Chinese Named Entity Recognition Augmented with Lexicon Memory

Yi Zhou (周 奕), Xiao-Qing Zheng* (郑骁庆), Member, CCF, ACM and Xuan-Jing Huang (黄萱菁), Member, CCF, ACM

School of Computer Science, Fudan University, Shanghai 200438, China
Shanghai Key Laboratory of Intelligent Information Processing, Shanghai 200438, China

E-mail: yizhou17@fudan.edu.cn; zhengxq@fudan.edu.cn; xjhuang@fudan.edu.cn

Received November 12, 2020; accepted December 7, 2021.

Abstract Inspired by the concept of content-addressable retrieval from cognitive science, we propose a novel fragment-based Chinese named entity recognition (NER) model augmented with a lexicon-based memory in which both character-level and word-level features are combined to generate better feature representations for possible entity names. Observing that the boundary information of entity names is particularly useful to locate and classify them into pre-defined categories, position-dependent features, such as prefix and suffix, are introduced and taken into account for NER tasks in the form of distributed representations. The lexicon-based memory is built to help generate such position-dependent features and deal with the problem of out-of-vocabulary words. Experimental results show that the proposed model, called LEMON, achieved state-of-the-art performance with an increase in the F1-score up to 3.2% over the state-of-the-art models on four different widely-used NER datasets.

Keywords named entity recognition (NER), lexicon-based memory, content-addressable retrieval, position-dependent feature, neural network

1 Introduction

Named entity recognition (NER) aims to locate and classify elements in sentences into pre-defined categories such as the names of persons, organizations, and locations. Many NER systems were developed by either rule-based approaches or statistical methods. Rule-based systems identify entity names by applying linguistic grammar rules governing the derivation of entity names[1], while statistical methods identify entity names based on the distribution of their components and their relation to the whole estimated from a larger corpus[2, 3]. Recently, deep neural networks have been explored to identify named entities[4], and several network architectures have been investigated for NER tasks, such as recurrent neural networks (RNNs)[5, 6] with the encoder-decoder framework[7, 8]. Neural models have gained remarkable success due to their extraordinary ability to memorize some cases and generalize to other unseen cases[9]. However, these models still suffer from two challenging problems when they are applied to the NER task: ambiguous word boundaries and out-of-vocabulary words.

Ambiguity of Word Boundaries. Traditional approaches to Chinese NER can be roughly divided into two paradigms: character-based and word-based models. Character-based models usually are not comparable to their word-based competitors due to the lack of explicit word-level information[10, 11], while word-based models suffer from the issue of error propagation. The results of word segmentation largely determine the boundaries of named entities, and the errors made in the word segmentation step greatly impact the final NER performance. Zhang and Yang[12] proposed a lattice-based model to encode a sequence...
of characters as well as every possible word that matches against a lexicon\[13\]. However, the salient boundary features (prefix and suffix) of each name candidate might be blurred because they consider all possible segmentations, but only a few of them are feasible, possibly introducing unnecessary noise. Named entities often occur in the form of a fragment (a sequence of contiguous words) rather than a single character or word\[14\], which implies that fragment-based models deserve further exploring.

**Out-of-Vocabulary Words.** Unknown words account for several times more errors than name boundary ambiguities. The adverse effects of unknown words could be much alleviated if word-level information can be integrated for the recognition in form of their embeddings and such word embeddings can be learned from a large text corpus in an unsupervised way. As shown in Fig.1, “Microsoft” would be correctly recognized with a higher probability because its embedding is close to those of “Google” and “Amazon” in the embedding space. Similar rules also apply to location entities such as “Rome”, “Tokyo”, and “Beijing”. However, for people or places that are not well known by the public, their names, such as “王奕” (Wang Yi) and “临州大学” (Linzhou University), are unlikely to appear in any dictionary. In these cases, their position-dependent information (prefix or suffix) would be quite helpful for the recognition. For example, most Chinese people’s names start with a common surname followed by one or two characters. Organization names usually begin with the name of a city or a country, and end with one of few words like “公司 (company)”, “大学 (university)”, and “医院 (hospital)”.

In this study, we propose a fragment-based approach to address the above problems, which combines features at different levels of granularity. Particularly, we explore the feasibility of extracting position-dependent features (including prefix, suffix, and infix) and integrating them into neural NER models in the form of their distributed representations. It has been proven to be fruitful to incorporate a lexicon (an external dictionary) for NER\[5, 15\], even though such word-level features are simply derived by string matching in a rigid, discrete manner. Constructing a lexicon by manually collecting gazetteer entries such as common person surnames and geographical terms is time-consuming and labor-intensive. Therefore, it is

**Training Instance:** 特朗普在夏威夷发了一条推特。
Trump (PER) posted a tweet in Hawaii (LOC).

**Training Instance:** 我在微软发了一条推特。
I posted a tweet in Microsoft (ORG).

**Training Instance:** 我在罗马发了一条推特。
I posted a tweet in Rome (LOC).

\[Fig.1. Illustrative instances demonstrate that the position-dependent features can benefit the NER task. For the instances involving “Microsoft” and “Rome”, the semantic information of the words alone is enough for the recognition because their word embeddings are close to those of their kinds. Conversely, lesser-known individuals such as “Wang Yi” or organizations like “Linzhou University” often encounter out-of-vocabulary issues. The position-dependent information (e.g., the prefix “Wang” or the suffix “University”) could be useful for the recognition.\]
worth exploring the possibility of deriving such features automatically.

The fragment-based approach conforms to the way how human beings recognize entity names. Given a fragment, a person’s attention will be drawn towards the content most relevant to his/her memory, which can be regarded as content-addressable retrieval, a concept borrowed from cognitive science to artificial intelligence. From the viewpoint of cognitive systems, a biological brain does not learn by a single and global optimization principle, but is modular and composed of distinct subsystems, such as memory and control, which can interact with each other.

Inspired by this finding from cognitive science, we propose a fragment-based model, called LEMON, which is augmented with a lexicon-based memory for Chinese NER. The model consists of three main sub-modules: a character encoder that imitates the process of scanning each character in an input sentence to capture the global semantics, a fragment encoder that simulates the procedure of reading a sub-sequence (such as words or fragments) in the sentence, and a memory that stores plenty of words that have ever seen. A ranking algorithm is used to determine whether a fragment is a valid name and which category it belongs to by taking its prefix, suffix, and infix features into account. Experimental results show that the proposed model achieved state-of-the-art results on four different benchmark datasets. The source code of LEMON can be downloaded from the GitHub website.

The remainder of this paper is organized as follows. Section 2 presents a brief overview of related work in NER. In Section 3, we introduce our fragment-based Chinese NER model augmented with a lexicon-based memory. Experimental results on four different benchmark datasets are reported in Section 4. The conclusion and future work are summarized in Section 5.

2 Related Work

The NER task has been a long-standing challenge for the natural language processing (NLP) community, and the problem of Chinese NER is far from being solved. We here focus on the work most related to this study and refer readers to two recent survey articles.

2.1 Local Detection Approach

A local detection approach was firstly proposed by Xu et al. for both mention detection and name entity classification tasks. Their model uses a fixed-sized ordinary forgetting encoding (FOFE) to represent all possible fragments in an input text. Our model differs from theirs in that we adopt a character encoder to capture the relationship between the fragments and their contexts to better provide global contextual features. Besides, the position-dependent features are introduced and used for each candidate name with the help of a lexicon-based memory.

2.2 Attention Mechanism

The attention mechanism was first proposed for machine translation, which learns an alignment between a source language and a target one by estimating their correlation. It also has been applied to NER tasks in several ways: integrating character-level information by attending to characters, capturing global contextual features by attending to different sentences in a document, and adopting an adaptive co-attention between texts and images. The memory network was first introduced for question answering. To the best of our knowledge, this study is among the first ones to incorporate word-level features via memory networks for named entity recognition.

3 Model

We describe the architecture of our proposed model in the following. As shown in Fig.2, LEMON mainly consists of three modules: a character encoder that maps each character into its feature vector, a fragment encoder which can transform any variable-length sub-sequence in an input sentence into a fixed-sized vector representation, and a lexicon memory that is designed to help disambiguate the word boundaries and deal with the out-of-vocabulary problem by providing the syntactic and semantic features for possible words appeared in any fragment.

3.1 Character Encoder

Given an input sentence \( S = s_{[1:n]} \) consisting of \( n \)
characters, each character \( s_i \in S_{[1:n]} (1 \leq i \leq n) \) is first mapped into its feature vector \( w_i \). It has been proven that the features derived from the results of word segmentation and part-of-speech (POS) tagging are useful for NER tasks\(^{[29, 30]} \), and thus we augment each character representation with its word-level and POS information in a “soft” way. As shown in Fig.3, the BMES scheme is used to represent the results of word segmentation\(^{[31]} \). In this scheme, each character will be assigned one of four possible boundary tags: “B” for a character located at the beginning of a word, “M” for that inside of a word, “E” for that at the end of a word, and “S” for a character that is a word by itself. Each character is also assigned a POS tag the same as that of the word to which it belongs. The feature vector of each character is obtained by concatenating the feature vectors from the three parts as follows:

\[
w_i = E_{\text{char}}^i \oplus E_{\text{seg}}^i \oplus E_{\text{pos}}^i;
\]

where \( E_{\text{char}}^i, E_{\text{seg}}^i, \) and \( E_{\text{pos}}^i \) are the feature vectors of the character, word boundary label and POS tag for the character at the \( i \)-th position respectively. The symbol \( \oplus \) is used to denote the concatenation operator for feature vectors. The character encoder is used to obtain the context-aware representation of each character in a given sentence \( s_{[1:n]} \):

\[
t_i = \text{ENCODER}_{\text{character}}(w_{[1:n]}),\]

where \( t_i \in \mathbb{R}^{d_i} \), and \( d_i \) is the size of the context-aware character feature vectors produced by the encoder. The character encoder could be implemented with various network architectures including bi-directional long short-term memory (LSTM)\(^{[32, 33]} \) and transformer\(^{[34]} \). In theory, the LSTMs can keep track of arbitrary long-term dependencies in the input sequences, and the transformers are able to capture the dependencies between different words with any distance in a text.

### 3.2 Fragment Encoder

The fragment encoder is used to produce a feature vector for each \( n \)-gram in an input sentence. Given a sequence of characters, \( T = t_{[i:j]} \in \mathbb{R}^{(j-i+1)\times d_i} \), where \( d_i \) is the dimensionality of the context-aware character representations, the fragment encoder learns to map the matrix \( T \) to a fixed-sized vector \( f_{[i:j]} \in \mathbb{R}^{d_f} \), where \( d_f \) is the dimensionality of fragment embeddings.

\[
f_{[i:j]} = \text{ENCODER}_{\text{fragment}}(t_{[i:j]}),
\]

where \([i:j]\) denotes a candidate fragment spanning from character \( i \) to \( j \).

Assuming that the maximum length of named entities is \( m \), for an input sentence consisting of \( n \) characters, the number of all possible fragments would be \( m \times (2n - m + 1)/2 \). The computational complexity of enumerating all the fragments is \( O(mn) \), which is quite time-consuming. We design a method by bi-directional recurrent neural networks (Bi-RNNs) to make use of the inherent recursive structures of the fragments, which greatly reduces the complexity. Note that the representations of shorter fragments can be used to generate those of longer ones, and thus all the possible fragments can be enumerated in \( O(n) \) time. Unfortunately, this efficient method cannot be applied when the fragment encoder is built on the transformer. If the transformer is applied, it requires \( O(mn) \) time to enumerate all the possible fragments.
The increase in the computational cost will greatly reduce the speed of training and inference. Therefore, we rule out the possibility of building the fragment encoder based on the transformer.

The fragment encoder also can be implemented with other network architectures. Xu et al.\cite{10.1007/978-3-319-23461-7_34} chose to use the Fixed-Sized Ordinary Forgetting Encoding (FOFE) to implement this encoder, which incorporates a forgetting gate to capture the positional information of characters or words\cite{10.1007/s10479-018-1659-8}. The bag-of-words method that simply averages the representations of words or characters has also been explored to build such a fragment encoder\cite{10.1007/978-3-319-23461-7_34,10.1007/978-3-319-23461-7_34}. We will report and compare the results achieved by the models implemented with different fragment encoders in Section 4.

3.3 Lexicon Memory

In the following, we describe how to automatically construct a lexicon from a large unlabeled dataset and how the constructed lexicon can be used to derive the word-level, position-dependent features by performing pattern matching operations (i.e., content-addressable retrieval) against a fragment over the lexicon in four different ways.

3.3.1 Lexicon Construction

The lexicon used in this study does not work as a simple gazetteer (i.e., a vocabulary consisting of known entity names), but it contains all the possible words extracted from a dataset, which allows us to leverage a large unlabeled dataset to obtain rich features about the words. Following Zhang and Yang\cite{10.1007/s10479-018-1659-8}, the lexicon is built by automatically segmenting the Chinese Gigaword dataset\cite{catalog.ldc.upenn.edu/LDC2011T13} and collecting the resulting words. The embeddings of the words in the lexicon are learned by the Word2Vec toolkit\cite{10.1007/s10479-018-1659-8} with the same dataset. Due to the ambiguity of Chinese word segmentation, there may exist multiple smaller parts of a word in the lexicon, which reflect different levels of granularity. For example, the named entity “财政部” (Ministry of Finance) contains four smaller parts: “财政” (public finance), “财” (finance), “政” (administration), and “部” (ministry). The words at the finer level of granularity, such as “部” (ministry), can provide finer and more valuable word-level features for the NER task.

3.3.2 Matching Mode

Given a fragment $s_{[i:j]}$, we perform the pattern matching against it over the constructed lexicon $V$. We define four types of matching modes as follows.

- **Exact Matching.** If there exists one word in the lexicon that is the same as the fragment, we call it an exact matching found for this fragment.
- **k-Prefix Matching.** If the first $k$ characters of a fragment are matched against a word, we call it $k$-prefix matching. For example, a fragment “xyz” matches “xy” in a 2-prefix matching mode. Such matching patterns provide informative features to recognize the named entities whose prefixes are usually chosen from a limited number of words, such as commonly-used Chinese surnames like “上官” (Shangguan) and “司马” (Sima).
- **k-Suffix Matching.** We call it $k$-suffix matching if the last $k$ characters of a fragment (i.e., “xyz”) is matched against a word (i.e., “yz”). These matching patterns are quite useful to recognize the entities whose names end with one of the certain words. For example, many locations and organizations share similar suffixes, such as “省” (Province) and “部” (Ministry).
- **Infix Matching.** If a word can be found in the middle of a fragment, it is an infix matching. Its role is slightly different from the above-mentioned modes, and such a matching serves as a hint that a fragment might contain a nested structure.

For the Chinese NER task, the first (or last) one and two characters are relatively more important than those in other positions. Therefore, we group the results of different types of matching into multiple buckets according to their importance. We define LEMON-$K$ in the way that for each distinct $k$, the feature derived from $k$-prefix (or $k$-suffix) matching is placed into a separate bucket if $k \leq K$, while the remaining features ($k > K$) are grouped into a single bucket. The value of $K$ is a hyper-parameter that needs to be tuned.

3.4 Attention over Lexicon Memory

The memory network\cite{10.1007/s10479-018-1659-8,10.1007/s10479-018-1659-8} is a natural choice to implement the concept of the content-addressable retrieval, and can be applied to extract the relevant features from a lexicon-based memory. Specifically, given a matching instance $m_{[l:e]}$, where $l$ denotes a
matched word in the lexicon, and $c$ denotes a mode in which it is matched, both of the matched word and the corresponding matching mode would be mapped into two feature vectors respectively. These two vectors are then concatenated to produce a memory unit as follows.

$$m_{(i,c)} = E_{I}^{\text{lex}} \oplus E_{c}^{\text{mod}},$$

where $E_{I}^{\text{lex}} \in \mathbb{R}^{n_{\text{lex}} \times d_{\text{lex}}}$, $n_{\text{lex}}$ is the size of the lexicon, $d_{\text{lex}}$ is the dimensionality of the vector space into which all the words in the lexicon are embedded, $E_{c}^{\text{mod}} \in \mathbb{R}^{n_{\text{mod}} \times d_{\text{mod}}}$, $n_{\text{mod}}$ is the number of the matching modes, and $d_{\text{mod}}$ is the dimensionality of the feature vectors used to represent different matching modes.

For a fragment $s_{[i:j]}$, we first find all of its matched words against the lexicon, then group them into multiple buckets in the way described in Subsection 3.3.2, and finally assemble them into a matrix $M \in \mathbb{R}^{n_{\text{lex}} \times d_{\text{lex}}}$. This matrix is a lexicon memory dynamically created for the fragment, where $d_{m} = d_{\text{lex}} + d_{\text{mod}}$ and $n_{m}$ is the number of matches founded. Given a fragment representation $f$ and its corresponding lexicon memory $M$, a scaled bi-linear attention is performed for $f$ over $M$ like [23, 34],

$$\text{attention}(f, M) = \text{softmax} \left( \frac{f W M^T}{\sqrt{d_m}} \right) M,$$

where $W \in \mathbb{R}^{d_{f} \times d_{m}}$ is a parameter matrix to be tuned, and $d_{f}$ is the size of fragment vector representations.

### 3.5 Classification and Decoding

#### 3.5.1 Loss Function

For a fragment (or a span), its vector representation $f$ and the result of the attention operation over the lexicon, namely $\text{attention}(f, M)$, are concatenated to produce the final representation $r_{[i:j]}$ for the fragment. This representation is then fed into a multi-layer feed-forward neural network to predict the type of entities. If a fragment does not belong to any entity, it would be labeled as “None”. We take a recently proposed focal loss as the training objective to mitigate the sample-imbalance problem [39].

$$\text{loss}(p_i) = -\alpha_i (1 - p_i)^\gamma \log(p_i),$$

where $p_i$ denotes the predicted probability of the true label, $\alpha_i$ is a parameter vector for the true label which will be tuned during the training process, and $\gamma$ is a hyper-parameter that governs the relative importance of the positive samples with the negative ones. If the values of $\alpha_i$ and $\gamma$ are both set to 1, the focal loss is reduced to the standard cross-entropy loss.

#### 3.5.2 Decoding Strategy

A decoding layer is stacked on top of the entity detector which helps to resolve the issue that some overlapped fragments might be all recognized as valid entities [14]. The following rules will be applied in the decoding process.

- A threshold $\rho$ is used to filter the recognition results. A fragment is identified as an entity if the model assigns the highest probability to this entity type and the probability is greater than $\rho$; otherwise, it will be recognized as “None”.
- If a recognized entity contains another candidate (nested) entity, only the outer entity will remain for further processing.
- If two identified entities overlap each other, only the one with the higher probability is considered.

We find that such a decoding strategy works well although it applies only three simple rules to filter out the candidates. This strategy also can be used to recognize the nested entities by removing the second rule.

### 4 Experiments

We conducted three sets of experiments. The goal of the first one is to test several variants for the different components (character and fragment encoders) on the development set of OntoNotes-4 [40] to gain some understanding of how the choice of architectures and features impacts the performance. The second set of experiments was run on four different benchmark datasets, and we report the results achieved by LEMON and other competitive models. In the third set, we investigated the effect of attention mechanism over the lexicon memory and how the choice of hyperparameters impacts on the performance. An ablation study was also conducted to assess whether the model can benefit from the introduced lexicon-based memory. The $F_1$-score was used to evaluate the performance, and this score is the harmonic mean of precision $P$ and recall $R$, which is defined as $2PR/(P + R)$.

#### 4.1 Experimental Settings

##### 4.1.1 Datasets

We evaluated our model on four different datasets
using the standard split of data: the OntoNotes-4\cite{40}, MSRA\cite{41}, Weibo NER\cite{29, 42}, and Resume\cite{12} datasets, widely used to evaluate the performance of NER models. The statistics of these four datasets are listed in Table 1. As mentioned in Subsection 3.1, each character should be assigned with a soft-word label and a POS tag. All the datasets were segmented and tagged by using the THULAC toolkit\cite{43}, which achieved about 88% of F1-score in the word segmentation on these datasets. For the OntoNotes-4 dataset, the gold segmentation results and part-of-speech tags are available, and we report the results both with and without gold segmentation and POS tags.

| Dataset       | #Train ($\times 10^3$) | #Dev ($\times 10^3$) | #Test ($\times 10^3$) | Domain       |
|---------------|------------------------|----------------------|-----------------------|--------------|
| OntoNotes-4\cite{40} | 15.7                   | 4.30                 | 4.30                  | News         |
| MSRA\cite{41}    | 46.4                   | --                   | 4.40                  | News         |
| Weibo NER\cite{29, 42} | 1.4                    | 0.27                 | 0.27                  | Social media |
| Resume\cite{12}  | 3.8                    | 0.46                 | 0.48                  | Resume       |

Note: "#Train", "#Dev", and "#Test" denote the sizes of training, development, and test datasets, respectively. -- means not available.

### 4.1.2 Training Details

The proposed model was implemented by using the PyTorch deep learning framework\cite{44}. All the experiments were run on a computer equipped with an Intel Xeon processor and eight NVIDIA TITAN-XP GPU cards. We tuned all the hyper-parameters on the development set of the OntoNotes-4 dataset. The sizes of word and character embeddings were both set to 50, and the sizes of soft-word and POS tag embeddings were both set to 25. We found in the preliminary experiments that it achieved a good trade-off between the training speed and performance when setting the dimensionality of character embeddings to 50. Moreover, Zhang and Yang\cite{12} (one of the strongest competitors) also set the size of character embeddings to 50. For a fair comparison, we also used the same method to initialize the character embeddings as theirs and pre-trained both the word and character embeddings on the Chinese Giga-word by the Word2Vec toolkit\cite{37}.

The Dropout mechanism was applied to the character encoder at the embedding layer with a drop rate of 0.3. All learned parameters were updated by the Adam Optimizer\cite{45}, and we chose to use a sparse version of the Adam Optimizer to update the pre-trained embeddings. The learning rate was set to $1.0 \times 10^{-5}$, and the weight decay to $1.0 \times 10^{-7}$. The weight decay was set to $1.0 \times 10^{-5}$ for the Weibo NER dataset because we found that a larger value may make the network hard to converge on this dataset.

### 4.2 Choice of Architectures and Features

We carried out a set of preliminary experiments on the development set of OntoNotes-4 to select the network architecture by trying several different components, and to gain some understanding of how the choice of features impacts the performance.

#### 4.2.1 Impact of Different Architectures

We tried several combinations of different character and fragment encoders to find a better solution for NER. Three different types of networks were tested as the character encoder, and we also tried three different architectures for the fragment encoder. A simple embedding layer serves as a baseline for the character encoder. Two popular sequence models of a transformer with six layers, eight heads, and the hidden size of 512, and a bi-directional LSTM with two layers and the hidden size of 256 were also evaluated. As to the fragment encoder, we tested three different implementations of Bag-of-Words (BOW), FOFE ($\alpha =0.5$)\cite{14}, and bi-directional LSTM. Xu et al.\cite{14} used an embedding layer as the character encoder and an FOFE as the fragment encoder, and they predicted the type of an n-gram candidate with the help of its left and right contexts. We also tried to integrate such contextual information for NER, but the results of preliminary experiments showed that its contribution to the performance is negligible.

The results of different combinations on the development set of OntoNotes-4 are shown in Table 2. A special string "--" denotes that the result is not available in that case. The performances of all models will drop about 4% in the F1-score if we used the results of word segmentation and POS-tagging automatically produced by the THULAC toolkit rather than their ground truth. It shows that the NER performance is significantly impacted by the results of the upstream tasks due to the error propagation. Unless otherwise specified, the used Bi-RNNs were all implemented by bi-directional LSTMs.

**Character Encoder.** Bi-RNN always outperforms other character encoders due to its ability in modelling long-term dependencies. The transformer per-
forms slightly better than the baseline although it achieved impressive results in the machine translation. In our model, the positional information is quite important to produce the features of each character, which are further processed to generate fragment-level representations for the prediction. Unlike Bi-RNNs, the transformer contains no recurrence, and “positional encodings” are added to the input embeddings, which makes it possible for the transformer to model the order of a sequence. However, experimental results show that the positional information provided by the positional embeddings is not sufficient enough to perform NER tasks, which partly explains why the character encoder based on Bi-RNN performs better than that based on the transformer. As stated by Devlin et al., another possible explanation is that the number of training sentences is not sufficient enough to match the model capacity of the transformer.

**Fragment Encoder.** Bi-RNN performs better than the other encoders, especially when the character encoder is not built based on the Bi-RNN. BOW performs inferior to others since it is unable to model the order information in a sequence, which is critical for the entity recognition. FOFE learns to produce a linear combination of the word representations in a subsequence, which is less flexible than Bi-RNN in the sequence modeling since the latter is capable of learning non-linear combinations over an entire sequence.

**Lexicon Memory.** The constructed lexicon memory greatly boosts the performance of any combination of components, with an average increase of about 5% in F1-score. It can be taken as strong empirical evidence that the introduced lexicon memory can significantly improve the models’ performance for the named entity recognition tasks.

### 4.2.2 Impact of Different Features

We also trained an LSTM-CRF model as a representative of traditional approaches for comparison by using an open-source neural sequence labeling toolkit, named NCRF[47]. The impact of different features is shown in Table 3. The experimental results demonstrated that the features derived from the results of word segmentation and POS-tagging make positive contributions to all the models no matter whether they are labeled by humans or produced by an automatic toolkit. Experimental results also showed that the results of word segmentation have a relatively little impact on the NCRF model. NCRF performs NER tasks by the sequence labeling while LEMON does the same tasks by classifying all the possible fragments into the pre-defined categories of named entities. When evaluating each possible fragment, the word boundary information is relatively more important for LEMON to decide whether it is a named entity and which category it belongs to.

As shown in Table 3, LEMON can still beat the LSTM-CRF model by about 5% in F1-score without using any word segmentation or part-of-speech information, which shows that the introduced lexicon memory indeed provides the valuable word-level, position-dependent features via the attention mechanism.

### 4.3 Results

LEMON-2 achieved state-of-the-art results on all the four NER datasets. The ending number “2” de-
notes how many distinct \( k \) values are used to group the features derived by the pattern matching against the lexicon (see Subsection 3.3.2). As shown in Table 4 and Table 5, LEMON outperforms the Lattice LSTM on both the MSRA and Resume NER datasets. Besides, our model also achieved the highest F1-score on the OntoNotes-4 dataset (see Table 6). Note that the Weibo NER data is extracted from the social media, which contains just 1.4k samples and is full of non-standard expressions. The problems of out-of-vocabulary (OOV) and word boundary ambiguity are more serious in this dataset than the others. As we can see from Table 7, LEMON still performs better than the other competitors by a fairly significant margin (at least 3% increase in F1-score).

We also conducted experiments on the same four datasets to see how far we could go by using a pre-trained BERT as the base model of the character encoder. As discussed in Subsection 3.2, if the fragment encoder is built on the transformer, it will greatly reduce the speed of training and inference, and thus we still used the Bi-RNN to implement the fragment encoder in this variant. If the pre-trained BERT is applied, the performance will increase by about 1.9% of F1-score on average across the four datasets.

Table 3. Results on the OntoNotes-4 Development Set Using Different Features

| Model       | Ground Truth | Automatically Labelled |
|-------------|--------------|------------------------|
|             | P (%) | R (%) | F1 (%) | P (%) | R (%) | F1 (%) |
| NCRF Char   | 66.37 | 60.21 | 63.14 | --    | --    | --     |
| NCRF Char + Seg | 70.58 | 69.96 | 70.27 | 70.77 | 63.33 | 66.85  |
| NCRF Char + Pos | 71.81 | 74.48 | 73.12 | 70.20 | 70.26 | 70.23  |
| NCRF Char + Seg + Pos | 75.63 | 72.35 | 73.08 | 72.88 | 68.18 | 70.45  |
| Without Lex Char | 67.60 | 55.03 | 60.67 | --    | --    | --     |
| Without Lex Char + Seg | 72.16 | 66.09 | 68.99 | 70.48 | 62.65 | 66.33  |
| Without Lex Char + Pos | 74.39 | 65.44 | 69.63 | 72.87 | 63.73 | 67.99  |
| Without Lex Char + Seg + Pos | 74.97 | 72.23 | 73.58 | 76.29 | 64.43 | 69.86  |
| Lex Char | 77.27 | 60.73 | 68.01 | --    | --    | --     |
| Lex Char + Seg | 78.40 | 70.75 | 74.38 | 76.11 | 64.63 | 69.91  |
| Lex Char + Pos | 77.71 | 72.35 | 74.93 | 77.46 | 66.89 | 71.79  |
| Lex Char + Seg + Pos | 78.70 | 74.95 | 76.78 | 76.41 | 68.61 | 72.30  |

Note: The best results are highlighted in bold fonts.

Table 4. Results on the MSRA Dataset

| Model       | P (%)   | R (%)   | F1 (%)  |
|-------------|---------|---------|---------|
| Chen et al.[48] | 91.22   | 81.71   | 86.20   |
| Zheng et al.[49] | 92.20   | 90.08   | 91.18   |
| Lu et al.[50] | --      | --      | 87.94   |
| Dong et al.[51] | 91.28   | 90.62   | 90.95   |
| Zhang and Yang[12] | 93.57   | 92.79   | 93.18   |
| LEMON       | 95.40   | 91.77   | 93.55   |

Note: The best results are highlighted in bold.

Table 5. Results on the Resume NER Dataset

| Model       | P (%)   | R (%)   | F1 (%)  |
|-------------|---------|---------|---------|
| Word†      | 93.72   | 93.44   | 93.58   |
| Char†      | 93.66   | 93.31   | 93.48   |
| Word + Char + BiChar† | 94.07 | 94.42 | 94.24 |
| Char + BiChar + Softword† | 94.53 | 94.29 | 94.41 |
| Zhang and Yang[12] | 94.81 | 94.11 | 94.46 |
| LEMON      | 95.59   | 94.07   | 94.82   |

Note: The models indicated with † are those in which the sequence labeling model (LSTM + CRF) was used. The best results are highlighted in bold.

Table 6. Results on the OntoNotes-4 Dataset

| Model       | P (%)   | R (%)   | F1 (%)  |
|-------------|---------|---------|---------|
| Wang et al.[52]† | 76.43   | 72.32   | 74.32   |
| Che et al.[53]† | 77.71   | 72.51   | 75.02   |
| Yang et al.[54]† | 72.98   | 80.15   | 76.40   |
| Zhang and Yang[12] | 76.35   | 71.56   | 73.88   |
| LEMON†     | 79.27   | 78.29   | 78.78   |
| LEMON      | 80.61   | 71.05   | 75.53   |

Note: The model with “†” are those in which the gold results of word segmentation were used. The best results are highlighted in bold.

Table 7. Results on the Weibo NER Dataset

| Model       | P (%)   | R (%)   | F1 (%)  |
|-------------|---------|---------|---------|
| Peng and Dredze[29] | --     | --      | 58.99   |
| He and Sun[54] | --     | --      | 58.23   |
| Zhang and Yang[12] | --     | --      | 58.79   |
| LEMON      | 70.86   | 55.42   | 62.19   |

Note: The best results are highlighted in bold.

---

©https://huggingface.co/datasets/weibo_ner, Sept. 2023.
Specifically, it boosts the F1-scores to 94.31%, 95.76%, 79.26%, and 64.42% on the MSRA, Resume NER, OntoNotes-4, and Weibo NER datasets, respectively. Although it would be interesting to see whether other pre-trained language models, such as ERNIE\cite{55}, XLNet\cite{56}, and ELMo\cite{57}, and knowledge graph-enhanced variants like K-BERT\cite{58} can be integrated and used to further improve the NER performance in the proposed approach, in this study we would like to focus on how to incorporate a lexicon for NER using the memory network, and leave how to better integrate these pre-trained language representation models as future work.

4.4 Discussion

We also conducted some experiments on the Weibo NER dataset to investigate the effect of the attention mechanism over the lexicon memory, and how the choice of the threshold values and focal loss coefficients impacts the NER performance.

4.4.1 Attention over Lexicon Memory

A heap map (shown in Fig.4) illustrates which words will be given more weights computed by the attention scores over the lexicon memory. As shown in this heat map, our model can learn to assign more weights on the keywords of named entities, and the attentions are sharp for those words particularly informative for the NER task.

Taking the entities of “ORG” (organization) as examples, more weights are placed to the last two characters, such as “中心” (center), “政府” (government), “学校” (school), and “组织” (organization). It is in accordance with a common sense that the last characters are more important in identifying Chinese organization names. We also found a similar pattern when recognizing person names. For instance, some famous persons’ names, such as “蔡元培” (Yuanpei Cai), can be matched exactly and successfully recognized, while for the names of less well-known persons, the first character (i.e., surname) tends to be given more attention as we expected.

4.4.2 Coefficient of Focal Loss

We reported in Fig.5 the speeds of convergence (during the training process) versus different values of $\gamma$ used in the focal loss. If the value of $\gamma$ is set to zero, the focal loss will be reduced to the cross-entropy loss. When the cross-entropy loss is used, the model is still trapped at an extremely low performance even after 15 epochs, which indicates that such a loss is not good for the situation where a severe sample imbalance presents in the training set. Note that the NER models usually suffer from the sample imbalance because most candidates will be labeled as “None”. The model with the focal loss converges relatively faster because this loss will adaptively assign different update rates to misclassified samples according to how hard they are recognized. Although the model trained with focal loss did not outperform that with cross-entropy, the used focal loss does help to speed up the training process.

4.4.3 Decoding Threshold

We reported in Fig.6 the F1-scores under different settings of LEMON on the development set of the
Weibo NER dataset. LEMON-2 generally performs better than LEMON-0 and LEMON-1 because the features derived from the 1-prefix, 2-prefix, 1-suffix and 2-suffix matches are all useful for NER and they cannot be mixed into a single bucket as we described in Subsection 3.3.2.

We found that the value of the threshold $\gamma$ should be in the range of $[0.2, 0.3]$. As shown in Fig.6, if the value of $\gamma$ is greater than or equal to 2, the performance will be more sensitive to the values of the threshold $\gamma$. If $\gamma$ is less than 0.3 and $\gamma$ is set to 3 the performance will drop dramatically. One reasonable explanation is that the focal loss tends to update the parameters too aggressively by a larger learning step for the samples that are hard to be recognized, especially when the probabilities assigned to these samples are pretty low.

Fig.5. F1-scores versus different training epochs.

Fig.6. F1-scores versus different values of $\gamma$ (the decoding threshold). The horizontal line represents the results yielded by the models without the decoding step.
4.5 Ablation Study

We report in Table 8 the recall of out-of-vocabulary (OOV) achieved by LEMON in different situations where the results of word segmentation may be correct or incorrect and the lexicon could be used or not on the test set of OntoNotes-4. From these numbers, we can understand how the results of word segmentation and the presence of lexicon-based memory impact the performance of LEMON. The OOV recall rates are listed separately for different categories of named entities including “Organization”, “Location”, “Person”, “GPE” (geographic entity), and “None”.

| Category   | Segmentation | Lexicon | R (%) |
|------------|--------------|---------|-------|
| Organization | Incorrect   | w/o     | 48.7  |
|             | Correct      | w/o     | 65.8  |
|             | Correct      | with    | 67.1  |
| Location    | Incorrect   | w/o     | 14.8  |
|             | Correct      | w/o     | 14.3  |
|             | Correct      | with    | 29.7  |
| Person      | Incorrect   | w/o     | 69.6  |
|             | Correct      | w/o     | 89.3  |
|             | Correct      | with    | 93.1  |
| GPE         | Incorrect   | w/o     | 38.5  |
|             | Correct      | w/o     | 47.8  |
|             | Correct      | with    | 87.4  |
| None        | Incorrect   | w/o     | 47.9  |
|             | Correct      | w/o     | 68.9  |
|             | Correct      | with    | 81.0  |

Note: We report in this table the recall of out-of-vocabulary in different situations where the results of word segmentation are correct or incorrect and the lexicon could be used (“with”) or not (“w/o”). The best results are highlighted in bold.

The overall recall has increased from 85.25% to 87.62% after the lexicon-based memory is incorporated. Taking the categories of “GPE” and “Person” as examples, the OOV recall of “GPE” increases from 47.8% to 87.4%, and that of “Person” increases from 89.3% to 93.1%. It can be seen from Table 8 that correct word segmentation results help to improve the OOV recall generally, but a slight drop is observed for the “Location” category even though the gold word segmentation results are used. However, when the lexicon is used, the OOV recall for the “Location” category increases by about 15%.

Although we found that it boosts the performance for all the categories by making use of the lexicon-based memory and word segmentation results, the OOV recall for the “Location” category is still dissatisfied. One possible explanation is that this category is easily confused with the other categories. For example, “Russia” and “Britain” belong to the “GPE” category while “Europe”, “Middle East”, and “East Village” are included in the “Location” category in the training set.

5 Conclusions

In this study, we proposed LEMON, a fragment-based model enhanced by a lexicon-based memory for the Chinese named entity recognition task. The lexicon-based memory is introduced to derive position-dependent features, such as prefix and suffix, for each possible fragment in an input sentence, and such position-dependent features have been empirically proven to be effective in dealing with the out-of-vocabulary words. The experimental results demonstrated that the model leveraging the position-dependent features produced by the lexicon-based memory achieved state-of-the-art performance with an increase in the F1-score up to 3.2% over the baseline models on four different NER datasets.

It would be interesting to see whether existing pre-trained language models, such as ERNIE[55], XLNet[56], and ELMo[57], and knowledge graph enhanced variants like K-BERT[58], can be integrated and used to further improve the NER performance in the proposed approach. Although we tested several configurations in which a pre-trained BERT was taken as the base model for the character encoder, in this study we would like to focus on how to incorporate a lexicon for NER via the memory network in a “soft” manner. We believe that this work is a complement to those of pre-trained language representations, and leave searching for the optimal combination as future work.

Acknowledgement The authors would like to thank the anonymous reviewers for their valuable comments.

Conflict of Interest The authors declare that they have no conflict of interest.

References

[1] Kapetanios E, Tatar D, Sacarea C. Natural Language Processing: Semantic Aspects. CRC Press, 2013. DOI: 10.1201/b15472.

[2] Lin D K, Wu X Y. Phrase clustering for discriminative learning. In Proc. the 2009 Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the
Nothman J, Ringland N, Radford W, Murphy T, Curran J R. Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, 2013, 194: 151–175. DOI: 10.1016/j.artint.2012.03.006.

Collobert R, Weston J, Bottoni L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 2011, 12: 2493–2537.

Chiu J P C, Nichols E. Named entity recognition with bidirectional LSTM-CRF models for sequence tagging. arXiv: 1508.01991, 2015. https://arxiv.org/abs/1508.01991, Oct. 2023.

Lample G, Ballesteros M, Subramanian S, Kawakami K, Dyer C. Neural architectures for named entity recognition. In *Proc. the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Jun. 2016, pp.260–270. DOI: 10.18653/v1/N16-1030.

Huang S, Sun X, Wang H F. Addressing domain adaptation for Chinese word segmentation with global recurrent structure. In *Proc. the 8th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Nov. 2017, pp.184–193. DOI: 10.1007/978-3-030-01716-3_3.

Chen L Z, Moschitti A. Learning to progressively recognize new named entities with sequence to sequence models. In *Proc. the 27th International Conference on Computational Linguistics*, Aug. 2018, pp.2181–2191.

Zhang C Y, Bengio S, Hardt M, Recht B, Vinyals O. Understanding deep learning requires rethinking generalization. arXiv: 1611.03530, 2017. https://arxiv.org/abs/1611.03530, Oct. 2023.

He J Z, Wang H F. Chinese named entity recognition and word segmentation based on character. In *Proc. the 6th SIGHAN Workshop on Chinese Language Processing*, Jan. 2008, pp.128–132.

Liu Z X, Zhu C H, Zhao T J. Chinese named entity recognition with a sequence labeling approach: Based on characters, or based on words? In *Proc. the 2010 Advanced Intelligent Computing Theories and Applications, and the 6th International Conference on Intelligent Computing*, Aug. 2010, pp.634–640.

Zhang Y, Yang J. Chinese NER using lattice LSTM. In *Proc. the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Jul. 2018, pp.1554–1564. DOI: 10.18653/v1/P18-1144.

Li J, Sun A X, Han J L, Li C L. A survey on deep learning for named entity recognition. *IEEE Trans. Knowledge and Data Engineering*, 2022, 34(1): 50–70. DOI: 10.1109/TKDE.2020.2981314.

Xu M B, Jiang H, Watcharawittayakul S. A local detection approach for named entity recognition and mention detection. In *Proc. the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, July 2017, pp.1237–1247. DOI: 10.18653/v1/P17-1114.

Chiu J P C, Nichols E. Named entity recognition with bidirectional LSTM-CNNs. *Trans. Association for Computational Linguistics*, 2016, 4: 357–370. DOI: 10.1162/tacl_a_00104.

Hassabis D, Kumaran D, Summerfield C, Botvinick M. Neuroscience-inspired artificial intelligence. *Neuron*, 2017, 95(2): 245–258. DOI: 10.1016/j.neuron.2017.06.011.

Hopfield J J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 1982, 79(8): 2554–2558. DOI: 10.1073/pnas.79.8.2554.

Anderson J R, Bothell D, Byrne M D, Douglass S, Lebiere C, Qin Y L. An integrated theory of the mind. *Psychological Review*, 2004, 111(4): 1036–1060. DOI: 10.1037/0033-295X.111.4.1036.

Shallice T. From Neuropsychology to Mental Structure. Cambridge University Press, 1988. DOI: 10.1017/CBO9780511526817.

Yadav V, Bethard S. A survey on recent advances in named entity recognition from deep learning models. In *Proc. the 27th International Conference on Computational Linguistics*, Aug. 2018, pp.2145–2158.

Zhang S L, Jiang H, Xu M B, Hou J F, Dai L R. The fixed-size ordinarily-forgetting encoding method for neural network language models. In *Proc. the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, July 2015, pp.495–500. DOI: 10.3115/v1/P15-2081.

Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv: 1409.0473, 2016. https://arxiv.org/abs/1409.0473, Oct. 2023.

Luong T, Pham H, Manning C D. Effective approaches to attention-based neural machine translation. In *Proc. the 2015 Conference on Empirical Methods in Natural Language Processing*, Sept. 2015, pp.1412–1421. DOI: 10.18653/v1/D15-1166.

Rei M, Crichton G, Pyysalo S. Attending to characters in neural sequence labeling models. In *Proc. the 26th International Conference on Computational Linguistics: Technical Papers*, Dec. 2016, pp.309–318.

Xu G H, Wang C Y, He X F. Improving clinical named entity recognition with global neural attention. In *Proc. the 2nd International Joint Conference on Web and Big Data*, July 2018, pp.264–279. DOI: 10.1007/978-3-319-96893-2_20.

Zhang Q, Fu J L, Liu X Y, Huang X J. Adaptive co-attention network for named entity recognition in tweets. In *Proc. the 32nd AAAI Conference on Artificial Intelligence and the 30th Innovative Applications of Artificial Intelligence Conference and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence*, Feb. 2018, pp.5674–5681.
[28] Hammerton J. Named entity recognition with long short-term memory. In Proc. the 7th Conference on Natural Language Learning at HLT-NAACL 2003, May 2003, pp.172–175.

[29] Peng N Y, Dredze M. Improving named entity recognition for Chinese social media with word segmentation representation learning. In Proc. the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Aug. 2016, pp.149–155. DOI: 10.18653/v1/P16-2025.

[30] Yang J, Teng Z Y, Zhang M S, Zhang Y. Combining discrete and neural features for sequence labeling. In Proc. the 17th International Conference on Computational Linguistics and Intelligent Text Processing, Mar. 2018, pp.140–154. DOI: 10.1007/978-3-319-75477-2_9.

[31] Xue N W, Shen L B. Chinese word segmentation as LMR tagging. In Proc. the 2nd SIGHAN Workshop on Chinese Language Processing, July 2003, pp.176–179. DOI: 10.3115/1119250.1119278.

[32] Elman J L. Finding structure in time. Cognitive Science, 1990, 14(2): 179–211. DOI: 10.1207/s15516709cog1402_1.

[33] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[34] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L, Polosukhin I. Attention is all you need. In Proc. the 31st International Conference on Neural Information Processing Systems, Dec. 2017, pp.6000–6010. DOI: 10.5555/3295222.3295349.

[35] Joulin A, Grave E, Bojanowski P, Mikolov T. Bag of tricks for efficient text classification. In Proc. the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, Apr. 2017, pp.427–431.

[36] Conneau A, Kruszewski G, Lample G, Barrault L, Baroni M. What you can cramo into a single $\texttt{WordPiece}$ vector: Probing sentence embeddings for linguistic properties. In Proc. the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), July 2018, pp.2126–2136. DOI: 10.18653/v1/P18-1198.

[37] Mikolov T, Sutskever I, Chen K, Corrado G, Dean J. Distributed representations of words and phrases and their compositionality. In Proc. the 26th International Conference on Neural Information Processing Systems, Dec. 2013, pp.3111–3119.

[38] Suhbaatar S, Szlam A, Weston J, Fergus R. End-to-end memory networks. In Proc. the 28th International Conference on Neural Information Processing Systems, Dec. 2015, pp.2440–2448.

[39] Lin T Y, Goyal P, Girshick R, He K M, Dollár P. Focal loss for dense object detection. In Proc. the 2017 IEEE International Conference on Computer Vision, Oct. 2017, pp.2999–3007. DOI: 10.1109/ICCV.2017.324.

[40] Welchedel R, Palmer M, Marcus M, Hovy E, Pradhan S, Ramshaw L, Xue N W, Taylor A, Kaufman J, Franchini M, El-Bachouti M, Belvin R, Houston A. OntoNotes release 4.0. Linguistic Data Consortium, Philadelphia, 2011. DOI: 10.35111/gfjf-7r50.

[41] Levow G A. The third international Chinese language processing Bakeoff: Word segmentation and named entity recognition. In Proc. the 5th SIGHAN Workshop on Chinese Language Processing, July 2006, pp.108–117.

[42] Peng N Y, Dredze M. Named entity recognition for Chinese social media with jointly trained embeddings. In Proc. the 2015 Conference on Empirical Methods in Natural Language Processing, Sept. 2015, pp.548–554. DOI: 10.18653/v1/D15-1064.

[43] Sun M S, Chen X X, Zhang K X, Guo Z P, Liu Z Y. THULAC: An efficient lexical analyzer for Chinese. Technical Report, Tsinghua University, 2016. https://nlp.csai.tsinghua.edu.cn/project/thulac/, Oct. 2023. (in Chinese)

[44] Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z M, Desmaison A, Antiga L, Lerer A. Automatic differentiation in PyTorch. In Proc. the 31st Conference on Neural Information Processing Systems, Dec. 2017.

[45] Kingma D P, Ba J. Adam: A method for stochastic optimization. In Proc. the 3rd International Conference on Learning Representations, May 2015.

[46] Devlin J, Chang M W, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proc. the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Jun. 2019, pp.4171–4186. DOI: 10.18653/v1/N19-1423.

[47] Yang J, Zhang Y. NCRF++: An open-source neural sequence labeling toolkit. In Proc. the ACL 2018, System Demonstrations, July 2018, pp.74–79. DOI: 10.18653/v1/P18-4013.

[48] Chen A T, Peng F C, Shan R, Sun G. Chinese named entity recognition with conditional probabilistic models. In Proc. the 5th SIGHAN Workshop on Chinese Language Processing, July 2006, pp.173–176.

[49] Zhang S X, Qin Y, Wen J, Wang X J. Word segmentation and named entity recognition for SIGHAN Bakeoff3. In Proc. the 5th SIGHAN Workshop on Chinese Language Processing, July 2006, pp.158–161.

[50] Lu Y, Zhang Y, Ji D H. Multi-prototype Chinese character embedding. In Proc. the 10th International Conference on Language Resources and Evaluation, May 2016, pp.855–859.

[51] Dong C H, Zhang J J, Zong C Q, Hattori M, Di H. Character-based LSTMCRF with radical-level features for Chinese named entity recognition. In Natural Language Understanding and Intelligent Applications, Lin C Y, Xue N W, Zhao D Y, Huang X J, Feng Y S (eds.), Springer, 2016, pp.239–250. DOI: 10.1007/978-3-319-50496-4_20.

[52] Wang M Q, Che W X, Manning C D. Effective bilingual constraints for semi-supervised learning of named entity recognizers. In Proc. the 27th AAAI Conference on Artificial Intelligence, July 2013, pp.919–925.

[53] Che W X, Wang M Q, Manning C D, Liu T. Named entity recognition with bilingual constraints. In Proc. the
Yi Zhou et al.: Chinese Named Entity Recognition Augmented with Lexicon Memory

Yi Zhou is a post-graduate student at School of Computer Science, Fudan University, Shanghai. He received his Bachelor’s degree in computer science from Fudan University, Shanghai, in 2018. His research interests include natural language processing, deep learning, and machine learning.

Xiao-Qing Zheng is an associate professor of School of Computer Science, Fudan University, Shanghai. He received his Ph.D. degree in computer science from Zhejiang University, Hangzhou, in 2007. He was an international faculty fellow at the Sloan School of Management, Massachusetts Institute of Technology, Boston. His research interests include natural language processing, machine learning, and semantic Web.

Xuan-Jing Huang is a professor of School of Computer Science, Fudan University, Shanghai. She received her Ph.D. degree in computer science from Fudan University, Shanghai, in 1998. Her research interest includes artificial intelligence, natural language processing, information retrieval, and social media processing. She served as the PC Co-Chair of EMNLP 2021, CCL 2019, CCL 2016, NLPCC 2017, SMP 2015, the organizer of WSDM 2015, and the competition chair of CIKM 2014. She has been included in the 2020 Women in AI List and AI 2000 Most Influential Scholar Annual List.