

# CHINESE NAMED ENTITY RECOGNITION

XIAOWEN WANG<sup>1</sup>

YIZHI JIANG<sup>1</sup>

## CONTENTS

|     |                                  |    |
|-----|----------------------------------|----|
| 1   | Introduction                     | 2  |
| 2   | Related Works                    | 3  |
| 2.1 | NER . . . . .                    | 3  |
| 2.2 | LSTM and BiLSTM . . . . .        | 3  |
| 2.3 | CRF . . . . .                    | 4  |
| 2.4 | BERT . . . . .                   | 6  |
| 3   | Our Achievements                 | 8  |
| 3.1 | NER Model . . . . .              | 8  |
| 3.2 | Web Application Client . . . . . | 8  |
| 4   | Experiments                      | 9  |
| 4.1 | Dataset . . . . .                | 9  |
| 4.2 | Evaluation . . . . .             | 10 |
| 5   | Conclusion                       | 11 |

---

<sup>1</sup> School of Software Engineering, Tongji University

## 1 INTRODUCTION

Named entity recognition (NER) (also known as entity recognition, entity segmentation, and entity extraction) is a sub-task of information extraction through unstructured data. It aims to locate and classify named entities in text into predefined categories, such as people, organizations, location and so on. Named entity recognition is one of the basic tasks in the field of natural language processing (NLP), and it is also an indispensable component of many natural language processing technologies such as information extraction, information retrieval, machine translation, and question answering systems. From the perspective of natural language processing, NER can be regarded as a type of unregistered word recognition in lexical analysis. [1] It is the problem of the largest number of unregistered words, the most difficult recognition, and the greatest impact on word segmentation. According to the results of SIGHAN (<http://www.sighan.org/>) Bakeoff data evaluation, the loss of segmentation accuracy caused by unregistered words is at least 5 times greater than the ambiguity, which shows the importance of named entity status. [2]

In this project, we addressed the problem of Chinese NER, and (a) investigated the general technology and development of named entity recognition, (b) used the Pytorch framework to build two models for completing NER tasks, (c) built The Chinese NER web application and (d) compared and evaluated the effects of different models in the People's Daily data set.

Research on English named entity recognition started earlier. In 1991, Rau published a related research article on "Extracting and Identifying Company Names" at the 7th IEEE Artificial Intelligence Application Conference. For the first time, he described a system for extracting and identifying company names. This system mainly uses heuristic algorithms and hand-written rules method.[3]. In 1996, the evaluation of named entities was introduced into MUC-6[4] as a sub-task of information extraction, followed by MET-2 of MUC-7[5], and a series of international IEER-99, CoNLL-2002, CoNLL-2003, IREX, LREC, Named entity recognition was used as one of the designated tasks in the meeting.

Compared with English, the task of Chinese NER is more complicated. Due to factors such as word segmentation, the difficulty is mainly reflected in the following aspects:

- 1) Named entities are diverse and numerous, and new named entities are constantly emerging, such as new names of people and places.
- 2) The structure of named entities is more complex, and there is no certain limit on the length of certain types of named entity words. Different entities have different structures. The law can be followed; transliteration also exists in personal names, and there is no uniform word-formation standard. Therefore, the recall rate for such named entities is relatively low.
- 3) In different fields and scenarios, the expansion of named entities is different, and there is a problem of fuzzy classification. The boundaries between different named entities are unclear, and people's names often appear in place and organization names. Methods to properly label these named entity types often involve context and semantic analysis.
- 4) In different cultures, fields and backgrounds, the extension of named entities is different. The delimitation and type determination of named entities have not yet formed a strict naming convention that they all follow.

Our project is based on the above background.

## 2 RELATED WORKS

### 2.1 NER

Many methods have been proposed for NER task. Early studies on NER often exploit SVMs [6], HMMs[7] and CRFs[8], heavily relying on feature engineering. Zhou et al.[9] formulate Chinese NER as a joint identification and categorization task. In recent years, neural network models have been introduced to NER task [10, 11, 12]. Huang et al. [11] exploit BiLSTM to extract features and feed them into CRF decoder. After that, the BiLSTM-CRF model is usually exploited as the baseline. Lample et al. [13] use a character LSTM to represent spelling characteristics. In addition, Wang et al. [14] propose a gated convolutional neural network (GCNN) model for Chinese NER. Peng and Dredze [12] propose a joint model for Chinese NER, which are jointly trained with CWS task. However, the specific features brought by CWS task can lower the performance of the Chinese NER task.

### 2.2 LSTM and BiLSTM

Recurrent Neural Network (RNN) is a type of recursive neural network that takes sequence data as input, performs recursion in the evolution direction of the sequence, and connects all nodes (circular units) in a chain network. The research on recurrent neural networks started in the 1980s and 1990s, and developed into one of the deep learning algorithms in the early 21st century. Bidirectional RNN (Bi-RNN) and Long Short-Term Memory networks (LSTM)[15] are common recurrent neural networks. Recurrent neural networks are memorable, share parameters, and have Turing completeness, so they have certain advantages when learning the non-linear features of a sequence. Recurrent neural networks have applications in Natural Language Processing (NLP), such as speech recognition, language modeling, and machine translation. They are also used for various types of time series forecasting. A recurrent neural network constructed by the Convolutional Neural Network (CNN) is introduced to handle computer vision problems that include sequence inputs.

Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work.<sup>1</sup> They work tremendously well on a large variety of problems, and are now widely used. LSTMs are explicitly designed to avoid the long-term dependency problem. All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

In Figure 2, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations.

The LSTM model can better capture the long-distance dependencies. Because LSTM can learn what information to remember and what information to forget through the training process. But there is still a problem in modeling sentences with LSTM: it is impossible to encode information

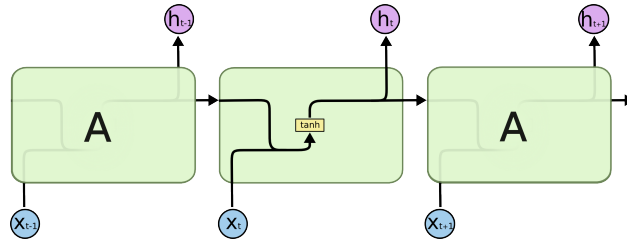


Figure 1: The repeating module in a standard RNN contains a single layer.

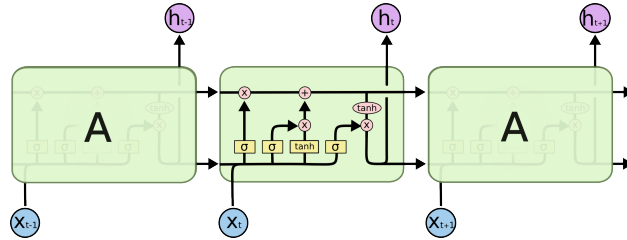


Figure 2: The repeating module in an LSTM contains four interacting layers.

from back to front. In more fine-grained classification, such as the five classification tasks for strong meaning, weak meaning, neutral, weak derogation, and strong derogation, attention needs to be paid to the interaction between affective words, degree words, and negative words. . For example, "This restaurant is too dirty, not as good as next door". "No" here is a modification to the degree of "dirty". BiLSTM can better capture two-way semantic dependencies.

Similar to the LSTM calculation process, BiLSTM adds a reverse operation on top of it. It can be understood as inverting the input sequence and calculating the output again according to the LSTM method. The final result is a simple stacking of the results of the forward LSTM and the results of the reverse LSTM. In this way, the model can consider the context information, and the model is called Bidirectional LSTM (Bi-LSTM).

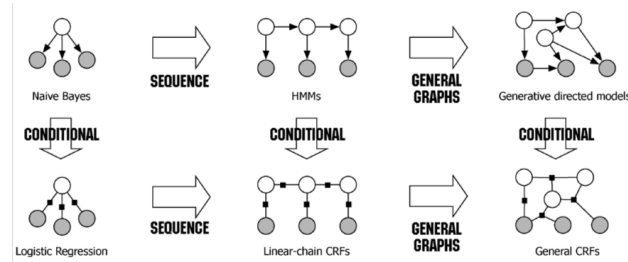
### 2.3 CRF

Conditional Random Fields (CRF) is a conditional probability distribution model for a given set of input sequences and another set of output sequences. It has been widely used in natural language processing.

Conditional random fields (CRFs) are a class of statistical modeling method often applied in pattern recognition and machine learning and used for structured prediction. Whereas a classifier predicts a label for a single sample without considering "neighboring" samples, a CRF can take context into account. To do so, the prediction is modeled as a graphical model, which implements dependencies between the predictions. What kind of graph is used depends on the application. For example, in natural language processing linear chain CRFs are popular, which implement sequential dependencies in the predictions. In image processing the graph typically connects locations to nearby and/or similar locations to enforce that they are treated similarly.

CRFs are a type of discriminative undirected probabilistic graphical model. They are used to encode known relationships between observations and construct consistent interpretations and are often used for labeling or parsing of sequential data, such as natural language processing or

biological sequences and in computer vision. Specifically, CRFs find applications in POS tagging, shallow parsing, named entity recognition, gene finding and peptide critical functional region finding, among other tasks, being an alternative to the related hidden Markov models (HMMs). In computer vision, CRFs are often used for object recognition and image segmentation.



**Figure 3:** Relations between Bayes, logistic regression, HMMs, linear-chain CRF, generative models and general CRFs.

A random field is a whole composed of several positions. When a value is randomly assigned to each position according to a certain distribution, the whole is called a random field. Markov random field is a special case of random field. It assumes that the assignment of a certain position in the random field is only related to the assignment of its adjacent position, and has nothing to do with the assignment of its non-adjacent position. CRF is a special case of Markov random field. It assumes that there are only two variables  $X$  and  $Y$  in Markov random field.  $X$  is generally given, and  $Y$  is generally the output under the given  $X$  condition. In this way, Markov random fields are specialized into conditional random fields. In our ten-word sentence tagging example,  $X$  is a word and  $Y$  is a part of speech. Therefore, if we assume that it is a Markov random field, then it is also a CRF.

For CRF, give an accurate mathematical language description: Let  $X$  and  $Y$  be random variables, and  $P(Y | X)$  be the conditional probability distribution of  $Y$  given  $X$ , if the random variable  $Y$  constitutes a Markov random field, The conditional probability distribution  $P(Y | X)$  is called a conditional random field.

In CRFs, our input data is sequential, and we have to take previous context into account when making predictions on a data point. To model this behavior, we will use Feature Functions, that will have multiple input values, which are going to be:

1. The set of input vectors,  $X$
2. The position  $i$  of the data point we are predicting
3. The label of data point  $i-1$  in  $X$
4. The label of data point  $i$  in  $X$

And the feature function is defined as  $f(X, i, l_{i-1}, l_i)$ . The purpose of the feature function is to express some kind of characteristic of the sequence that the data point represents.

We use Conditional Random Fields by first defining the feature functions needed, initializing the weights to random values, and then applying Gradient Descent iteratively until the parameter values (in this case, lambda) converge. We can see that CRFs are similar to Logistic Regression, since they use the Conditional Probability distribution, but we extend the algorithm by applying Feature functions as our sequential inputs.

## 2.4 BERT

BERT (Bidirectional Encoder Representations from Transformers) is a recent paper published by researchers at Google AI Language. It has caused a stir in the Machine Learning community by presenting state-of-the-art results in a wide variety of NLP tasks, including Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and others.

BERT's key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modelling. This is in contrast to previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training. The paper's results show that a language model which is bidirectionally trained can have a deeper sense of language context and flow than single-direction language models. In the paper, the researchers detail a novel technique named Masked LM (MLM) which allows bidirectional training in models in which it was previously impossible.

In the field of computer vision, researchers have repeatedly shown the value of transfer learning — pre-training a neural network model on a known task, for instance ImageNet, and then performing fine-tuning — using the trained neural network as the basis of a new purpose-specific model. In recent years, researchers have been showing that a similar technique can be useful in many natural language tasks.

A different approach, which is also popular in NLP tasks and exemplified in the recent ELMo paper, is feature-based training. In this approach, a pre-trained neural network produces word embeddings which are then used as features in NLP models.

### 2.4.1 Workflow of BERT

BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. In its vanilla form, Transformer includes two separate mechanisms — an encoder that reads the text input and a decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary. The detailed workings of Transformer are described in a paper by Google.

As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional, though it would be more accurate to say that it's non-directional. This characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word).

The chart below is a high-level description of the Transformer encoder. The input is a sequence of tokens, which are first embedded into vectors and then processed in the neural network. The output is a sequence of vectors of size  $H$ , in which each vector corresponds to an input token with the same index.

When training language models, there is a challenge of defining a prediction goal. Many models predict the next word in a sequence (e.g. "The child came home from \_\_\_\_"), a directional approach which inherently limits context learning. To overcome this challenge, BERT uses two training strategies:

**MASKED LM (MLM)** Before feeding word sequences into BERT, 15% of the words in each sequence are replaced with a [MASK] token. The model then attempts to predict the original value of the masked words, based on the context provided by the other, non-masked, words in the se-

quence. The BERT loss function takes into consideration only the prediction of the masked values and ignores the prediction of the non-masked words. As a consequence, the model converges slower than directional models, a characteristic which is offset by its increased context awareness.

**NEXT SENTENCE PREDICTION (NSP)** In the BERT training process, the model receives pairs of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document. During training, 50% of the inputs are a pair in which the second sentence is the subsequent sentence in the original document, while in the other 50% a random sentence from the corpus is chosen as the second sentence. The assumption is that the random sentence will be disconnected from the first sentence.

When training the BERT model, Masked LM and Next Sentence Prediction are trained together, with the goal of minimizing the combined loss function of the two strategies.

### 2.4.2 Fine Tuning with BERT

BERT can be used for a wide variety of language tasks, while only adding a small layer to the core model:

1. Classification tasks such as sentiment analysis are done similarly to Next Sentence classification, by adding a classification layer on top of the Transformer output for the [CLS] token.
2. In Question Answering tasks (e.g. SQuAD v1.1), the software receives a question regarding a text sequence and is required to mark the answer in the sequence. Using BERT, a Q&A model can be trained by learning two extra vectors that mark the beginning and the end of the answer.
3. In Named Entity Recognition (NER), the software receives a text sequence and is required to mark the various types of entities (Person, Organization, Date, etc) that appear in the text. Using BERT, a NER model can be trained by feeding the output vector of each token into a classification layer that predicts the NER label.

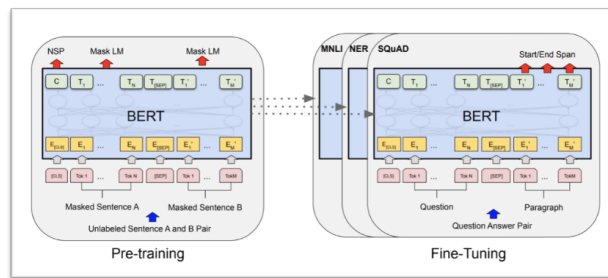


Figure 4: Fine Tuning with BERT

In the fine-tuning training, most hyper-parameters stay the same as in BERT training, and the paper gives specific guidance on the hyper-parameters that require tuning. The BERT team has used this technique to achieve state-of-the-art results on a wide variety of challenging natural language tasks.

### 3 OUR ACHIEVEMENTS

#### 3.1 NER Model

In this project, we combined the exploration of Chinese named entity recognition and chose to use the mature architecture of Embedding-BiLSTM-CRF as our basic model. In addition, we also try to combine the BERT model with our basic model, replace the original embedding layer by pre-training the BERT model, and update the model through Fine-Tuning. We call this model an advanced model.

The input of the basic model is a sequence, and the elements in the sequence correspond to the id of each word in the sentence. The Embedding layer of the model converts the id of the word into a vector corresponding to the word, and the BiLSTM layer encodes and decodes the vector and inputs it into the CRF layer. The CRF layer performs path calculation on the input sequence, and finally obtains the label id corresponding to each word. The schematic diagram of the model is shown in the Figure 5

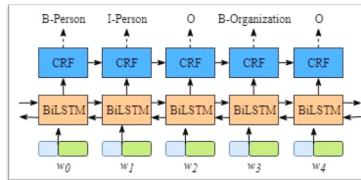


Figure 5: Basic model for Chinese NER

The advanced model is similar to the basic model, except that the Embedding layer is replaced with a pre-trained BERT model, and the input of the model is a word sequence. The BERT model converts the word sequence into the corresponding sequence vector. The subsequent process is the same as the basic model. The schematic diagram of the model is shown in the Figure 6

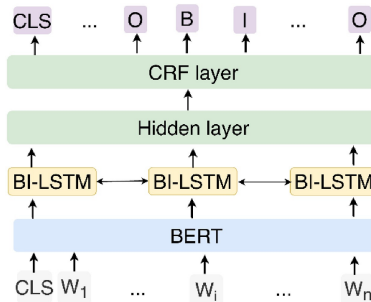


Figure 6: Advanced model for Chinese NER

#### 3.2 Web Application Client

We have developed a simple web application for the Chinese named entity recognition task, which is convenient for users to quickly extract person names, places and organizations from text. The application uses Python as the back end and the front end is developed using the Bootstrap framework. The initial interface of the application is shown in the Figure 7.



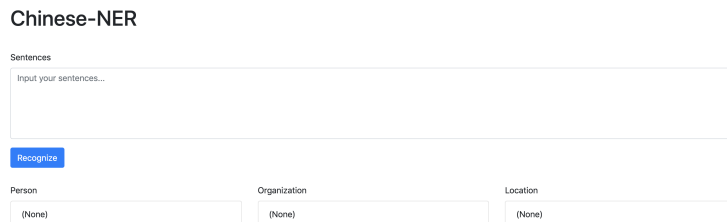


Figure 7: Original Web UI

Enter a sentence in the input box, click the "recognize" button, and the sentence will be sent to the backend by Ajax. The back end will load the trained model, make predictions on the received sentences, and return the prediction results to the front end in the form of a Json file. After the front end receives the returned result, it will refresh part of the webpage to show the prediction result. An example of prediction is shown in the Figure 8.

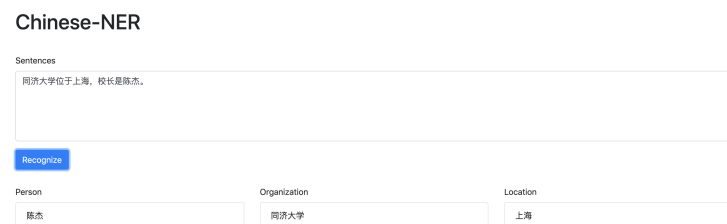


Figure 8: Predict a sentence

In addition to the prediction of a single sentence, the application also supports the simultaneous prediction of multiple sentences. An example is shown in the Figure 9.



Figure 9: Predict sentences

## 4 EXPERIMENTS

### 4.1 Dataset

The People's Daily dataset is one of the most commonly used datasets for the Chinese NER task. The data set contains three tags: LOC (place name), ORG (institution name), and PER (person name). The data is labeled using the BIO labeling strategy. The statistical results of the data are shown in the Figure 10.

| Dataset | # sentence | # word | # LOC | # ORG | # PER |
|---------|------------|--------|-------|-------|-------|
| TRAIN   | 20864      | 979180 | 16571 | 9277  | 8144  |
| DEV     | 2318       | 109870 | 1951  | 984   | 884   |
| TEST    | 4636       | 219197 | 3658  | 2185  | 1864  |

Figure 10: Statistics of People’s Daily Dataset

## 4.2 Evaluation

In order to compare the effects of the basic model and the advanced model, we set up multiple sets of parameters, constructed two types of models for training, and verified the model effects in the test set. The selected evaluation indicators for verification are precision rate, recall rate and f1-score. For each prediction result, we consider the prediction to be correct if and only if the predicted start and end coordinates of the named entity are the same as the golden tag. Given the correct prediction number  $n_{\text{correct}}$ , the number of entities in the golden tag  $n_{\text{golden}}$ , and the number of entities in the prediction result  $n_{\text{predict}}$ , we give the following calculation:

$$\text{precision} = n_{\text{correct}} / n_{\text{predict}}$$

$$\text{recall} = n_{\text{correct}} / n_{\text{golden}}$$

$$f_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

For the basic model, we selected the embedding layer dimension from 50,100,200, the hidden size of Bi-LSTM layer from 128,256, the number of Bi-LSTM layers from 1,2,3, and learning rate from 0.01, 0.001. We used the SGD optimizer to optimize the model, and trained the model for 100 epochs. During the training process, we decide whether to end the training early based on the model’s loss on the validation set. For the advanced model, we used the same parameter set as the basic model for experiments, but trained the model for 10 epochs with Adam optimizer. By comparing the f1-scores of different models in the test set, we get the optimal model parameters as shown in the Table 1.

Table 1: Optimized hyper parameters of models

| model          | # embedding dim | # hidden dim | # Bi-LSTM layer | # learning rate |
|----------------|-----------------|--------------|-----------------|-----------------|
| basic model    | 200             | 256          | 1               | 0.01            |
| advanced model | 200             | 256          | 2               | 0.001           |

The optimal experimental results with the test set are shown in the Table 2.

From the experimental results, it can be seen that after introducing a pre-trained BERT model as an embedding layer, the overall effect of the model is greatly improved in a short time.

Table 2: Optimized results with test dataset

| model          | precision | recall | f1-score |
|----------------|-----------|--------|----------|
| basic model    | 0.8564    | 0.8126 | 0.8339   |
| advanced model | 0.9512    | 0.9242 | 0.9375   |

## 5 CONCLUSION

In this project, we explored the task of Chinese named entity recognition. We combed the development history and important methods of named entity recognition technology, and selected the most widely used Bi-LSTM + CRF model as the basic model to complete the Chinese NER task. In addition, we also introduced a BERT model to improve the base model, that is, replace the embedded layer in the base model with a pre-trained BERT model. We used the People's Daily data set for experiments, and trained and evaluated the basic model and advanced model. The experimental results show that the advanced model can greatly improve the accuracy of the model prediction, the recall rate and f1-score. We also developed a corresponding Web Application for the Chinese NER task, which is convenient for users to use the trained model to predict Chinese sentences.

In this project, the BERT + Bi-LSTM + CRF model we implemented has the advantages of high accuracy, high recall, simple training, and small scale. In addition, when using this model to complete NER tasks, additional data pre-processing operations such as sentence segmentation, part-of-speech analysis, and word feature modeling are not required, which improves the model's efficiency. Our project provides users with a web interface to complete Chinese NER tasks intuitively.

There are still some areas for improvement in our project. The current advanced model is constructed by connecting the Bi-LSTM + CRF model after the BERT model. For the way of using other models (such as GCNN model, GNU model, etc.) combined with the BERT model to complete the Chinese NER task, we have not done much exploration. The improvement and development of Web Application is also one of our future work directions. We should also try to train and evaluate models on larger data sets (such as MSAR), and further explore the significance of the advanced model proposed in this paper for the Chinese NER task.

## REFERENCES

- [1] Zhen Sun and Huilin Wang. Overview on the advance of the research on named entity recognition. *New Technology of Library and Information Service*, 26(6):42–47, 2010.
- [2] Richard Sproat and Thomas Emerson. The first international Chinese word segmentation bakeoff. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, pages 133–143, Sapporo, Japan, July 2003. Association for Computational Linguistics.
- [3] L. F. Rau. Extracting company names from text. In [1991] *Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application*, volume i, pages 29–32, Feb 1991.
- [4] Ralph Grishman and Beth Sundheim. Message understanding conference- 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996.
- [5] N. A. Chinchor. Overview of MUC-7/MET-2. In *Proceedings of the 7<sup>th</sup> Message Understanding Conference (MUC7)*, April 1998. Available at [http://www.muc.saic.com/proceedings/muc\\_7\\_toc.html](http://www.muc.saic.com/proceedings/muc_7_toc.html).
- [6] Hideki Isozaki and Hideto Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1, COLING '02*, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.
- [7] Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a high-performance learning name-finder. In *Fifth Conference on Applied Natural Language Processing*, pages 194–201, Washington, DC, USA, March 1997. Association for Computational Linguistics.
- [8] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.
- [9] J. Zhou, W. Qu, and F. Zhang. Chinese named entity recognition via joint identification and categorization. *Chinese Journal of Electronics*, 22:225–230, 04 2013.
- [10] Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch.
- [11] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging.
- [12] Nanyun Peng and Mark Dredze. Improving named entity recognition for Chinese social media with word segmentation representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 149–155, Berlin, Germany, August 2016. Association for Computational Linguistics.
- [13] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition.

- [14] Chunqi Wang, Wei Chen, and Bo Xu. Named entity recognition with gated convolutional neural networks. In Maosong Sun, Xiaojie Wang, Baobao Chang, and Deyi Xiong, editors, *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 110–121, Cham, 2017. Springer International Publishing.
- [15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.