &= c_5 \oplus w_3 \\
(cw)_6 &= c_6 \oplus w_4
\end{aligned} \tag{12}$$

Word Segmentation and Entity Type

In Chinese, an entity sequence is very likely to be a word segmentation sequence after word segmentation processing. This is an important consideration when we design the structure of the model. Let's take a simple example: “小明去北京了。” The word segmentation result obtained after using the word segmentation is: “小明”, “去”, “北京”, “了”. The correct entity type recognition result for this sentence is as follows: PER, PER, O, LOC, LOC, O. It is obvious that the result after word segmentation and the result after entity type recognition are completely matched. In order to better learn this kind of word-level association. The vectors that contain word information are put into the type recognition module for training.

We make statistics on the Chinese entity recognition dataset ***Resume***(Zhang and Yang, 2018) to better illustrate the relationship between word segmentation results and entity types. In a total of 3821 sentences, the number of entities and the number of words after word segmentation are respectively counted. Count the number of completely matched and partially matched. Completely matched means that an entity sequence can correspond to a word segmentation sequence. Partially matched means that an entity sequence can be composed of multiple word segmentation sequences. Table 1 shows that among a total of 9376 entities, there are 4836 that can be completely matched and 9358 are partially matched(including complete matched). It can be seen that the relationship between the word segmentation result and the entity type is very close.

3.2.2 LSTM Layer and CRF Layer

For the LSTM layer and the CRF layer in the type recognition task, we use the same network structure

Completely Matched Number	4836 (51.6%)
Partially Matched Number	9358 (99.8%)
Total Entity Number	9376

Table 1: The match between the word segmentation result and the entity (shown in brackets is the percentage of **matched number** to the **total number** of entities).

as in the NER task. We put the input $S_{cw} =< cw_1, cw_2, cw_3, cw_4, cw_5, cw_6 >$, that incorporates the word information into BiLSTM. Then use CRF as the decoding layer.

3.3 Joint Training

There are two modules in our proposed model: NER Module and Type Recognition Module. We can train the whole model with multitask training. We minimize the negative log-probability of the correct sequence of labels for the NER Module and the Type Module:

$$\begin{aligned}
\mathcal{L}^{NER} &= -log(p(\hat{y}^{NER}|X)) \\
\mathcal{L}^{Type} &= -log(p(\hat{y}^{Type}|X))
\end{aligned} \tag{13}$$

where X represents input sequence, and $\hat{y}^{NER}$ and $\hat{y}^{Type}$ represent the correct sequence of labels of the NER Module and Type Module respectively. Then, the final multitask loss is a weighted sum of the two losses:

$$\mathcal{L} = \mathcal{L}^{NER} + \mathcal{L}^{Type} \tag{14}$$

4 Experiments

In this section, the datasets, model implementation details, experimental results and ablation experiments will be introduced.

4.1 Datasets

The model was evaluated on two Chinese NER datasets, including MSRA(Levow, 2006) and Resume NER(Zhang and Yang, 2018).

- MSRA - MSRA is an entity recognition dataset in the news field marked by Microsoft Research Asia, and it is also one of the entity recognition task datasets of SIGNAN backoff 2006. The dataset contains more than 50000 pieces of Chinese entity recognition and annotation data. The entity types are divided into three categories: Person, Location and Organization.

Dataset	Type	Train	Dev	Test
MSRA	sentence	45k	3.4k	3.4k
	char	2171.5k	172.6k	172.6k
ResumeNER	sentence	3.8k	0.46k	0.48k
	char	124.1k	13.9k	15.1k

Table 2: Statistics of datasets

Models	P	R	F1
Chen et al. (2006)	91.22	81.71	86.20
Zhang et al. (2006)	92.20	90.18	91.18
Zhou et al. (2013)	91.86	88.75	90.28
Dong et al. (2016)	91.28	90.62	90.95
LSTM+CRF	90.74	86.96	88.81
BERT+CRF	93.40	94.12	93.76
BERT+LSTM+CRF	95.06	94.61	94.83
Zhang and Yang (2018)	93.57	92.79	93.18
Ma et al. (2020)	94.73	93.40	94.06
BERT+Ma et al. (2020)	95.75	95.10	95.42
Xuan et al. (2020)	-	-	94.35
Liu et al. (2021)	-	-	95.70
EiCi(ours)	**96.24**	**95.50**	**95.87**

Table 3: Performance on MSRA

- Resume - Resume NER is generated based on the resume summary data of senior managers of listed companies. The data contained 1027 resume abstracts. The entity annotations are divided into 8 categories, including Name, Location, Citizenship, Race, Professional, Education, Organization and Title.

Table 2 shows the relevant statistical data of the datasets.

4.2 Baseline Models

We will compare EiCi with some conventional NER models and some models that use word information.

Compared models include the best statistical models on these dataset, which leveraged rich handcrafted features(Chen et al., 2006; Zhang et al., 2006; Zhou et al., 2013), character embedding features(Lu et al., 2016; Peng and Dredze, 2016), radical features(Dong et al., 2016), cross-domain data, and semi-supervised data(He and Sun, 2017). Models using word information: Lattice-LSTM(Zhang and Yang, 2018), SoftLexicon(Ma et al., 2020), FLAT(Xuan et al., 2020) and LEBERT(Liu et al., 2021) are also added for comparison.

4.3 Implementation Details

EiCi is constructed based on BERT_{BASE}(Devlin et al., 2019), with 12 layers of transformers is initialized using the Chinese-BERT checkpoint from huggingface. For the output of the presentation layer, the dimension of the character-level vector is 768, and the dimension of the word-level vector is 128. The output dimension of the encoding layer LSTM is 64.

Models	P	R	F1
LSTM+CRF	93.66	93.31	93.48
BERT+CRF	94.87	**96.50**	95.68
BERT+LSTM+CRF	95.75	95.28	95.51
Zhang and Yang (2018)	94.81	94.11	94.46
Gui et al. (2019)	95.37	94.84	95.11
Ma et al. (2020)	95.30	95.77	95.53
BERT+Ma et al. (2020)	**96.08**	96.13	96.11
Xuan et al. (2020)	-	-	95.45
Liu et al. (2021)	-	-	96.08
EiCi(ours)	96.01	96.24	**96.12**

Table 4: Performance on Resume

Hyperparameters. The model uses the AdamW(Loshchilov and Hutter, 2017) as the optimizer. The optimizer initial learning rate of 2e-5 for the parameter of BERT and 2e-3 for other module. For all the datasets, we apply the same 10 epochs for training. On the MSRA datasets, the max length of the sequence is set to 64, and the training batch size is set to 128. On the Resume datasets, the max length of the sequence is set to 32, and the training batch size is set to 32.

All experiments are conducted on the same machine with 8-cores of Intel(R) Xeon(R) Gold 6226R CPU@2.90GHz and Nvidia Tesla-P100-16GB GPU.

4.4 Experimental Results

Table 3 and Table 4 shows the experimental results of EiCi and the baseline models. For the comparison of model performance, we mainly divide it into two aspects.

Firstly, EiCi is compared with the conventional sequence labeling models. Comparing the conventional LSTM model and the LSTM model with BERT pre-trained knowledge, our model has made great progress.

Secondly, compare with existing models using word information fusion. The main comparison models are SoftLexicon(Ma et al., 2020), FLAT(Xuan et al., 2020) and LEBERT(Liu et al., 2021). The methods of adding word information between these models are also different. SoftLexicon is achieved by splicing word vectors at different

Models	Resume	MSRA
EiCi(ours)	96.12	95.87
- Word Information	95.76	95.72
- Type Module & Word Information	95.57	95.60

Table 5: An ablation study of the proposed model

positions in the presentation layer. FLAT is an improvement from Lattice-LSTM(Zhang and Yang, 2018). LEBERT is achieved by adding word information to the bottom layer of the BERT pre-trained model. The proposed model does not spend extra time training and using external pre-trained word vectors, and the overall performance is slightly better than the previous models using word information.

4.5 Ablation Study

In order to verify the contribution of each component of the proposed model, we conducted ablation experiments on all datasets, as shown in Table 5.

(1) The first innovation of the proposed model is the BERT-based word vectors in the presentation layer. Therefore, we first remove the BERT-based word vectors. In Type Module and NER Module, we just use character-level vectors. It can be seen from Table 5 that on the two datasets, the overall performance of the model will be greatly reduced when the word information added in the representation layer is removed.

(2) The second innovation is the addition of a type recognition module to assist in improving the effect of entity recognition. Because in the proposed model, word information is only added in the type recognition module. We remove the word information and the type recognition module together, and then compare it with the model that only removes the word information. This comparison is used to measure the impact of adding the type recognition module on the performance of the model. From Table 5, it can be observed that the performance of the model will further decline after removing the type recognition module. It can be inferred that the Type Module also plays a positive role in improving the effect of the model.

Through the ablation experiments, it is not difficult to see that the two main improvements of this model have improved the effect of entity recognition to varying degrees. However, it can be observed that the performance of the model drops even more after removing the word information of the presentation layer. Therefore, adding word information has a greater contribution to the improvement of the model effect.

5 Conclusion

In this paper, we propose a new way to incorporate word information, and add a new module to make better use of the word information. It uses the BERT pre-trained model to generate word vectors without using external pre-trained word vectors. The advantage is that it can save the time spent on pre-training word vectors, making the training of the model more efficient. In addition, the word vectors generated based on BERT can better learn contextual information, so it can better represent words than the word vectors previously used.

References

Aitao Chen, Fuchun Peng, Roy Shan, and Gordon Sun. 2006. Chinese named entity recognition with conditional probabilistic models. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 173–176.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuan-Jing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1197–1206.

Pengxiang Cheng and Katrin Erk. 2020. Attending to entities for better text understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7554–7561.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370.

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.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-based lstm-crf with radical-level features for chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications*, pages 239–250. Springer.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence*, 165(1):91–134.

G.D. Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Tao Gui, Ruotian Ma, Qi Zhang, Lujun Zhao, Yu-Gang Jiang, and Xuanjing Huang. 2019. Cnn-based chinese ner with lexicon rethinking. In *IJCAI*, pages 4982–4988.

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 267–274.

Hangfeng He and Xu Sun. 2017. A unified model for cross-domain and semi-supervised named entity recognition in chinese social media. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Jingzhou He and Houfeng Wang. 2008. Chinese named entity recognition and word segmentation based on character. In *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing*.

Z. Huang, X. Wei, and Y. Kai. 2015. Bidirectional lstm-crf models for sequence tagging. *Computer Science*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Gina-Anne Levow. 2006. The third international chinese language processing bakeoff: Word segmentation and named entity recognition. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 108–117.

Wei Liu, Xiyan Fu, Yue Zhang, and Wenming Xiao. 2021. Lexicon enhanced chinese sequence labeling using bert adapter. 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 5847–5858, Online. Association for Computational Linguistics.

Wei Liu, Tongge Xu, Qinghua Xu, Jiayu Song, and Yueran Zu. 2019. An encoding strategy based word-character lstm for chinese ner. 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 2379–2389.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Yanan Lu, Yue Zhang, and Donghong Ji. 2016. Multi-prototype chinese character embedding. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 855–859.

Ruotian Ma, Minlong Peng, Qi Zhang, Zhongyu Wei, and Xuanjing 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, Online. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Diego Mollá, Menno Van Zaanen, Daniel Smith, et al. 2006. Named entity recognition for question answering.

Nanyun Peng and Mark Dredze. 2016. Learning word segmentation representations to improve named entity recognition for chinese social media.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, M. Neumann, M. Iyyer, M. Gardner, and L. Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Desislava Petkova and W Bruce Croft. 2007. Proximity-based document representation for named entity retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 731–740.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Yuanhe Tian, Guimin Chen, Han Qin, and Yan Song. 2021. Federated chinese word segmentation with global character associations. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

Yuanhe Tian, Yan Song, Fei Xia, Tong Zhang, and Yonggang Wang. 2020. Improving chinese word segmentation with wordhood memory networks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

S. Vandenhende, S. Georgoulis, M. Proesmans, D Dai, and L Van Gool. 2020. Revisiting multi-task learning in the deep learning era.

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.

Zhenyu Xuan, Rui Bao, and Shengyi Jiang. 2020. Fgn: Fusion glyph network for chinese named entity recognition. In *China Conference on Knowledge Graph and Semantic Computing*, pages 28–40. Springer.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*, pages 29–48.

Suxiang Zhang, Ying Qin, Wen-Juan Hou, and Xiaojie Wang. 2006. Word segmentation and named entity recognition for sighan bakeoff3. In *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pages 158–161.

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.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Junsheng Zhou, Weiguang Qu, and Fen Zhang. 2013. Chinese named entity recognition via joint identification and categorization. *Chinese journal of electronics*, 22(2):225–230.