Chinese Relation Extraction with Multi-Grained Information and External Linguistic Knowledge

Ziran Li[1,2*], **Ning Ding**[1,2*], **Zhiyuan Liu**[2,3,4], **Hai-Tao Zheng**[1,2†], **Ying Shen**[5]
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[3]Institute for Artificial Intelligence, Tsinghua University, Beijing, China
[4]State Key Lab on Intelligent Technology and Systems, Tsinghua University, Beijing, China
[5]School of Electronics and Computer Engineering, Peking University Shenzhen Graduate School
{lizr18,dingn18}@mails.tsinghua.edu.cn

Abstract

Chinese relation extraction is conducted using neural networks with either character-based or word-based inputs, and most existing methods typically suffer from segmentation errors and ambiguity of polysemy. To address the issues, we propose a multi-grained lattice framework (MG lattice) for Chinese relation extraction to take advantage of multi-grained language information and external linguistic knowledge. In this framework, (1) we incorporate word-level information into character sequence inputs so that segmentation errors can be avoided. (2) We also model multiple senses of polysemous words with the help of external linguistic knowledge, so as to alleviate polysemy ambiguity. Experiments on three real-world datasets in distinct domains show consistent and significant superiority and robustness of our model, as compared with other baselines. The source code of this paper can be obtained from https://github.com/thunlp/Chinese_NRE.

1 Introduction

Relation extraction (RE) has a pivotal role in information extraction (IE), aiming to extract semantic relations between entity pairs in natural language sentences. In downstream applications, this technology is a key module for constructing large-scale knowledge graphs. Recent developments in deep learning have heightened the interest for neural relation extractions (NRE), which attempt to use neural networks to automatically learn semantic features (Liu et al., 2013; Zeng et al., 2014, 2015; Lin et al., 2016; Zhou et al., 2016; Jiang et al., 2016).

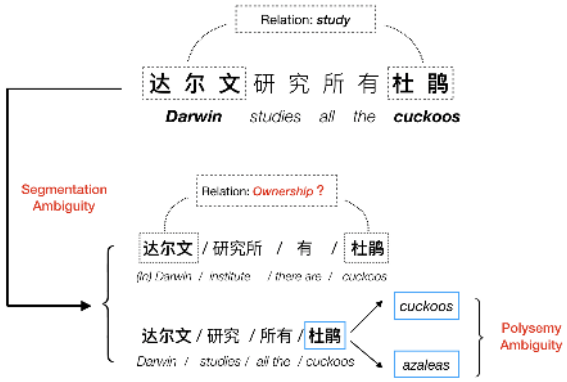


Figure 1: An example of segmentation ambiguity and polysemy ambiguity in Chinese RE.

Although it is not necessary for NRE to perform feature engineering, they ignore the fact that different language granularity of input will have a significant impact on the model, especially for Chinese RE. Conventionally, according to the difference in granularity, most existing methods for Chinese RE can be divided into two types: character-based RE and word-based RE.

For the character-based RE, it regards each input sentence as a character sequence. The shortcoming of this kind of method is that it cannot fully exploit word-level information, capturing fewer features than the word-based methods. For the word-based RE, word segmentation should be first performed. Then, a word sequence is derived and fed into the neural network model. However, the performance of the word-based models could be significantly impacted by the quality of segmentation.

For example, as shown in Fig 1, the Chinese sentence "达尔文研究所有杜鹃 (Darwin studies all the cuckoos)" has two entities, which are "达尔文 (Darwin)" and "杜鹃 (cuckoos)", and the relation between them is *Study*. In this case, the

* indicates equal contribution
† Corresponding author: Hai-Tao Zheng. (E-mail: zheng.haitao@sz.tsinghua.edu.cn)

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4377–4386
Florence, Italy, July 28 - August 2, 2019.

correct segmentation is “达尔文 (Darwin) / 研究 (studies) / 所有 (all the) / 杜鹃 (cuckoos)” . Nevertheless, semantics of the sentence could become entirely different as the segmentation changes. If the segmentation is “达尔文 (In Darwin) / 研究所 (institute) / 有 (there are) / 杜鹃 (cuckoos)”, the meaning of the sentence becomes ’there are cuckoos in Darwin institute’ and the relation between “达尔文 (Darwin)” and “杜鹃 (cuckoos)” turns into *Ownership*, which is wrong. Hence, neither character-based methods nor word-based methods can sufficiently exploit the semantic information in data. Worse still, this problem becomes severer when datasets is finely annotated, which are scarce in number. Obviously, to discover high-level entity relationships from plain texts, we need the assistance of comprehensive information with various granularity.

Furthermore, the fact that there are many polysemous words in datasets is another point neglected by existing RE models, which limits the ability of the model to explore deep semantic features. For instance, the word “杜鹃” has two different senses, which are ’cuckoos’ and ’azaleas’. But it’s difficult to learn both senses information from plain texts without the help of external knowledge. Therefore, the introduction of external linguistic knowledge will be of great help to NRE models.

In this paper, we proposed the multi-granularity lattice framework (MG lattice), a unified model comprehensively utilizes both internal information and external knowledge, to conduct the Chinese RE task. (1) The model uses a lattice-based structure to dynamically integrate word-level features into the character-based method. Thus, it can leverage multi-granularity information of inputs without suffering from segmentation errors. (2) Moreover, to alleviate the issue of polysemy ambiguity, the model utilizes HowNet (Dong and Dong, 2003), which is an external knowledge base manually annotates polysemous Chinese words. Then, the senses of words are automatically selected during the training stage and consequently, the model can fully exploit the semantic information in data for better RE performance.

Sets of experiments has been conducted on three manually labeled RE datasets. The results indicate that our model significantly outperforms multiple existing methods, achieving state-of-the-art results on various datasets across different domains.

2 Related Work

Recent years RE, especially NRE, has been widely studied in the NLP field. As a pioneer, (Liu et al., 2013) proposed a simple CNN RE model and it is regarded as one seminal work that uses a neural network to automatically learn features. On this basis, (Zeng et al., 2014) developed a CNN model with max-pooling, where positional embeddings were first used to represent the position information. Then the PCNNs model (Zeng et al., 2015) designed the multi-instance learning paradigm for RE. However, the PCNNs model suffers the issue of the selection of sentences. To address the problem, Lin et al. (2016) applied the attention mechanism over all the instances in the bag. Further, Jiang et al. (2016) proposed a model with multi-instance and multi-label paradigms. Although PCNNs models are more efficient, they cannot exploit contextual information like RNNs. Hence, LSTM with attention mechanism was also applied to the RE task (Zhang and Wang, 2015; Zhou et al., 2016; Lee et al., 2019).

Existing methods for Chinese RE are mostly character-based or word-based implementations of mainstream NRE models (Chen and Hsu, 2016; Rönnqvist et al., 2017; ZHANG et al., 2017; Xu et al., 2017). In most cases, these methods only focus on the improvement of the model itself, ignoring the fact that different granularity of input will have a significant impact on the RE models. The character-based model can not utilize the information of words, capturing fewer features than the word-based model. On the other side, the performance of the word-based model is significantly impacted by the quality of segmentation (Zhang and Yang, 2018). Although some methods are used to combine character-level and word-level information in other NLP tasks like character-bigrams (Chen et al., 2015; Yang et al., 2017) and soft words (Zhao and Kit, 2008; Chen et al., 2014; Peng and Dredze, 2016), the information utilization is still very limited.

Then, tree-structured RNNs was proposed to address the shortcomings. Tai et al. (2015) proposed a tree-like LSTM model to improve the semantic representation. This type of structure has been applied into various tasks, including human action recognition (Sun et al., 2017), NMT encoders (Su et al., 2017), speech tokenization

(Sperber et al., 2017) and NRE (Zhang and Yang, 2018). Although the lattice LSTM model can exploit word and word sequence information, it still could be severely affected by the ambiguity of polysemy. In other words, these models cannot handle the polysemy of words with the change of language situation. Therefore, the introduction of external linguistic knowledge is very necessary. We utilize sense-level information with the help of HowNet proposed by Dong and Dong (2003), which is a concept knowledge base that annotates Chinese with correlative word senses. In addition, the open-sourced HowNet API (Qi et al., 2019) is also used in our work.

3 Methodology

Given a Chinese sentence and two marked entities in it, the task of Chinese relation extraction is to extract semantic relations between the two entities. In this section, we present our MG lattice model for Chinese relation extraction in detail. As shown in Fig 2, the model could be introduced from three aspects:

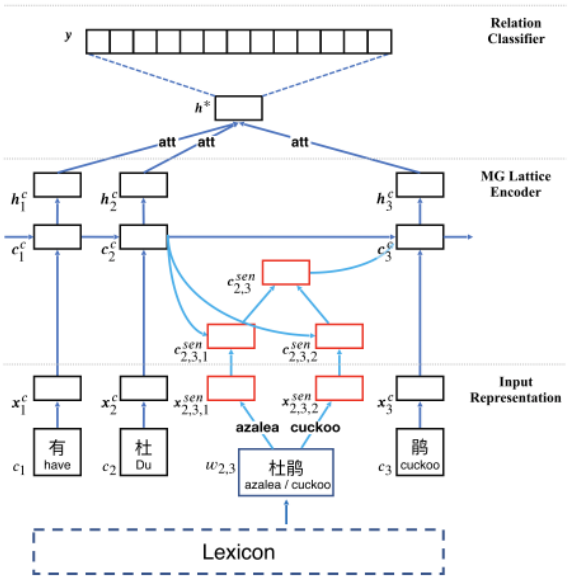


Figure 2: MG lattice framework. [1]

Input Representation. Given a Chinese sentence with two target entities as input, this part represents each word and character in the sentence. Then the model can utilize both word-level and character-level information.

MG Lattice Encoder. Incorporating external knowledge into word sense disambiguation, this part uses a lattice-structure LSTM network to construct a distributed representation for each input instance.

Relation Classifier. After the hidden states are learned, a character-level mechanism is adapted to merge features. Then the final sentence representations are fed into a softmax classifier to predict relations.

We will introduce all the three parts in the following subsections in detail.

3.1 Input Representation

The input of our model is a Chinese sentence s with two marked entities. In order to utilize multi-granularity information, we represent both characters and words in the sentence.

3.1.1 Character-level Representation

Our model takes character-based sentences as direct inputs, that is, regarding each input sentence as a character sequence. Given a sentence s consisting of M characters $s = \{c_1, ..., c_M\}$, we first map each character c_i to a vector of d^c dimensions, denoted as $x_i^{ce} \in \mathbb{R}^{d^c}$, via the Skip-gram model (Mikolov et al., 2013).

In addition, we leverage position embeddings to specify entity pairs, which are defined as the relative distances from the current character to head and tail entities (Zeng et al., 2014). Specifically, the relative distances from the i-th character c_i to the two marked entities are denoted as p_i^1 and p_i^2 respectively. We calculate p_i^1 as below:

$$p_i^1 = \begin{cases} i - b^1 & i < b^1, \\ 0 & b^1 \le i \le e^1, \\ i - e^1 & i > e^1, \end{cases} \quad (1)$$

where b^1 and e^1 are the start and end indices of the head entity. The computation of p_i^2 is similar to Eq. 1. Then, p_i^1 and p_i^2 are transformed into two corresponding vectors, denoted as $\boldsymbol{x}_i^{p_1} \in \mathbb{R}^{d^p}$ and $\boldsymbol{x}_i^{p_2} \in \mathbb{R}^{d^p}$, by looking up a position embedding table.

Finally, the input representation for character c_i, denoted as $\boldsymbol{x}_i^c \in \mathbb{R}^d$ ($d = d^c + 2 \times d^p$), is concatenated by character embedding $\boldsymbol{x}_i^{ce}$, position embeddings $\boldsymbol{x}_i^{p_1}$ and $\boldsymbol{x}_i^{p_2}$:

$$\boldsymbol{x}_i^c = [\boldsymbol{x}_i^{ce}; \boldsymbol{x}_i^{p_1}; \boldsymbol{x}_i^{p_2}]. \quad (2)$$

Then, the representation of characters $\boldsymbol{x}^c = \{\boldsymbol{x}_1^c, ..., \boldsymbol{x}_M^c\}$ will be directly fed into our model.

[1] In order to keep the figure clear and concise, we do not show gate cells and the backward direction.

3.1.2 Word-level Representation

Although our model takes character sequences as direct inputs, in order to fully capture word-level features, it also needs the information of all potential words in the input sentences. Here, a potential word is any character subsequence that matches a word in a lexicon $\mathbb{D}$ built over segmented large raw text. Let $w_{b,e}$ be such a subsequence starting from the b-th character to the e-th character. To represent $w_{b,e}$, we use the word2vec (Mikolov et al., 2013) to convert it into a real-valued vector $\boldsymbol{x}^w_{b,e} \in \mathbb{R}^{d^w}$.

However, the word2vec method maps each word to only one single embedding, ignoring the fact that many words have multiple senses. To tackle this problem, we incorporate HowNet as an external knowledge base into our model to represent word senses rather than words.

Hence, given a word $w_{b,e}$, we first obtain all K senses of it by retrieving the HowNet. Using $Sense(w_{b,e})$ to denote the senses set of $w_{b,e}$, we then convert each sense $sen_k^{(w_{b,e})} \in Sense(w_{b,e})$ into a real-valued vector $\boldsymbol{x}^{sen}_{b,e,k} \in \mathbb{R}^{d^{sen}}$ through the SAT model (Niu et al., 2017). The SAT model is on the basis of the Skip-gram, which can jointly learn word and sense representations. Finally, the representation of $w_{b,e}$ is a vector set denoted as $x^{sen}_{b,e} = \{\boldsymbol{x}^{sen}_{b,e,1}, ..., \boldsymbol{x}^{sen}_{b,e,K}\}$.

In the next section, we will introduce how our model utilizes sense embeddings.

3.2 Encoder

The direct input of the encoder is a character sequence, together with all potential words in lexicon $\mathbb{D}$. After training, the output of the encoder is the hidden state vectors $\boldsymbol{h}$ of an input sentence. We introduce the encoder with two strategies, including **the basic lattice** LSTM and the **multi-graind lattice** (MG lattice) LSTM.

3.2.1 Basic Lattice LSTM Encoder

Generally, a classical LSTM (Hochreiter and Schmidhuber, 1997) unit is composed of four basic gates structure: one input gate $\boldsymbol{i}_j$ controls which information enters into the unit; one output gate $\boldsymbol{o}_j$ controls which information would be outputted from the unit; one forget gate $\boldsymbol{f}_j$ controls which information would be removed in the unit. All three gates are accompanied by weight matrix $\boldsymbol{W}$. Current cell state $\boldsymbol{c}_j$ records all historical information flow up to the current time. Therefore, the character-based LSTM functions are:

$$\begin{cases} \boldsymbol{i}^c_j = \sigma(W_i \boldsymbol{x}^c_j + U_i \boldsymbol{h}^c_{j-1} + \boldsymbol{b}_i), \\ \boldsymbol{o}^c_j = \sigma(W_o \boldsymbol{x}^c_j + U_o \boldsymbol{h}^c_{j-1} + \boldsymbol{b}_o), \\ \boldsymbol{f}^c_j = \sigma(W_f \boldsymbol{x}^c_j + U_f \boldsymbol{h}^c_{j-1} + \boldsymbol{b}_f), \\ \tilde{\boldsymbol{c}}^c_j = \tanh(W_c \boldsymbol{x}^c_j + U_c \boldsymbol{h}^c_{j-1} + \boldsymbol{b}_c), \end{cases} \tag{3}$$

$$\boldsymbol{c}^c_j = \boldsymbol{f}^c_j \odot \boldsymbol{c}^c_{j-1} + \boldsymbol{i}^c_j \odot \tilde{\boldsymbol{c}}^c_j, \tag{4}$$

$$\boldsymbol{h}^c_j = \boldsymbol{o}^c_j \odot \tanh(\boldsymbol{c}^c_j), \tag{5}$$

where $\sigma()$ means the sigmoid function. Hence, the current cell state $\boldsymbol{c}_j$ will be generated by calculating the weighted sum using both previous cell state and current information generated by the cell (Graves, 2013).

Given a word $w_{b,e}$ in the input sentence which matches the external lexicon $\mathbb{D}$, the representation can be obtained as follows:

$$\boldsymbol{x}^w_{b,e} = \boldsymbol{e}^w(w_{b,e}), \tag{6}$$

where b and e denotes the start and the end of the word, and e^w is the lookup table . Under this circumstance, the computation of $\boldsymbol{c}^c_j$ incorporates word-level representation $\boldsymbol{x}^w_{b,e}$ to construct the basic lattice LSTM encoder. Further, a word cell $\boldsymbol{c}^w_{b,e}$ is used to represent the memory cell state of $\boldsymbol{x}^w_{b,e}$. The computation of $\boldsymbol{c}^w_{b,e}$ is:

$$\begin{cases} \boldsymbol{i}^w_{b,e} = \sigma(W_i \boldsymbol{x}^w_{b,e} + U_i \boldsymbol{h}^c_b + \boldsymbol{b}_i), \\ \boldsymbol{f}^w_{b,e} = \sigma(W_f \boldsymbol{x}^w_{b,e} + U_f \boldsymbol{h}^c_b + \boldsymbol{b}_f), \\ \tilde{\boldsymbol{c}}^w_{b,e} = \tanh(W_c \boldsymbol{x}^w_{b,e} + U_c \boldsymbol{h}^c_b + \boldsymbol{b}_c), \end{cases} \tag{7}$$

$$\boldsymbol{c}^w_{b,e} = \boldsymbol{f}^w_{b,e} \odot \boldsymbol{c}^c_b + \boldsymbol{i}^w_{b,e} \odot \tilde{\boldsymbol{c}}^w_{b,e}, \tag{8}$$

where $\boldsymbol{i}^w_{b,e}$ and $\boldsymbol{f}^w_{b,e}$ serve as a set of word-level input and forget gates.

The cell state of the e-th character will be calculated by incorporating the information of all the words that end in index e, which is $w^{b,e}$ with $b \in \{b' | w_{b',e} \in \mathbb{D}\}$. To control the contribution of each word, an extra gate $\boldsymbol{i}^c_{b,e}$ is used:

$$\boldsymbol{i}^c_{b,e} = \sigma(W \boldsymbol{x}^c_e + U \boldsymbol{c}^w_{b,e} + \boldsymbol{b}^l). \tag{9}$$

Then the cell value of the e-th character is computed by:

$$\boldsymbol{c}^c_e = \sum_{b \in \{b' | w_{b',e} \in \mathbb{D}\}} \boldsymbol{\alpha}^c_{b,e} \odot \boldsymbol{c}^w_{b,e} + \boldsymbol{\alpha}^c_e \odot \tilde{\boldsymbol{c}}^c_e, \tag{10}$$

where $\boldsymbol{\alpha}_{b,e}^{c}$ and $\boldsymbol{\alpha}_{e}^{c}$ are normalization factors, setting the sum to 1:

$$\boldsymbol{\alpha}_{b,e}^{c}=\frac{\exp(\boldsymbol{i}_{b,e}^{c})}{\exp(\boldsymbol{i}_{e}^{c})+\sum_{b'\in\{b''|w_{b'',e}\in\mathbb{D}\}}\exp(\boldsymbol{i}_{b',e}^{c})}, \quad (11)$$

$$\boldsymbol{\alpha}_{e}^{c}=\frac{\exp(\boldsymbol{i}_{e}^{c})}{\exp(\boldsymbol{i}_{e}^{c})+\sum_{b'\in\{b''|w_{b'',e}\in\mathbb{D}\}}\exp(\boldsymbol{i}_{b',e}^{c})}. \quad (12)$$

Finally, we use Eq. 5 to compute the final hidden state vectors $\boldsymbol{h}_{j}^{c}$ for each character of the sequence. This structure is also used in Zhang and Yang (2018).

3.2.2 MG Lattice LSTM Encoder

Although the basic lattice encoder can explicitly leverages character and word information, it could not fully consider the ambiguity of Chinese. For instance, as shown in Figure 2, the word $w_{2,3}$ (杜鹃) has two senses: $sen_{1}^{(w_{2,3})}$ represents 'azalea' and $sen_{2}^{(w_{2,3})}$ represents 'cuckoo', but there is only one representation for $w_{2,3}$ in the basic lattice encoder, which is $\boldsymbol{x}_{2,3}^{w}$.

To address this shortcoming, we improve the model by adding sense-level paths as external knowledge to the model. Hence, a more comprehensive lexicon would be constructed. As mentioned in 3.1, the representation of the k-th sense of the word $w_{b,e}$ is $\boldsymbol{x}_{b,e,k}^{sen}$.

For each word $w_{b,e}$ which matches the lexicon $\mathbb{D}$, we will take all its sense representations into the calculation. The computation of the k-th sense of word $w_{b,e}$ is:

$$\begin{cases} \boldsymbol{i}_{b,e,k}^{sen}=\sigma(W_i\boldsymbol{x}_{b,e,k}^{sen}+U_i\boldsymbol{h}_{b}^{c}+\boldsymbol{b}_i), \\ \boldsymbol{f}_{b,e,k}^{sen}=\sigma(W_f\boldsymbol{x}_{b,e,k}^{sen}+U_f\boldsymbol{h}_{b}^{c}+\boldsymbol{b}_f), \\ \tilde{\boldsymbol{c}}_{b,e,k}^{sen}=\tanh(W_c\boldsymbol{x}_{b,e,k}^{sen}+U_c\boldsymbol{h}_{b}^{c}+\boldsymbol{b}_c), \end{cases} \quad (13)$$

$$\boldsymbol{c}_{b,e,k}^{sen}=\boldsymbol{f}_{b,e,k}^{sen}\odot\boldsymbol{c}_{b}^{c}+\boldsymbol{i}_{b,e,k}^{sen}\odot\tilde{\boldsymbol{c}}_{b,e,k}^{sen}, \quad (14)$$

where $\boldsymbol{c}_{b,e,k}^{sen}$ represents the memory cell of the k-th sense of the word $w_{b,e}$. Then all the senses are merged into a comprehensive representation to compute the memory cell of $w_{b,e}$, which is denoted as $\boldsymbol{c}_{b,e}^{sen}$:

$$\boldsymbol{c}_{b,e}^{sen}=\sum_{k}\boldsymbol{\alpha}_{b,e,k}^{sen}\odot\boldsymbol{c}_{b,e,k}^{sen}, \quad (15)$$

$$\boldsymbol{\alpha}_{b,e,k}^{sen}=\frac{\exp(\boldsymbol{i}_{b,e,k}^{sen})}{\sum_{k'}^{K}\exp(\boldsymbol{i}_{b,e,k'}^{sen})}, \quad (16)$$

where $\boldsymbol{i}_{b,e,k}^{sen}$ is an extra gate to control the contribution of the k-th sense, and is computed similar as Eq. 9.

In this situation, all the sense-level cell states will be incorporated into the word representation $\boldsymbol{c}_{b,e}^{sen}$, which could better represent the polysemous word. Then, similar to Eq. 9 - 12, all the recurrent paths of words ending in index e will flow into the current cell $\boldsymbol{c}_{e}^{c}$:

$$\boldsymbol{c}_{e}^{c}=\sum_{b\in\{b'|w_{b',e}^{d}\in\mathbb{D}\}}\boldsymbol{\alpha}_{b,e}^{sen}\odot\boldsymbol{c}_{b,e}^{sen}+\boldsymbol{\alpha}_{e}^{c}\odot\tilde{\boldsymbol{c}}_{e}^{c}. \quad (17)$$

Finally, the hidden state $\boldsymbol{h}$ are still computed by Eq. 5 and then sent to the relation classifier.

3.3 Relation Classifier

After the hidden state of an instance $\boldsymbol{h}\in\mathbb{R}^{d^h\times M}$ is learnt, we first adopt a character-level attention mechanism to merge $\boldsymbol{h}$ into a sentence-level feature vector, denoted as $\boldsymbol{h}^{*}\in\mathbb{R}^{d^h}$. Here, d^h indicates the dimension of the hidden state and M is the sequence length. Then, the final sentence representation $\boldsymbol{h}^{*}$ is fed into a softmax classifier to compute the confidence of each relation.

The representation $\boldsymbol{h}^{*}$ of the sentence is computed as a weighted sum of all character feature vectors in $\boldsymbol{h}$:

$$\boldsymbol{H}=\tanh(\boldsymbol{h}), \quad (18)$$

$$\boldsymbol{\alpha}=\mathrm{softmax}(\boldsymbol{w}^{T}\boldsymbol{H}), \quad (19)$$

$$\boldsymbol{h}^{*}=\boldsymbol{h}\boldsymbol{\alpha}^{T}, \quad (20)$$

where $\boldsymbol{w}\in\mathbb{R}^{d^h}$ is a trained parameter and $\boldsymbol{\alpha}\in\mathbb{R}^{M}$ is the weight vector of $\boldsymbol{h}$.

To compute the conditional probability of each relation, the feature vector $\boldsymbol{h}^{*}$ of sentence S is fed into a softmax classifier:

$$\boldsymbol{o}=\boldsymbol{W}\boldsymbol{h}^{*}+\boldsymbol{b}, \quad (21)$$

$$p(\boldsymbol{y}|S)=\mathrm{softmax}(\boldsymbol{o}), \quad (22)$$

where $\boldsymbol{W}\in\mathbb{R}^{Y\times d^h}$ is the transformation matrix and $\boldsymbol{b}\in\mathbb{R}^{Y}$ is a bias vector. Y indicates the total number of relation types, and $\boldsymbol{y}$ is the estimated probability for each type. This mechanism is also applied to (Zhou et al., 2016).

Finally, given all (T) training examples $(S^{(i)},y^{(i)})$, we define the objective function using cross-entropy as follows:

$$J(\theta)=\sum_{i=1}^{T}\log p(y^{(i)}|S^{(i)},\theta), \quad (23)$$

where θ indicates all parameters of our model.

To avoid co-adaptation of hidden units, we apply dropout (Hinton et al., 2012) on the LSTM layer by randomly removing feature detectors from the network during forward propagation.

4 Experiments

In this section, we conduct a series of experiments on three manually labeled datasets. Our models show superiority and effectiveness compared with other models. Furthermore, generalization is another advantage of our models, because there are five corpora used to construct the three datasets, which are entirely different in topics and manners of writing. The experiments will be organized as follows:

(1) First, we study the ability of our model to combine character-level and word-level information by comparing it with char-based and word-based models;

(2) Then we focus on the impact of sense representation, carrying out experiments among three different kinds of lattice-based models;

(3) Finally, we make comparisons with other proposed models in relation extraction task.

4.1 Datasets and Experimental Settings

Datasets. We carry out our experiments on three different datasets, including Chinese SanWen (Xu et al., 2017), ACE 2005 Chinese corpus (LDC2006T06) and FinRE.

The Chinese SanWen dataset contains 9 types of relations among 837 Chinese literature articles, in which 695 articles for training, 84 for testing and the rest 58 for validating. The ACE 2005 dataset is collected from newswires, broadcasts, and weblogs, containing 8023 relation facts with 18 relation subtypes. We randomly select 75% of it to train the models and the remaining is used for evaluation.

For more diversity in test domains, we manually annotate the FinRE dataset from 2647 financial news in Sina Finance [2], with 13486, 3727 and 1489 relation instances for training, testing and validation respectively. The FinRE contains 44 distinguished relationships including a special relation NA, which indicates that there is no relation between the marked entity pair.

Evaluation Metrics. Multiple standard evaluation metrics are applied in the experiments, including the precision-recall curve, F1-score, Precision at top N predictions (P@N) and area under the curve (AUC). With comprehensive evaluations, models can be estimated from multiple angles.

[2]https://finance.sina.com.cn/

Hyper-parameter	value
learning rate	0.0005
dropout probability	0.5
char embedding size	100
lattice embedding size	200
position embedding size	5
LSTM hidden	200
regularization	1e-8

Table 1: Hyper-parameters

Parameter Settings. We tune the parameters of our models by grid searching on the validation dataset. Grid search is utilized to select optimal learning rate λ for Adam optimizer (Kingma and Ba, 2014) among $\{0.0001, 0.0005, 0.001, 0.005, \}$ and position embedding d_p in $\{5, 10, 15, 20\}$. Table 1 shows the values of the best hyper-parameters in our experiments. The best models were selected by early stopping using the evaluation results on the validation dataset. For other parameters, we follow empirical settings because they make little influence on the whole performance of our models.

Models		FinRE	SanWen	ACE
Word-based	Word-baseline	41.23	54.26	64.43
	+char CNN	41.60	56.62	68.86
	+char LSTM	42.20	57.92	69.81
Char-based	Character-baseline	40.50	60.34	71.52
	+softword	41.42	60.69	69.81
	+bichar	40.52	61.34	71.86
	+softword + bichar	42.03	61.75	72.63
Ours	Basic Lattice	47.41	63.88	77.12
	MG Lattice	**49.26**	**65.61**	**78.17**

Table 2: F1-scores of word-baselines, character baselines and lattice-based models on all datasets.

4.2 Effect of Lattice Encoder.

In this part, we mainly focus on the effect of the encoder layer. As shown in Table 2, we conducted experiments on char-based, word-based and lattice-based models on all datasets. The word-based and character-based baselines are implemented by replacing the lattice encoder with a bidirectional LSTM. In addition, character and word features are added to these two baselines respectively, so that they can use both character and word information. For word baseline, we utilize

Datasets	ACE-2005				SanWen				FinRE			
P@N	100	200	300	Mean	100	200	300	Mean	100	200	300	Mean
Basic Lattice	**99.01**	94.03	94.68	95.91	**96.04**	90.05	89.04	91.71	97.03	92.04	**90.70**	93.26
Basic Lattice (SAT)	97.03	97.01	96.01	96.69	93.07	93.03	**91.36**	92.49	98.02	**93.03**	90.70	93.92
MG Lattice	98.02	**97.51**	**96.01**	**97.18**	94.06	**93.03**	90.70	**92.60**	**100.0**	92.54	89.70	**94.08**

Table 3: Precision@N of lattice-based models on all datasets.

an extra CNN/LSTM to learn hidden states for characters of each word (char CNN/LSTM). For char baseline, bichar and softword (word in which the current character is located) are used as word-level features to improve character representation.

The lattice-based approaches include two lattice-based models, and both of them can explicitly leverage both character and word information. The basic lattice uses the encoder mentioned in 3.2.1, which can dynamically incorporate word-level information into character sequences. For MG lattice, each sense embedding will be used to construct an independent sense path. Hence, there is not only word information, but also sense information flowing into cell states.

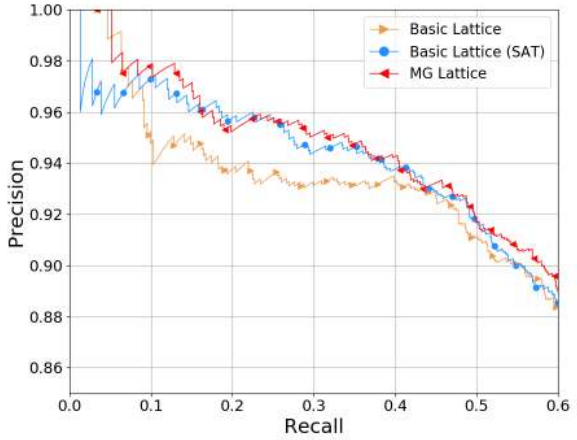


Figure 3: Precision-recall curves for three lattice-based models on ACE-2005.

Results of word-based model. With automatic word segmentation, the baseline of the word-based model yields 41.23%, 54.26% and 64.43% F1-score on three datasets. The F1-scores are increased to 41.6%, 56.62 and 68.86% by adding character CNN to the baseline model. Compared with the character CNN, character LSTM representation gives slightly higher F1-scores, which are 42.2%, 57.92%, and 69.81% respectively. The results indicate that character information will promote the performance of the word-based model, but the increase in F1-score is not significant.

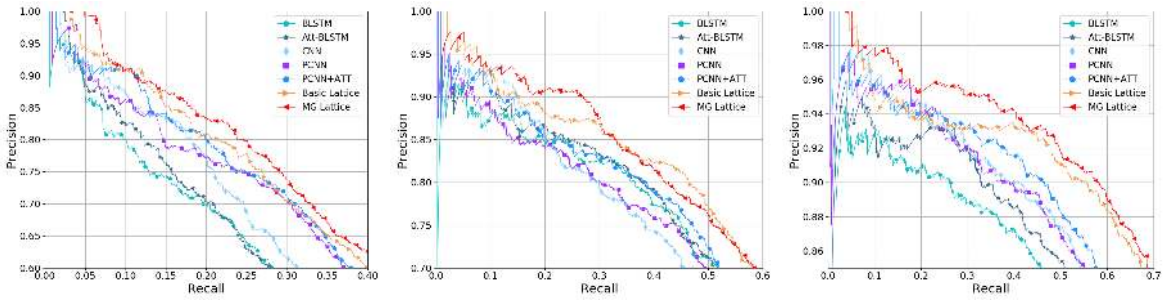
Results of character-based model. For the character baseline, it gives higher F1-scores compared with the word-based methods. By adding soft word feature, the F1-scores slightly increase on FinRE and SanWen dataset. Similar results are achieved by adding character-bigram. Additionally, a combination of both word features yields best F1-scores among character-based models, which are 42.03%, 61.75%, and 72.63%.

Results of lattice-based model. Although we take multiple strategies to combine character and word information in baselines, the lattice-based models still significantly outperform them. The basic lattice model improves the F1-scores of three datasets from 42.2% to 47.35%, 61.75% to 63.88% and 72.63% to 77.12% respectively. The results demonstrate the ability to exploit character and word sequence information of the lattice-based model. Comparisons and analysis of the lattice-based models will be introduced in the next subsection.

4.3 Effect of Word Sense Representations

In this section, we will study the effect of word sense representations by utilizing sense-level information with different strategies. Hence, three types of lattice-based models are used in our experiments. First, the basic lattice model uses word2vec (Mikolov et al., 2013) to train the word embeddings, which considers no word sense information. Then, we introduce the basic lattice (SAT) model as a comparison, for which the pre-trained word embeddings are improved by sense information (Niu et al., 2017). Moreover, the MG lattice model uses sense embeddings to build independent paths and dynamically selects the appropriate sense.

The results of P@N shown in Table 3 demonstrate the effectiveness of word sense representations. The basic lattice (SAT) gives better performance than the original basic lattice model thanks to considering sense information into word embeddings. Although the basic lattice (SAT) model reaches better overall results, the precision of the



(a) Results on FinRE (b) Results on SanWen (c) Results on ACE-2005

Figure 4: Precision-recall curves of BLSTM, Att-BLSTM, CNN, PCNN, PCNN+ATT, Basic lattice and MG lattice on all datasets. All models (except the Basic and MG lattice) are character-based.

top 100 instances is still lower than the lattice-basic model. Compared with the other two models, MG lattice shows superiority in all indexes of P@N, achieving the best results in the mean scores.

To compare and analyze the effectiveness of all lattice-based models more intuitively, we report the precision-recall curve of the ACE-2005 dataset in Figure 3 as an example. Although the basic lattice (SAT) model obtains better overall performance than the original basic lattice model, the precision is still lower when the recall is low, which corresponds to the results in Table 3. This situation indicates that considering multiple senses only in the pre-trained stage would add noise to the word representations. In other words, the word representation tends to favor the commonly used senses in the corpora, which will disturb the model when the correct sense of the current word is not the common one. Nevertheless, the MG lattice model successfully avoids this problem, giving the best performance in all parts of the curve. This result indicates that the MG lattice model is not significantly impacted by the noisy information because it can dynamically select the sense paths in different contexts. Although MG lattice model shows effectiveness and robustness on the overall results, it is worth noting that the improvement is limited. The situation indicates that the utilization of multi-grained information could still be improved. A more detailed discussion is in Section 5.

4.4 Final Results

In this section, we compare the performance of the lattice-based model with various proposed methods. The proposed models we selected are as follows:

Models	FinRE		SanWen		ACE-2005	
	AUC	F1	AUC	F1	AUC	F1
BLSTM	28.80	42.87	50.21	61.04	60.40	70.03
Att-BLSTM	27.81	41.48	50.42	59.48	61.85	70.69
CNN	27.12	41.47	47.81	59.42	64.49	72.41
PCNN	30.49	45.51	48.26	61.00	66.10	74.33
PCNN+Att	31.89	46.13	50.41	60.55	65.79	73.17
Basic Lattice	36.58	47.41	56.88	63.88	70.51	77.12
MG Lattice	**38.74**	**49.26**	**57.33**	**65.61**	**72.28**	**78.17**

Table 4: AUC and F1-scores of BLSTM, Att-BLSTM, CNN, PCNN, PCNN +Att, Basic lattice and MG lattice on all datasets. All models (except the Basic and MG lattice) are character-based.

CNN (Zeng et al., 2014) proposes a CNN model for relation extraction.

PCNN (Zeng et al., 2015) puts forward a piecewise CNN model with multi-instance learning.

BLSTM (Zhang and Wang, 2015) proposes a bidirectional LSTM model for relation extraction.

Att-BLSTM (Zhou et al., 2016) is a bidirectional LSTM model with word-level attention mechanism. [3]

PCNN+ATT (Lin et al., 2016) improves PCNN model with selective attention mechanism.

We conduct experiments on both character-based and word-based versions of the five models mentioned above. The results show that the character-based versions perform better than the word-based versions for all models on all datasets. Consequently, we only use the character-based version of the five selected models in the following experiments.

[3]For the sake of fairness, we add position embeddings to both BLSTM and Att-BLSTM, which are not used in the original papers.

For comprehensive comparison and analysis, we report precision-recall curves in Figure 4 and F1-scores and AUC in Table 4. From the results, we can observe that: (1) Lattice-based models significantly outperform other proposed models on datasets from different domains. Thanks to the polysemy information, the MG lattice model performs best among all models, showing superiority and effectiveness on the Chinese RE task. The results indicate that sense-level information could enhance the ability to capturing deep semantic information from text. (2) The gap between the basic lattice model and the MG lattice model becomes narrow on the dataset FinRE. The reason for this phenomenon is that FinRE is constructed from financial report corpus, and the words of financial reports are often rigorous and unambiguous. (3) In comparison, the PCNN and PCNN+ATT models perform worse in the SanWen and ACE datasets. The reason is that there are positional overlaps between entity pairs in these two datasets, making PCNN unable to take full advantage of the piece-wise mechanism. The results indicate that the PCNN-based methods have a high dependence on the form of the dataset. In comparison, our models show robustness on all three datasets.

5 Conclusion and Future Work

In this paper, we propose the MG lattice model for Chinese relation extraction. The model incorporates word-level information into character sequences to explore deep semantic features and avoids the issue of polysemy ambiguity by introducing external linguistic knowledge, which is regarded as sense-level information. We comprehensively evaluate our model on various datasets. The results show that our model significantly outperforms other proposed methods, reaching the state-of-the-art results on all datasets.

In the future, we will attempt to improve the ability of the MG Lattice to utilize multi-grained information. Although we have used word, sense and character information in our work, more level of information can be incorporated into the MG Lattice. From coarse to fine, sememe-level information can be intuitively valuable. Here, sememe is the minimum semantic unit of word sense, whose information may potentially assist the model to explore deeper semantic features. From fine to coarse, sentences and paragraphs should be taken into account so that a border range of contextual information can be captured.

6 Acknowledgement

This research is supported by the National Natural Science Foundation of China (Grant No. 61773229), the Basic Scientific Research Program of Shenzhen City (Grant No. JCYJ20160331184440545), and the Overseas Cooperation Research Fund of Graduate School at Shenzhen, Tsinghua University (Grant No. HW2018002). Moreover, Zhiyuan Liu is supported by the National Key Research and Development Program of China (No. 2018YFB1004503). Finally, we would like to thank the anonymous reviewers for their helpful feedback and suggestions.

References

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing 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.

Yanping Chen, Qinghua Zheng, and Wei Zhang. 2014. Omni-word feature and soft constraint for chinese relation extraction. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 572–581.

Yu-Ju Chen and Jane Yung-jen Hsu. 2016. Chinese relation extraction by multiple instance learning. In *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*.

Zhendong Dong and Qiang Dong. 2003. Hownet-a hybrid language and knowledge resource. In *Proceedings of NLP-KE*.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Xiaotian Jiang, Quan Wang, Peng Li, and Bin Wang. 2016. Relation extraction with multi-instance multi-label convolutional neural networks. In *Proceedings*

of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1471–1480.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Joohong Lee, Sangwoo Seo, and Yong Suk Choi. 2019. Semantic relation classification via bidirectional lstm networks with entity-aware attention using latent entity typing.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2124–2133.

ChunYang Liu, WenBo Sun, WenHan Chao, and Wanxiang Che. 2013. Convolution neural network for relation extraction. In *International Conference on Advanced Data Mining and Applications*, pages 231–242. Springer.

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.

Yilin Niu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Improved word representation learning with sememes. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2049–2058.

Nanyun Peng and Mark Dredze. 2016. Improving named entity recognition for chinese social media with word segmentation representation learning. *arXiv preprint arXiv:1603.00786*.

Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Qiang Dong, Maosong Sun, and Zhendong Dong. 2019. Openhownet: An open sememe-based lexical knowledge base. *arXiv preprint arXiv:1901.09957*.

Samuel Rönnqvist, Niko Schenk, and Christian Chiarcos. 2017. A recurrent neural model with attention for the recognition of chinese implicit discourse relations. *arXiv preprint arXiv:1704.08092*.

Matthias Sperber, Graham Neubig, Jan Niehues, and Alex Waibel. 2017. Neural lattice-to-sequence models for uncertain inputs. *arXiv preprint arXiv:1704.00559*.

Jinsong Su, Zhixing Tan, Deyi Xiong, Rongrong Ji, Xiaodong Shi, and Yang Liu. 2017. Lattice-based recurrent neural network encoders for neural machine translation. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Lin Sun, Kui Jia, Kevin Chen, Dit-Yan Yeung, Bertram E Shi, and Silvio Savarese. 2017. Lattice long short-term memory for human action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2147–2156.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Jingjing Xu, Ji Wen, Xu Sun, and Qi Su. 2017. A discourse-level named entity recognition and relation extraction dataset for chinese literature text. *arXiv preprint arXiv:1711.07010*.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. *arXiv preprint arXiv:1704.08960*.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.

Qian-qian ZHANG, Meng-dong CHEN, and Lianzhong LIU. 2017. An effective gated recurrent unit network model for chinese relation extraction. *DEStech Transactions on Computer Science and Engineering*, (wcne).

Yue Zhang and Jie Yang. 2018. Chinese ner using lattice lstm. *arXiv preprint arXiv:1805.02023*.

Hai Zhao and Chunyu Kit. 2008. Unsupervised segmentation helps supervised learning of character tagging for word segmentation and named entity recognition. In *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing*.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 207–212.