



An input information enhanced model for relation extraction

Ming Lei¹ · Heyan Huang¹ · Chong Feng¹ · Yang Gao¹ · Chao Su¹

Received: 7 March 2019 / Accepted: 7 August 2019
© Springer-Verlag London Ltd., part of Springer Nature 2019

Abstract

We present a novel end-to-end model to jointly extract semantic relations and argument entities from sentence texts. This model does not require any handcrafted feature set or auxiliary toolkit, and hence it could be easily extended to a wide range of sequence tagging tasks. A new method of using the word morphology feature for relation extraction is studied in this paper. We combine the word morphology feature and the semantic feature to enrich the representing capacity of input vectors. Then, an input information enhanced unit is developed for the bidirectional long short-term memory network (Bi-LSTM) to overcome the information loss caused by the gate operations and the concatenation operations in the LSTM memory unit. A new tagging scheme using uncertain labels and a corresponding objective function are exploited to reduce the interference information from non-entity words. Experiments are performed on three datasets: The New York Times (NYT) and ACE2005 datasets for relation extraction and the SemEval 2010 task 8 dataset for relation classification. The results demonstrate that our model achieves a significant improvement over the state-of-the-art model for relation extraction on the NYT dataset and achieves a competitive performance on the ACE2005 dataset.

Keywords Relation extraction · Information extraction · Natural language processing · Deep learning

1 Introduction

Relation extraction is an important research topic in information extraction. It is very helpful for information retrieval, question answering, machine translation and other natural language processing (NLP) tasks. Besides, it is also an essential step for constructing knowledge bases automatically. Relation extraction aims to extract structured information from unstructured or semi-structured documents, namely, to assign appropriate predefined relation types to the entity pairs extracted from given texts. Generally, a triple is used as the format of the structured representation. For example, the sentence “On November 15th, 2017, US president Donald Trump arrived in Australia for a state visit” contains a triple as follows: (entity 1:

Donald Trump, relation type: Entity-Destination, entity 2: Australia), where “Entity-Destination” is a predefined target relation type, “Donald Trump” is the first entity, and “Australia” is the second entity. It could be seen that the relation type “Entity-Destination” is mainly dependent on the words or phrases “**arrived in**,” which we call keywords for the relation type. Furthermore, these sentences, such as “The visitors **arrive in** London on time,” “Tim **arrives in** London and meets his new friends,” “After **arriving in** Paris, Bob feels very excited” and “Donald Trump’s **arrival in** Australia for a state visit happened in November,” contain the keywords as “**arrive in**, **arrives in**, **arriving in**, **arrival in**,” respectively, which express the same relation type. It seems to indicate that the keywords with the similar morphology express the same relation type. Moreover, the entity words with the similar morphology are the same entity type. For example, “**worker**,” “**farmer**,” “**teacher**,” “**driver**” are person entities; “**Georgia**,” “**California**,” “**Virginia**” are location entities; “**Hopeful**,” “**thankful**,” “**careful**” are non-entity words. We could see that both entity recognition and relation extraction are word morphology-related tasks, so word morphology information could be a useful feature for relation extraction.

✉ Chong Feng
fengchong@bit.edu.cn

Ming Lei
3120160450@bit.edu.cn

Heyan Huang
hhy63@bit.edu.cn

¹ Beijing Institute of Technology, Beijing, China

It goes without saying that besides the morphological information, the semantic information of words is also a useful feature for relation extraction. Therefore, we integrate word morphology encoding vectors with word embedding vectors to capture the morphological and semantic features. We introduce the redundancy encoding technique to encode character-level word morphology feature. These researches [1–4] have used the morphological vectors successfully in text classification and named entity recognition. This paper is to apply word morphology vectors for relation extraction. Different from [1], we adopt the redundancy encoding instead of the one-hot encoding technique. Unlike the researches [2–4], word morphology encoding rather than word morphology embedding is utilized in this paper. Compared with the one-hot encoding for characters, the redundancy encoding could overcome the problem of sparsity and increase the code distance (see Sect. 3.1.1) to raise robustness. Word morphology embedding is a technique inspired by word embedding. It maps all words to a continuous real-value vector space, while the redundancy encoding employs discrete digit vectors to represent words. Although the two methods are based on the same point: the more similar the morphology of the words, the less the distance between the word vectors, the redundancy encoding could increase the code distance between different characters, thus improving the system robustness.

In addition, relation extraction can be converted into a sequential token tagging problem by a tagging scheme [5, 6]. As an effective way for sequence tagging tasks, LSTMs [7] have been successfully applied to the end-to-end models for relation extraction. LSTMs are capable of learning long-distance correlations in a word sequence by utilizing a memory unit, which consists of three gates: an input gate, a forget gate and an output gate. The input gate controls the extent to which the input information flows into the cell. The forget gate controls the extent to which the history state remains in the cell. The output gate controls the extent to which the current state in the cell is transmitted to the output of the LSTM unit. That is to say, the input gate allows part of information to be inputted; the output gate allows part of information to be outputted; and the forget gate allows part of information to be memorized. Therefore, the gate operations could cause information loss. Besides, the concatenation operations of the input vectors and the state vectors could cause the input information dilution, just as pouring water into milk can dilute the milk. As shown in Fig. 1, when the input information from the starting point arrives at the output of the memory unit, it needs to go through one concatenation and three gate operations.

Therefore, although LSTMs are capable of memorizing the useful history information by using the memory units,

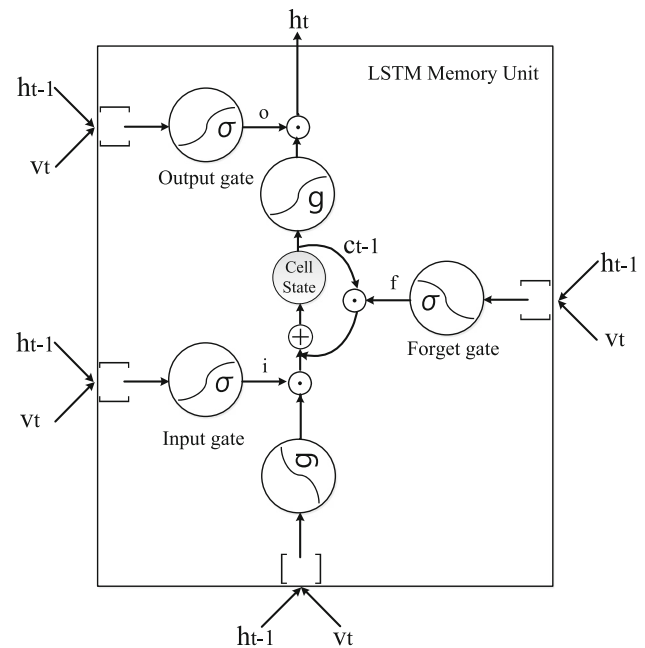


Fig. 1 The architecture of a standard LSTM memory unit. The signs \odot , \oplus , $[]$ denote point-wise product (Hadamard product), point-wise sum and concatenation operation, respectively. i , f , o are the outputs of the input gate, the forget gate and the output gate, respectively. v is the input; h is the output; c is the cell state of the memory unit, and t represents at t time step. σ is the sigmoid function, and g is the activation function

the concatenation and gate operations would lead to the loss of the current input information. Aiming to reduce the information loss caused by the input passing through the gates and the information dilution caused by the concatenation operation, this paper tries to study a new BI-LSTM [8] with input information enhanced (Bi-LSTM-IE) model to strengthen the input information.

The existing researches transforming relation extraction into a tagging task have become more and more popular in recent years; however, all of them treat non-entity words as one type and tag them with a single label. For instance, in the above-mentioned example sentence, the non-entity words “On,” “November,” “for,” “a,” “arrived” and “in” are tagged with a label “N”. In fact, these words include both the keywords and the non-keywords for the relation type. Obviously, it may be unreasonable to tag them with one label. In order to solve the problem of tagging non-entity words with one label, we devise a novel tagging scheme, in which the label of a word is made up of three parts. For a non-entity word, the entity part is denoted by the sign “N”, and the relation part and the number part are denoted by the sign “X”. The cost of “X” is not calculated in the objective function, so its value is uncertain. We use the uncertain labels “N–X” to tag the non-entity words and reduce the interference information from them. The

experiment results demonstrate the improvement is effective.

The contributions of this paper can be summarized as follows:

- We present a new method of encoding the word morphology feature and introduce the redundancy encoding technique to improve robustness.
- We design an input information enhanced unit for LSTMs to tackle the problem of the input loss and dilution.
- We first exploit a novel tagging approach using uncertain labels to reduce the interference information from non-entity words.
- We propose a Bi-LSTM-IE model for relation extraction, which outperforms the state-of-the-art model on the NYT dataset.

This paper is organized as follows: we discuss the related work in Sect. 2, and then we describe the architecture of our model in Sect. 3. We present the input layer with the morphologic and semantic information in Sect. 3.1, illustrate the input information enhanced unit in Sect. 3.2 and introduce the decoding layer with the anti-interference objective function in Sect. 3.3. Finally, we perform experiments and analyze the results in Sect. 4.

2 Related work

Current systems for relation extraction could be categorized into two classes: the semi-supervised models based on handcrafted matching rules [9–17] and the supervised models based on hand-labeled training data [5, 6, 18–22]. Among the semi-supervised methods, some studies [9, 13] apply a set of matching rules to extract relations, some [10, 11, 15, 17] rely on a bootstrapping technique for extracting relations, in which a five tuple pattern is iteratively used to match the candidate relations, and others [12, 14] build a series of propagated rules on a graph to tag the unlabeled relations. There is no need of massive annotated data for the semi-supervised models. Nevertheless, it is difficult to design proper rules or patterns not only with high precision but also with high coverage.

The supervised models are mainly divided into: the methods [18–20] based on kernel functions which manually select feature sets to classify the relations and the methods [5, 6, 21] based on neural networks (NNs) which automatically learn latent features to extract relations. The available feature sets generally include lexical, semantic and syntactic features, such as word embeddings, part-of-speech tags, dependency types, entity labels, position information, hyponymy relations and so on. In recent years, the word morphology feature has aroused attention in these

studies [1–4, 23, 24]. The previous work [1] exploited the one-hot encoding and a convolutional neural network (CNN) [25] model with the word morphology feature only to classify texts. The work encoded 70 characters into 70-dimensional vectors, in which the code distance was 1. Thus, it suffered from the problem of sparsity. By contrast, our work encodes 56 characters into 9-dimensional vectors, in which the code distance is 6. We obtain larger code distance with fewer dimensions. So, our method boosts robustness while saving computation. The researches [2, 4, 23, 24] used CNNs to encode morphological information of a word into its character-level representation, and the research [3] employed a Bi-LSTM model for training word vectors which contained both semantic and morphological information. These researches [2–4, 23] mapped all words to continuous real-value vectors and applied them to NLP tasks successfully. The redundancy encoding employs discrete digit vectors to represent words, which can increase the code distance between different characters, thus improving the system robustness. So, in this article, the morphology information is represented by the discrete value vectors.

The NN-based models, which can take advantage of the handcrafted features or not, include the RNN-based [5, 26–28], the CNN-based [21, 29–31] and the combined [32]. However, the above-mentioned researches adopted the pipelined manner. For example, the work [21] exploited a CNN model along the shortest dependency paths with WordNet to classify relations and the work [5] recognized entity pairs with POS tags firstly then used a dependency tree-LSTM to classify relations. The pipelined manner makes relation extraction easy to deal with, but it neglects the relevance between entities and relations. Moreover, it could lead to error propagation from the upstream to the downstream. To solve these problems, the models [6, 33–36] of jointly extracting relations and entities are proposed. In this paper, we present a Bi-LSTM-based joint model for relation extraction.

In theory, the NN-based methods are capable of learning the latent features and patterns required, but the performance of these systems mainly depends on the quantity and quality of training data. To reduce human involvements, the researches [37–40] studied a kind of distant supervised methods which yielded the annotated data by using knowledge bases instead of human labors. We conduct the experiments for relation extraction on the NYT dataset which was constructed by the work [40] in the distant supervised manner.

3 Bi-LSTM-IE model

LSTMs can learn long-term dependencies in the sequences. Bi-LSTMs are capable of capturing the full context information from the past (history) and future. On the basis of Bi-LSTMs, we present a new end-to-end model via tagging the word sequence from a sentence to extract relations. The overview of the model is shown in Fig. 2. It consists of 3 layers; those are the input layer with the morphological and semantic feature information, the encoding layer with the input enhanced unit and the decoding layer with the denoising objective function. The role of the input layer is to provide useful feature information as much as possible for the encoding layer. In the input layer, we exploit the redundancy encoding technique to improve robustness and combine morphological and semantic information to extract relations. The word morphology encoding vector v_c is used to capture the morphological feature, and the word embedding vector v_b is used to capture the semantic feature. The input word vector at t time step v_t is produced by concatenating v_c and v_b . In the encoding layer, we present an input information enhanced unit to overcome the information loss and then integrate it into a standard Bi-LSTM to generate the encoding vectors. The dashed line part is the information enhanced unit, which is a fully connected layer (FC) activated by the tanh function. In the decoding layer, we utilize the uncertain labels to reduce the interference information caused by non-entity words. A single-layer Bi-LSTM and three hierarchical fully connected layers (FCs) are used to separate relation and entity information from the encoding vectors. The bottom of the

decoding layer is a plain single-layer Bi-LSTM, and the top is a group of three hierarchical FCs e , r and n normalized by the *softmax* function. The layers e , r and n output likelihood probabilities of the entity type, the relation type and the entity number, respectively. The output of the entity type layer e is a part of the input of the relation type layer r and the entity number layer n . The output of the relation type layer r is a part of the input of the entity number layer n (see formulas 16–18 for the specific operations). This architecture can well model the causal link between the entity type, the relation type and the entity number. “E”, “1”, “ED”, “N”, “X” are the output labels of the words in the example sentence. A detailed description as follows:

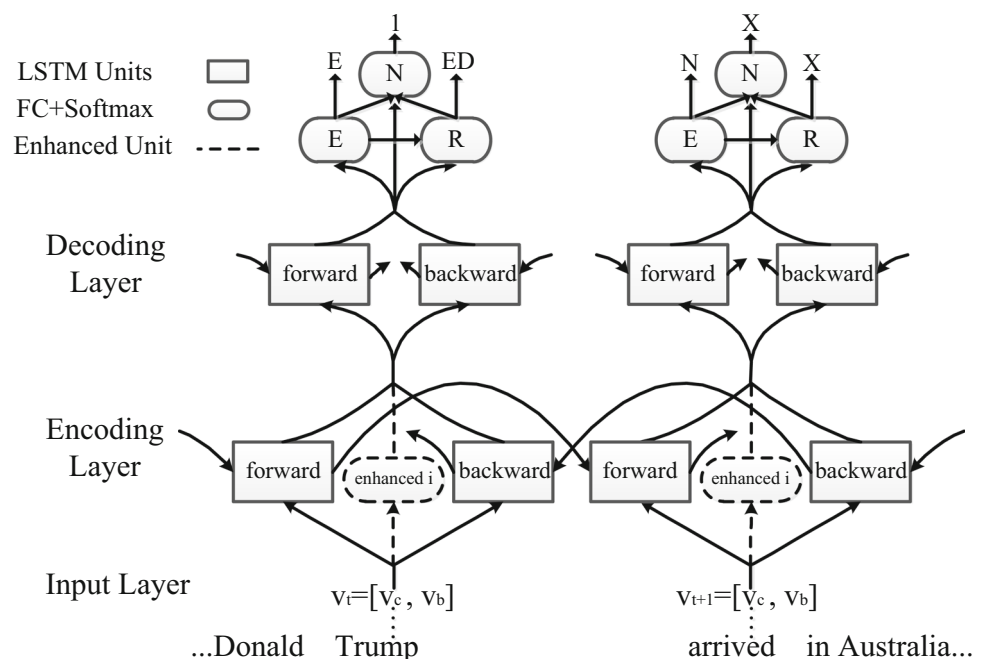
3.1 Input layer with the morphologic and semantic feature information

By virtue of carrying abundant semantic information, word embeddings are very helpful for NLP tasks. But they have the shortcoming of lacking in the word morphology feature information. Considering relation extraction is a not only semantics related but also word morphology-related task, we combine word encodings with word embeddings to capture the morphological and semantic information in the input layer.

3.1.1 Word encoding

Before encoding words, we replace all number words with a special word “num”. We encode each character into a

Fig. 2 The overview of the end-to-end Bi-LSTM-IE model. It consists of the input layer, the encoding layer and the decoding layer



unique vector firstly, then pad each word with space characters or truncate it to preset length WL , lastly according to the characters in the word concatenate the character vectors as the word encoding vector v_c . We produce the character vectors by the redundancy encoding technique, which is used in computation network communication to improve channel reliability. Next, we will review the redundancy encoding and elaborate how to encode characters.

In information theory, code distance D in an encoding system is defined as the minimum Hamming distance between any two codes. If $p = (p_1, p_2, \dots, p_n)$ and $q = (q_1, q_2, \dots, q_n)$ are two codes in an encoding system C , the Hamming distance d_{pq} between p and q is computed by the following formula:

$$d_{pq} = \sum_{i=1}^n |p_i - q_i|. \quad (1)$$

And the code distance $D(C)$ in encoding system C is given by:



$$D(C) = \min_{p \neq q} d_{pq} \quad (p, q \in C). \quad (2)$$

In computer network communication, the ability of detecting and correcting errors is determined by the code distance of the encoding system. As shown in Fig. 3, the ability of detecting and correcting errors of the binary encoding, the one-hot encoding and the redundancy encoding is compared. In this example, with the code distance of 1, the binary encoding does not have the ability of detecting and correcting errors. With the code distance of 2, the one-hot encoding has the ability of detecting 1 error. With the code distance of 5, the redundancy encoding has the ability of detecting at most 4 errors and correcting at most 2 errors. Thus, an encoding system with code distance D can detect at most $D - 1$ errors and correct at most $\lfloor (D - 1)/2 \rfloor$ errors. That is to say, the larger the

code distance of an encoding system, the stronger its ability of detecting and correcting errors. For neural networks, the ability of detecting and correcting errors can increase the robustness of the model. Based on this theory, to enlarge the code distance, we adopt the balanced ternary redundancy encoding technique in our model. The balanced ternary is a base 3 number system in which the digits have the values -1 , 0 and 1 . It can increase the code distance, and well match the tanh activation function, the value range of which is $(-1, 1)$. The specific procedure encoding the characters is as follows:

1. First of all, we use a $CS - 1$ (CS is the character vector size)-dimensional all-zero vector to encode the space character denoted as "sc".
2. With the space character vector as the base point, we employ exhaustive algorithm to search all other legal codes between which the Hamming distance is larger than or equal to the code distance D .
3. The legal codes are randomly allocated to the letters "a-z", apostrophe hyphen, and other characters (excluding "A-Z" and "0-9") denoted as "oc" 29 characters in total.
4. We add one dimension 1 to the front of the vectors of "a-z" as the vectors of "A-Z", and add one dimension -1 to the front of the vectors of "a-z" as the final vectors of "a-z".
5. We add one dimension 0 to the front of the vectors of non-letter characters as the final vectors of the non-letter characters, including "sc", apostrophe, hyphen, and "oc".
6. We obtain the CS -dimensional vector representations of all the 56 characters.

From the generation process, we can see that the encoding system has the following characteristics:

S: Sender
R: Receiver
TP: True Positive Example
FP: False Positive Example
FN: False Negative Example
Normal Channel: 
Noisy Channel: 
Cause \rightarrow Effect
P(X): The Probability of X





Binary Encoding A : 1 B : 0	S: 1  1 S: 1  0	R: A \rightarrow TP(A)+1 R: B \rightarrow FP(B)+1, FN(A)+1
One-hot Encoding A : 0 1 B : 1 0	S: 0 1  1 1	R: Error \rightarrow Request retransmission
Redundancy Encoding A : 1 1 1 1 1 B : 0 0 0 0 0	S: 1 1 1 1 1  1 1 0 0 1	R: Error, P(A) > P(B) \rightarrow Correct error to 11111, TP(A)+1

Fig. 3 A comparison of the ability of detecting and correcting errors between the three encoding methods. We regard the computer communication system as a classification task model. If the sender sends "A", the receiver may receive not "A" under the condition of the noisy channel: (1) With the code distance of 1, the binary encoding does not have the ability of detecting and correcting errors, so a false negative example of "A" and a false positive example of

"B" will be produced, namely $FN(A) + 1$ and $FP(B) + 1$. (2) With the code distance of 2, the one-hot encoding has the ability of detecting 1 error, so after detecting the error, the receiver will request retransmission. (3) With the code distance of 5, the redundancy encoding has the ability of detecting at most 4 errors and correcting at most 2 errors, so the receiver will correct errors according to the probability and the true positive example still can be produced

- It is a $CS = 9$ -dimensional encoding system, the code distance of which is $D = 6$. We obtain larger code distance with fewer dimensions. So, our method boosts robustness while saving computation.
- The first dimension value of all the upper case letters is 1, the lower case letters is -1 and the non-letter characters is 0. The advantage is that the three categories of characters can be well distinguished.
- Except for the first dimension, the vector of a lower case letter and the vector of its upper case letter are the same. The motivation is to treat them as a letter as well as differentiate them.
- After treating the number words as the word “num,” the encoding system can encode all words into vectors, and even has the ability of detecting at most 5 errors and correcting at most 2 errors.

3.1.2 Word embedding

The semantic information which is abundant in the word embedding vectors is an important feature for relation extraction. Therefore, the word embedding vectors v_b are used to capture the semantic information of the input words. They are generated by running the word2vec3 CBOW model [41] with the following settings on the training corpus: the dimension size of word vectors is $d(v_b) = 300 - d(v_c)$, the window size is 5, the threshold for occurrence of words is $1e - 3$, the number of negative examples is 6, the starting learning rate is 0.05, and the number of iterations is 200. This process is completed in an unsupervised manner, so any annotated data are not required.

3.1.3 Word concatenating

We concatenate the word encoding vectors v_c with the word embedding vectors v_b as the input word vectors $v = [v_c, v_b]$, in which v_c are used to capture the morphological feature and v_b are used to capture the semantic feature.

3.2 Encoding layer with the input information enhanced unit

We employ an improved single-layer Bi-LSTM as the encoding layer. In the standard LSTM model, the input information is concatenated with the hidden state at last time step h_{t-1} firstly and then passes through the input gate, the forget gate and the output gate successively. We argue that the concatenation operation leads to input information dilution, and the transmission process results in input information loss. To solve these problems, we propose the

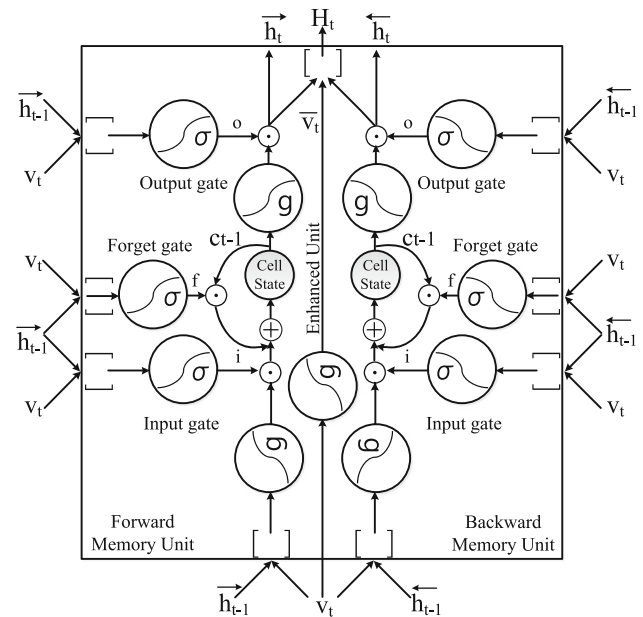


Fig. 4 The architecture of Bi-LSTM-IE unit. The signs $\odot, +, []$ denote point-wise product (Hadamard product), point-wise sum and concatenation operations, respectively. i, f, o are outputs of the input gate, the forget gate and the output gate, respectively. v is the input. \vec{h} is the output of the forward memory unit. \overleftarrow{h} is the output of the backward memory unit. \bar{v}_t is the output of the enhanced unit. H is the output of Bi-LSTM-IE unit. c is the cell state, and t represents at time step. σ is the sigmoid function, and g is the activation function

Bi-LSTM with input information enhanced (Bi-LSTM-IE) model as shown in Fig. 4. In this model, the input vector v without gate and concatenation operations is transformed by a fully connected layer to generate the enhanced input information \bar{v} , and then \bar{v} is concatenated with the forward encoding vector \vec{h} and the backward encoding vector \overleftarrow{h} as the encoding layer's output vector $H = [\vec{h}, \bar{v}, \overleftarrow{h}]$. At the time step t , H_t carries the information of a whole sentence, in which \vec{h}_t carries the past, \overleftarrow{h}_t carries the future and \bar{v}_t carries the current information of the word. Detailed operations in one direction are calculated by the following formulas:

$$i_t = \sigma(W_i[v_t, h_{t-1}] + b_i) \quad (3)$$

$$f_t = \sigma(W_f[v_t, h_{t-1}] + b_f) \quad (4)$$

$$o_t = \sigma(W_o[v_t, h_{t-1}] + b_o) \quad (5)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c[v_t, h_{t-1}] + b_c) \quad (6)$$

$$h_t = o_t \odot \tanh(c_t) \quad (7)$$

where v_t is the input vector, and i_t, f_t, o_t are outputs of the input gate, the forget gate and the output gate, respectively. The operator \odot denotes the point-wise multiplication (Hadamard product), and $[]$ denotes the concatenation

operation. Here, c_t is the state of the LSTM cell, and c_{t-1} is the state at last time step. The encoding vector h_t is the output of the LSTM unit in one direction. The output of the encoding layer H_t , which is also the input of the decoding layer, is given as follows:

$$\bar{v}_t = \tanh(W_h v_t + b_h) \quad (8)$$

$$v_t^{(2)} = H_t = \left[\vec{h}_t, \bar{v}_t, \overleftarrow{h}_t \right] \quad (9)$$

where \bar{v}_t is the enhanced input information, \vec{h}_t is the encoding vector in the forward direction, and \overleftarrow{h}_t is the encoding vector in the backward direction.

3.3 Decoding layer with the anti-Interference objective function

A standard one-layered Bi-LSTM and three hierarchical fully connected layers (FCs) are used as the encoding layer, in which we employ a novel tagging scheme with a matching objective function to reduce interfere caused by non-entity words.

3.3.1 Tagging scheme

As shown in Fig. 5, each word is tagged by a label, which consists of three parts: the entity part, the number part and the relation part. In the entity part, “E” represents an entity, while “N” represents a non-entity. In the number part, “1” and “2” denote the roles of the entity in a relation triple. “E1” indicates that the word belongs to entity 1, and “E2” indicates that the word belongs to entity 2. “ED,” “CE” and others represent the relation types. “E0-R0” represents an entity which belongs to the “None” relation type. The non-entities are tagged by the uncertain labels “N-X.” The cost of “X” is not calculated in the objective function, so its value is uncertain and it can represent anything. It means that if a word is a non-entity, we are unnecessary to concern which relation type it belongs to. It is unreasonable to tag all the non-entity words, for example “arrived,” “caused,” “for,” “a,” “which,” “the” with a single label. The advantage of introducing the uncertain labels is that this approach can reduce the interference caused by the non-entity words. As far as we know, it is the

first time that the uncertain labels are applied in neural network models.

3.3.2 Decoding method

We consider the decoding vector $H_t^{(2)}$ contains both entity and relation information. We adopt the hierarchical decoding approach to extract the information separately. As illustrated in Fig. 2, The bottom of the decoding layer is a standard single-layered Bi-LSTM, and the top is a group of three hierarchical fully connected layers (FC) e , r and n normalized by the *softmax* function. The FC e , r and n output likelihood probabilities of the entity type, the relation type and the entity number, respectively. The output of the entity type layer e is used as a part of the input of the relation type layer r and the entity number layer n , and the output of the relation type layer r is used as a part of the input of the entity number layer n (see formulas 16–18 for the specific operations). This architecture well describe the causal link between the entity type, the relation type and the entity number. Concrete operations are defined by the following formulas:

$$i_t^{(2)} = \sigma(W_i^{(2)}[v_t^{(2)}, h_{t-1}^{(2)}] + b_i^{(2)}) \quad (10)$$

$$f_t^{(2)} = \sigma(W_f^{(2)}[v_t^{(2)}, h_{t-1}^{(2)}] + b_f^{(2)}) \quad (11)$$

$$o_t^{(2)} = \sigma(W_o^{(2)}[v_t^{(2)}, h_{t-1}^{(2)}] + b_o^{(2)}) \quad (12)$$

$$c_t^{(2)} = f_t^{(2)} \odot c_{t-1}^{(2)} + i_t^{(2)} \odot \tanh(W_c^{(2)}[v_t^{(2)}, h_{t-1}^{(2)}] + b_c^{(2)}) \quad (13)$$

$$h_t^{(2)} = o_t^{(2)} \odot \tanh(c_t^{(2)}) \quad (14)$$

$$H_t^{(2)} = \left[\vec{h}_t^{(2)}, \overleftarrow{h}_t^{(2)} \right] \quad (15)$$

$$\tilde{y}_t^e = \text{softmax}(\tanh(W_e H_t^{(2)} + b_e)) \quad (16)$$

$$\tilde{y}_t^r = \text{softmax}(\tanh(W_r[H_t^{(2)}, \tilde{y}_t^e] + b_r)) \quad (17)$$

$$\tilde{y}_t^n = \text{softmax}(\tanh(W_n[H_t^{(2)}, \tilde{y}_t^e, \tilde{y}_t^r] + b_n)) \quad (18)$$

where the superscript (2) refers to the decoding layer, and \tilde{y}_t^e , \tilde{y}_t^r , \tilde{y}_t^n are the prediction vectors of the entity type, relation type and entity number, respectively. We use the

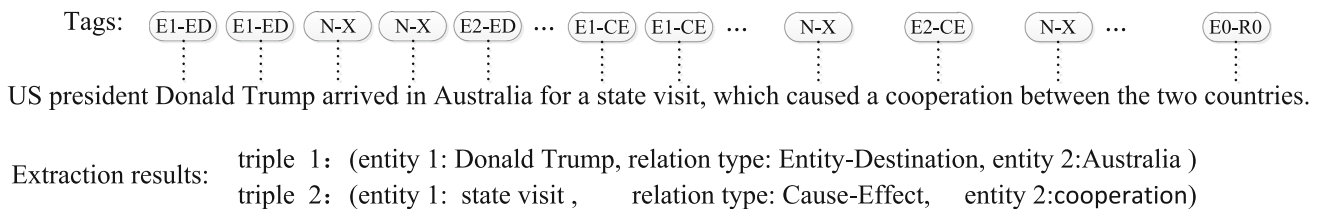


Fig. 5 The tagging and extracting results of an example sentence. “ED” is short for “Entity-Destination,” and “CE” is short for “Cause-Effect”

cross entropy loss \mathcal{L} to minimize the distance between the predicted values and the real values.

$$\mathcal{L}_t^e = - \sum_{j=1}^J y_{tj}^e \log \tilde{y}_{tj}^e \quad (19)$$

$$\mathcal{L}_t^r = - \sum_{k=1}^K y_{tk}^r \log \tilde{y}_{tk}^r \quad (20)$$

$$\mathcal{L}_t^n = - \sum_{l=1}^L y_{tl}^n \log \tilde{y}_{tl}^n \quad (21)$$

where \mathcal{L}_t^e , \mathcal{L}_t^r , \mathcal{L}_t^n are the losses of entity type, relation type and entity number, respectively. y_t^e , y_t^r , y_t^n are the real values of entity type, relation type and entity number, respectively. J , K , L are the dimension size of vectors y_t^e , y_t^r and y_t^n , respectively. We train the model by minimizing the objective function \mathcal{F} . The objective function is given by:

$$\mathcal{F} = \sum_{s=1}^S \sum_{t=1}^{T_s} \mathcal{L}_t^e + \xi(\alpha \mathcal{L}_t^r + \beta \mathcal{L}_t^n) \quad (22)$$

$$\xi = \sum (y_t^e u^e, \text{axis} = \text{entity_axis}) = \begin{cases} 1, & \text{if the word belongs to an entity} \\ 0, & \text{otherwise} \end{cases} \quad (23)$$

where S is the sentence number in training set and T_s is the length of the sentence s . Bias weights α and β determine the relative important degree of the two kinds of loss. ξ is a flag variable. When the current word is an entity, it equals 1 or it equals 0. u^e is the entity tag vector. The second parameter of *sum* function *axis* = *entity_axis* means calculating the sum in the entity dimension. For example, we define the entity tag vector $u^e = (0, 1)$, when the word is not an entity, $y_t^e = (1, 0)$, $\xi = 0 \times 1 + 1 \times 0 = 0$. It implies that the relation part and the number part of a non-entity word are not computed in the objective function. That is to say ξ realizes the function of the uncertain labels.

4 Experiments and analyses

4.1 Experiment settings

In order to test the effectiveness of the model in this paper, we design a series of experiments, including the comparison experiments of relation extraction on the NYT and ACE2005 datasets, a comparison experiment of relation classification on the SemEval-2010 dataset, and an ablation experiment in our model. Besides, we tune the hyperparameter WL to find the optimal number of characters representing the words and then compare our word

morphology encoding method with the one-hot encoding [1] and the word morphology embedding [2–4].

4.1.1 Datasets

We carry out experiments on three datasets. NYT¹ and ACE2005 are used to evaluate performance for relation extraction, and SemEval-2010 task 8 is used to measure performance for relation classification.

The NYT dataset is constructed by the work [40] in a distant supervised manner. The training set has 235,982 sentences and 25 relation types including “None” relation type in total. It contains 801,005 entities and 372,853 triples, 111,610 of which are practical relation triples excluding the “None” type. The test set has 395 sentences denoted by human labors, in which there are 3880 triples, 410 of which is practical relation type excluding the “None” type. It contains 1361 entities belonging to 3 entity types: “Person,” “Location” and “Organization.”

The ACE2005 dataset contains weblogs, broadcast news and newswire corpora in three languages. We choose the English part as a dataset. It defines 6 relation types: “PART-WHOLE,” “ORG-AFF,” “PER-SOC,” “ART,” “PHYS” and “GEN-AFF.” It contains 12,653 sentences and 7112 relation triples. There are 1808 sentences containing more than 2 relation triples. The most number of triples in a sentence is 11. So, ACE2005 is a multiple relation dataset.

SemEval-2010 task 8 is a dataset usually used to evaluate relation classification models. It defines nine practical relations and an “Other” relation type. The relation type “Other” indicates that the relation expressed in the sentence is not among the nine types. The entity types are made up of “entity 1” and “entity 2.” The training set contains 8000 and the test set contains 2717 sentences. Each sentence includes a pair of entities and a relation between them.

4.1.2 Baselines

We compare our model with the researches [6, 40] and some traditional baselines on NYT for the single relation extraction task and compare our model with the feature matrix joint model [36], the shortest dependency path tree (SPTree) [5] and the graph LSTM model [42] on ACE2005 for the multiple relation extraction task. CoType [40] employed a text segmentation algorithm to recognize entities and then embedded entities, relations, text features and type labels into two low-dimensional spaces to predict the unknown entities and relations. A feature set consists of

¹ The NYT dataset can be downloaded at: <https://github.com/shanzhenren/CoType>.

10 kinds of text features were used in this model, including the part-of-speech, entity head, entity order, entity distance and entity context. The work [6] exploited an end-to-end LSTM based model with a new tagging scheme to extract relation triples. Like our approach, this model did not build handcrafted feature sets, but input word vectors only to automatically learn latent features. Different from our system, this model labeled all the non-entity words with a tag “O” and entities with “B,” “I,” “E” and “S” (Begin, Inside, End, Single) tags to stand for position information of a word in entities. FCM [43] is a method using context representation to extract relations. DS-logistic [37] is a multi-class logistic classifier based on lexical features, syntactic features and the entity tag feature. Line [44] is a method which can embed undirected, directed or weighted graphs into a low-dimensional vector space to handle the link prediction problem. MultiR [39] is a probabilistic graphical model of multi-instance learning which extracts overlapping relations. DS-Joint [36] is an incremental joint framework to extract entities and relations using structured perceptron with beam-search. The results of these baselines are listed in the work [6, 40].

In relation classification task, we choose the researches [5, 29, 45] as baselines in which few auxiliary toolkits and feature sets were utilized. Moreover, we list the results of some traditional models described in [29]. SPTree [5] is a pipelined model based on LSTMs, which applied the part-of-speech feature for entity recognition and the dependency feature for relation classification. The work [45] assigned a vector and a matrix for each word to capture the meaning of the word and the neighboring words in a syntactic tree path. This Matrix Vector RNN (MVRNN) model did not use any hand-drafted feature but needed the help of a syntactic parser. CDNN [29] exploited a deep CNN to extract lexical and sentence level features for relation classification.

For the word morphology representation method, we compare the redundancy encoding with the one-hot encoding [1], the CNN-based word morphology representation [2, 4, 23] and the word morphology embedding [3] on the SemEval-2010 dataset for relation classification.

4.1.3 Evaluation metrics

For relation extraction on NYT, as the work [6] did, in the training process, we did not use the “Person,” “Location” and “Organization” labels of entities in the NYT dataset. Because when a relation triple is extracted, the entity type has been determined. For instance, in a triple (“A,” “people /person/birthplace,” “B”), we know “A” is a “Person” entity and “B” is a “Location” entity. We label entities with “E0,” “E1,” “E2” and non-entities with “N.” We follow the previous work [5, 6, 36, 40, 42] and adopt

standard Precision (P), Recall (R) and F1 values for evaluating the performance. The ground-truth relations are extracted, and the “None” relation type is excluded. Different from [6, 40], we regard a triple are correct only if both of the two entities and the relation type are correct.

For relation classification, as these studies [5, 21, 29, 45] did, we used the macro-average F1 (F1-macro) scores for evaluating the performance of the system.

4.1.4 Parameter settings

In the input layer, the encoding size of characters is $CS = 9$, and the code distance is $D = 6$. We select the preset length of the word WL among $\{5, 6, 7, 8, 9, 10, 11, 12\}$, and thus the dimension size of word encoding vector is $d(v_c) = CS \times WL$. To compare our model with the baselines fairly, after concatenating v_c and v_b , each word is represented by a 300-dimensional vector. So the input size is $IS = 300$. We set the dimension size of word embedding vectors $d(v_b) = 300 - d(v_c)$. In the encoding layer, the number of LSTM units in one direction and the number of the input enhanced units are all 300. In the decoding layer, the number of LSTM units in one direction is 900. The batch size BS on the NYT dataset is 390 and on the SemEval-2010 dataset is 100. We select learning rate LR among $\{0.005, 0.0005, 0.00005, 0.000005\}$, and bias weight α, β among $\{0.5, 0.75, 1, 1.25\}$. First of all, we fix $LR = 0.000005$ and $WL = 12$, and then divide the training set of SemEval-2010 into 16 groups, each of which contains 500 sentences. Next, we obtain the optimal values of α and β by a 16-fold cross-validation. After that, we fix the bias weights and select the optimal value of LR on the whole training set. Finally, we fix LR and select the optimal value of WL . The optimal configurations are: $\alpha = 1$, $\beta = 0.75$, $LR = 0.00005$ and $WL = 9$. The word embedding vectors are produced by running word2vec3 CBOW Model [41] on the training corpus. We leverage Adam algorithm [46] to compute gradients and BPTT algorithm to update parameters.

4.2 Results and analyses

The results of relation extraction on NYT are presented in Table 1. The models in the first five lines are the traditional baselines listed in [6, 40]. As shown, our system reaches 62.1% precision, 50.3% recall and 55.6% F1 score. It achieves the state of the art in precision and F1 and brings approximate improvement of 6 percentage points in the performance. Due to the more stringent evaluation metrics adopted, the advantage of our model is remarkable.

We perform the ablation experiment to demonstrate the effectiveness of the innovations. Moreover, we can observe which innovation contributes most in relation extraction.

Table 1 The results of relation extraction on NYT

Model	P	R	F1
FCM	0.553	0.154	0.240
DS + logistic	0.258	0.393	0.311
LINE	0.335	0.329	0.332
MultiR	0.338	0.327	0.333
DS-Joint	0.574	0.256	0.354
CoType [40]	0.423	0.511	0.463
Bi-LSTM [6]	0.615	0.414	0.495
Bi-LSTM-IE	0.621	0.503	0.556

Highest scores of P (precision), R (recall), and F1 are indicated in bold

Firstly, a baseline is constructed by removing all the improvements we proposed. We input the 300-dimensional word embedding vectors instead of the 300-dimensional concatenated vectors into the standard Bi-LSTM model using a common tagging scheme, which tags non-entity words with “N” rather than “N+X.” Meanwhile, we do not add the input enhanced unit to the model. After that, the innovations are added to the baseline model separately and at last added simultaneously. The results are shown in Table 2. It can be seen that all the innovations improve the performance for relation extraction at different degrees, among which the word morphology encoding technique contributes to entity recognition most and the uncertain label scheme contributes to relation extraction most.

From the contrast between the first line and the second line, we can see that adding word morphology encoding vectors to the input improves the performance for relation extraction and entity recognition, and thus the word morphology is a useful feature for both entity recognition and relation extraction.

From the contrast between the first line and the third line, it can be seen that when the input information is strengthened by the enhanced unit, the performance for both the tasks is remarkably improved. It proves that there exists information loss in the standard Bi-LSTM model. Our input information enhanced unit is effective.

Table 2 The ablation experiments for the model

Model	F1 (ER)	F1 (RE)
Bi-LSTM	0.539	0.447
Bi-LSTM + word encodings	0.580	0.498
Bi-LSTM + enhanced unit	0.561	0.481
Bi-LSTM + uncertain labels	0.537	0.503
Bi-LSTM + all	0.592	0.556

Bold values indicate the highest scores of F1 both on total and on single item. The highest score of F1 on single item shows which part contributes to entity recognition or relation extraction most

ER entity recognition, RE relation extraction

From the contrast between the first line and the fourth line, we find that although the uncertain label scheme degrades entity recognition slightly, it contributes most in relation extraction and makes an increment of 5.6 percentage points in F1. It implies that using uncertain labels is better than tagging all non-entity words with a label for relation classification. As shown in the fifth line, when all of the innovations are added, our system achieves the best result. It proves the effectiveness of our model.

Our model is a system for relation extraction. When we substitute real values of the entity types for predicted values in the test stage, it can be used to classify relations. The results for relation classification is listed in Table 3. The first four are SVM classifiers, and the succeeding two are RNN-based models. These methods design a series of features and take advantage of a variety of toolkits to classify relations, the performance of which is dependent on the quantity of the feature sets and toolkits. The more features the input contains, the better the performance is. Although there is no need of auxiliary feature sets and toolkits for our model except word encodings and embeddings which are produced in an unsupervised manner, our model outperforms all the baselines.

XuCNN [21] is a CNN-based relation classification model working on dependency path between subjects and objects. Using WordNet and other features, it applied a negative sampling strategy and achieved an F1-macro score of 0.856.

So far, the state of the art on SemEval-2010 task 8 is Att-CNN [30], which exploited multi-level attention mechanism and a position representation method to capture both entity-specific attention and relation-specific pooling attention. It obtained the state-of-the-art result for relation classification with an F1 score of 0.880.

Our model is not as good as the state of the art for two reasons: one is that the model in this paper does not use any auxiliary toolkit and feature set. The other is that the models above are single relation classification systems, which use the entity position information and can only perform relation classification. So, these models have advantages in single relation classification tasks. Our model can perform not only single but also multiple relation extraction and classification tasks, in which a sentence contains one or more relation triples. It is a well-known fact that a single relation extraction task has the constant complexity, while a multiple relation extraction has the exponential complexity.

Our model achieves a good performance on NYT, which is a dataset for single relation extraction. In order to evaluate the performance on multiple relation extraction tasks, we conduct experiments on the multiple relation dataset ACE2005. We compare our model with the feature matrix joint model [36], the shortest dependency path tree (SPTree) [5] and the graph LSTM model [42]. As shown in

Table 3 The results of relation classification on SemEval 2010

Model	Feature sets and toolkits	F1-macro
SVM	POS, stemming, syntactic patterns	0.601
SVM	Word pair, words in between	0.725
SVM	POS, stemming, syntactic patterns, WordNet	0.748
SVM	POS, prefixes, morphology, WordNet, dependency ProBank, FrameNet, NomPlus, TextRunner...	0.822
RNN	Parser, POS, NER, WordNet	0.776
MVRNN	Syntactic parser	0.791
CDNN	Lexical feature	0.733
	Word pair and around words, WordNet, Sentence feature	0.827
SPTree	Dependency parser, POS	0.844
XuCNN	Dependency parser, words around nominals, WordNet	0.856
Att-CNN	Entity position	0.880
Our model	–	0853

Bold value indicates the highest scores of F1-macro

Table 4 The results of multiple relation extraction on the ACE2005 dataset

Model	P	R	F1
Joint model [36]	65.4	39.8	49.5
SPTree [5]	57.2	54.0	55.6
Graph LSTM [42]	61.5	52.3	56.5
Our model	63.1	51.6	56.8

Bold values indicate the highest scores of P (precision), R (recall) and F1

Table 4, our model reaches Precision of 63.1%, Recall of 51.6% and F1 of 56.8%. Although our model is not the best in precision and recall, it outperforms the baselines in F1 value and achieves a competitive performance on ACE2005. The results prove the effectiveness of our model on the multiple relation extraction task.

Dropout [47] and L-norm regularization are two strategies to avoid overfitting, the former attempts to randomly discard excess units and weight parameters in the training process; the latter tries to make the parameters equal zero as many as possible. We test them in our model separately. The results show that Dropout (dropout ratio 0.5) achieves approximately 0.9% improvement in relation classification task, but it can hardly influence the relation extraction task on NYT. The reason for this may be that the NYT dataset is so big that overfitting is not a main problem, and Dropout strategy is useful in the small dataset. However, L-2 regularization cannot improve either of the tasks. Therefore, this method is not fit to our model.

In order to evaluate the word encoding method in this paper, we compare the redundancy encoding with the existing methods of word morphology representation. On SemEval 2010 for relation classification, we construct three baselines which represent the three different methods:

The first is the one-hot encoding. Just as the work [1], we apply 56-dimensional one-hot vectors to encode 56 characters in our alphabet and then concatenate character vectors as the word morphology representation, which is discrete real-value vectors. As in our model, the word length is $WL = 9$.

The second is the CNN-based character-level embedding. As the method in [2, 4, 23], we randomly initialize 56 25-dimensional vectors drawn from a uniform distribution with range $[-1, 1]$ as the initial character embedding vectors. For each word, we employ a convolution and a max layer to extract an 81-dimensional feature vector as the word morphology vector, which, together with the character vectors, is modified in the training process.

Char2Vec [3] employed a Bi-LSTM model with attention to learn character-level embedding vectors which captured semantic information by means of the morphology. Like this work, we set a context window size of 3 and take 11 negative samples per positive sample and then train the model on the SemEval-2010 corpus.

We exploit these three different methods to produce the word morphology vectors and then concatenate them with the word embedding vectors as the input of our model. To compare fairly, the same dimension size and the same embedding vectors are used except for the one-hot encoding, which needs 56-dimensional character vectors and word length of 9 characters. The results are shown in Table 5, and our method outperforms the continuous real-value representation and the one-hot encoding. However, as a discrete representation, by virtue of its sparsity and small code distance, the one-hot encoding performs worst.

In order to observe the optimal word length, we fix all other parameters and modify WL among $\{5, 6, 7, 8, 9, 10, 11, 12\}$. The predicted results for relation extraction on the NYT dataset are shown in Fig. 6. We

Table 5 The results of relation classification on SemEval-2010 using different morphology representations

Model	DSWM	DSWE	F1-macro
The one-hot encoding	504	219	0.720
The CNN-based embedding	81	219	0.791
The Char2Vec	81	219	0.744
The redundancy encoding	81	219	0.853

Bold value indicates the highest scores of F1-macro

DSWM dimension size of the word morphology vectors, *DSWE* dimension size of the word embedding vectors

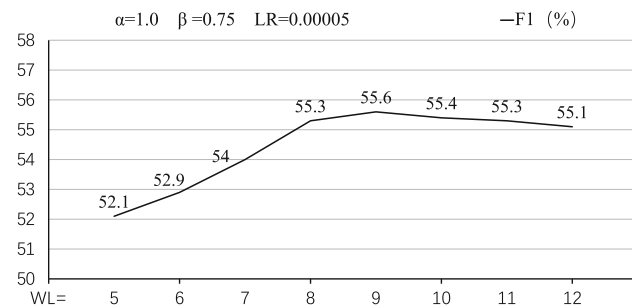


Fig. 6 The predicted results using different word lengths WL for relation extraction on the NYT dataset. The horizontal coordinate is the values of WL, and the vertical coordinate is F1 values. α , β are bias weights, and LR is the learning rate

can see that when WL is 9, the best performance is achieved. So, we can conclude that for our model of relation extraction, each word represented by the concatenation of the first 9 characters is the best choice.

5 Conclusion

We present a model not requiring auxiliary feature sets and toolkits for relation extraction. The results of the experiments have proved the effectiveness of the model, and we can obtain the following conclusions:

- Relation extraction is closely connected with the morphology of the keywords. Thus, the morphological feature is beneficial for this task.
- In the LSTM memory unit, there exists information loss and dilution. So, the enhanced unit is an effective improvement.
- The uncertain labels can reduce interference from the non-entity words.

We have made a discovery that each word represented by the first 9 letters is the best setting. Although our model is effective for the multiple relation extraction, it cannot deal with the overlapping relations. In future work, we will extend our model to solve the overlapping relation problems.

Acknowledgements The authors would like to thank Xiang Ren, Zeqiu, Wu and Wenqi He et al. for the public NYT dataset constructed by them. The authors are also grateful to Mikolov et al. for their public program training word embeddings. This research work is supported by the National Key Research and Development Program of China (Grant No. 2017YFB0803302), the National Natural Science Foundation of China (No. 61751201) and the National Key Research and Development Program of China (No. 2016QY03D0602).

Compliance with ethical standards

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

References

1. Zhang X, Zhao J, LeCun Y, (2015) Character-level convolutional networks for text classification. In: Proceedings of the 28th international conference on neural information processing systems - volume 1, NIPS'15, pp 649–657
2. Chiu J, Nichols E (2016) Named entity recognition with bidirectional LSTM-CNNs. *Trans Assoc Comput Linguist* 4:357
3. Cao K, Rei M (2016) A joint model for word embedding and word morphology. In: Proceedings of the 1st workshop on representation learning for NLP (Association for Computational Linguistics, 2016), pp 18–26. <https://doi.org/10.18653/v1/W16-1603>. <http://www.aclweb.org/anthology/W16-1603>
4. Ma X, Hovy E (2016) End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers) (Association for Computational Linguistics, 2016), pp 1064–1074. <https://doi.org/10.18653/v1/P16-1101>. <http://www.aclweb.org/anthology/P16-1101>
5. Miwa M, Bansal M (2016) Modeling joint entity and relation extraction with table representation. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers) (Association for Computational Linguistics, 2016), pp 1105–1116. <https://doi.org/10.18653/v1/P16-1105>. <http://www.aclweb.org/anthology/P16-1105>
6. Zheng S, Wang F, Bao H, Hao Y, Zhou P, Xu B (2017) Joint extraction of entities and relations based on a novel tagging scheme. In: Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: long papers) (Association for Computational Linguistics, 2017), pp 1227–1236. <https://doi.org/10.18653/v1/P17-1113>. <http://www.aclweb.org/anthology/P17-1113>
7. Hochreiter S, Schmidhuber J (1997) Backpropagation applied to handwritten zip code recognition. *Neural Comput* 9(8):1735. <https://doi.org/10.1162/neco.1997.9.8.1735>
8. Graves A, Jaitly N, Mohamed AR (2014) Hybrid speech recognition with deep bidirectional LSTM. *Automatic speech recognition and understanding IEEE*, pp 273–278
9. Hearst MA (1992) Automatic acquisition of hyponyms from large text corpora. In: COLING 1992 volume 2: the 15th international conference on computational linguistics. <http://www.aclweb.org/anthology/C92-2082>
10. Brin S (1999) Extracting patterns and relations from the World Wide Web. In: Selected papers from the international workshop on The World Wide Web and databases, WebDB '98. Springer, London, pp 172–183. <http://dl.acm.org/citation.cfm?id=646543.696220>
11. Agichtein E, Gravano L (2000) Snowball: extracting relations from large plain-text collections. In: Proceedings of the fifth

- ACM conference on digital libraries, DL '00. ACM, New York, pp 85–94. <https://doi.org/10.1145/336597.336644>
12. Blum A, Lafferty J, Rwebangira MR, Reddy R (2004) Semi-supervised learning using randomized mincuts. In: Proceedings of the twenty-first international conference on machine learning, ICML '04. ACM, New York, p 13. <https://doi.org/10.1145/1015330.1015429>
13. Oakes MP (2005) Using Hearst's rules for the automatic acquisition of hyponyms for mining a pharmaceutical corpus. In: International workshop text mining research, practice and opportunities, proceedings, Borovets, Bulgaria, 24 September 2005. Held in Conjunction with Ranlp 63–67
14. Chen J, Ji D, Tan C.L, Niu Z (2006) Relation extraction using label propagation based semi-supervised learning. In: Proceedings of the 21st international conference on computational linguistics and 44th annual meeting of the association for computational linguistics (Association for Computational Linguistics, 2006), pp 129–136. <http://www.aclweb.org/anthology/P06-1017>
15. Bunescu R, Mooney R (2007) Learning to extract relations from the Web using minimal supervision. In: Proceedings of the 45th annual meeting of the association of computational linguistics (Association for Computational Linguistics, 2007), pp 576–583. <http://www.aclweb.org/anthology/P07-1073>
16. Bollegala DT, Matsuo Y, Ishizuka M (2010) Relational duality: unsupervised extraction of semantic relations between entities on the Web. In: Proceedings of the 19th international conference on World Wide Web, WWW '10. ACM, New York, pp 151–160. <https://doi.org/10.1145/1772690.1772707>
17. Nakashole N, Tylanda T, Weikum G (2013) Fine-grained semantic typing of emerging entities. In: Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: long papers) (Association for Computational Linguistics, 2013), pp 1488–1497. <http://www.aclweb.org/anthology/P13-1146>
18. Zelenko D, Aone C, Richardella A (2003) Dropout: a simple way to prevent neural networks from overfitting. *Mach Learn Res* 3:1083
19. Bunescu RC, Mooney RJ (2005) Subsequence kernels for relation extraction. In: Proceedings of the 18th international conference on neural information processing systems, NIPS'05. MIT Press, Cambridge, pp 171–178. <http://dl.acm.org/citation.cfm?id=2976248.2976270>
20. Qian L, Zhou G, Kong F, Zhu Q, Qian P (2008) Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In: Proceedings of the 22nd international conference on computational linguistics (Coling 2008) (Coling 2008 Organizing Committee, 2008), pp 697–704. <http://www.aclweb.org/anthology/C08-1088>
21. Xu K, Feng Y, Huang S, Zhao D (2015) Semantic relation classification via convolutional neural networks with simple negative sampling. In: Proceedings of the 2015 conference on empirical methods in natural language processing (Association for Computational Linguistics, 2015), pp 536–540. <https://doi.org/10.18653/v1/D15-1062>. <http://www.aclweb.org/anthology/D15-1062>
22. Zhang H, Sun Y, Zhao M, Chow TWS, Wu QMJ (2019) Understanding subtitles by character-level sequence-to-sequence learning. *IEEE Trans Cybern*. <https://doi.org/10.1109/TCYB.2019.2900159>
23. dos Santos C, Guimarães V (2015) Boosting named entity recognition with neural character embeddings. In: Proceedings of the fifth named entity workshop (Association for Computational Linguistics, 2015), pp 25–33. <https://doi.org/10.18653/v1/W15-3904>. <http://www.aclweb.org/anthology/W15-3904>
24. Zhang H, Li J, Ji Y, Yue H (2017) Understanding subtitles by character-level sequence-to-sequence learning. *IEEE Trans Ind Inform* 13(2):616. <https://doi.org/10.1109/TII.2016.2601521>
25. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Long short-term memory. *Neural Comput* 1(4):541. <https://doi.org/10.1162/neco.1989.1.4.541>
26. Xu Y, Mou L, Li G, Chen Y, Peng H, Jin Z (2015) Classifying relations via long short term memory networks along shortest dependency paths. In: Proceedings of the 2015 conference on empirical methods in natural language processing (Association for Computational Linguistics, 2015), pp 1785–1794. <https://doi.org/10.18653/v1/D15-1206>
27. Xu Y, Jia R, Mou L, Li G, Chen Y, Lu Y, Jin Z (2016) Improved relation classification by deep recurrent neural networks with data augmentation. In: Proceedings of COLING 2016, the 26th international conference on computational linguistics: technical papers (The COLING 2016 Organizing Committee, 2016), pp 1461–1470. <http://www.aclweb.org/anthology/C16-1138>
28. Zhang S, Zheng D, Hu X, Yang M (2015) Bidirectional long short-term memory networks for relation classification. In: Proceedings of the 29th Pacific Asia conference on language, information and computation, pp 73–78. <http://www.aclweb.org/anthology/Y15-1009>
29. Zeng D, Liu K, Lai S, Zhou G, Zhao J (2014) Relation classification via convolutional deep neural network. In: Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers (Dublin City University and Association for Computational Linguistics, 2014), pp 2335–2344. <http://www.aclweb.org/anthology/C14-1220>
30. Wang L, Cao Z, de Melo G, Liu Z (2016) Relation classification via multi-level attention CNNs. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers) (Association for Computational Linguistics, 2016), pp 1298–1307. <https://doi.org/10.18653/v1/P16-1123>. <http://www.aclweb.org/anthology/P16-1123>
31. dos Santos C, Xiang B, Zhou B (2015) Classifying relations by ranking with convolutional neural networks. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: long papers) (Association for Computational Linguistics, 2015), pp 626–634. <https://doi.org/10.3115/v1/P15-1061>. <http://www.aclweb.org/anthology/P15-1061>
32. Vu NT, Adel H, Gupta P, Schütze H (2016) Combining recurrent and convolutional neural networks for relation classification. In: Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies (Association for Computational Linguistics, 2016), pp 534–539. <https://doi.org/10.18653/v1/N16-1065>. <http://www.aclweb.org/anthology/N16-1065>
33. Yang B, Cardie C (2013) Joint inference for fine-grained opinion extraction. In: Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: long papers) (Association for Computational Linguistics, 2013), pp 1640–1649. <http://www.aclweb.org/anthology/P13-1161>
34. Singh S, Riedel S, Martin B, Zheng J, McCallum A (2013) Joint inference of entities, relations, and conference. In: Proceedings of the 2013 workshop on automated knowledge base construction, AKBC '13. ACM, New York, pp 1–6. <https://doi.org/10.1145/2509558.2509559>. <http://doi.acm.org/10.1145/2509558.2509559>
35. Miwa M, Sasaki Y (2014) Modeling joint entity and relation extraction with table representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (Association for Computational Linguistics, 2014), pp 1858–1869. <https://doi.org/10.3115/v1/D14-1200>. <http://www.aclweb.org/anthology/D14-1200>

36. Li Q, Ji H, Incremental Joint Extraction of Entity Mentions and Relations. in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (Association for Computational Linguistics, 2014), pp. 402–412. <https://doi.org/10.3115/v1/P14-1038>. <http://www.aclweb.org/anthology/P14-1038>
37. Mintz M, Bills S, Snow R, Jurafsky D (2009) Distant supervision for relation extraction without labeled data. In: Proceedings of the joint conference of the 47th annual meeting of the acl and the 4th international joint conference on natural language processing of the AFNLP (Association for Computational Linguistics, 2009), pp 1003–1011. <http://www.aclweb.org/anthology/P09-1113>
38. Riedel S, Yao L, Mccallum A (2010) Modeling relations and their mentions without labeled text. In: European conference on machine learning and knowledge discovery in databases, pp 148–163
39. Hoffmann R, Zhang C, Ling X, Zettlemoyer L, Weld DS (2011) Knowledge-based weak supervision for information extraction of overlapping relations. In: Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies (Association for Computational Linguistics, 2011), pp 541–550. <http://www.aclweb.org/anthology/P11-1055>
40. Ren X, Wu Z, He W, Qu M, Voss C.R, Ji H, Abdelzaher TF, Han J (2017) CoType: joint extraction of typed entities and relations with knowledge bases. In: Proceedings of the 26th international conference on World Wide Web (International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2017), WWW '17, pp 1015–1024. <https://doi.org/10.1145/3038912.3052708>. <https://doi.org/10.1145/3038912.3052708>
41. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J (2013) Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th international conference on neural information processing systems - volume 2 (Curran Associates Inc., USA, 2013), NIPS'13, pp 3111–3119. <http://dl.acm.org/citation.cfm?id=2999792.2999959>
42. Peng N, Poon H, Quirk C, Toutanova K, Yih W (2017) Cross-sentence N-ary relation extraction with graph LSTMs. *Trans Assoc Comput Linguist* 5:101
43. Gormley MR, Yu M, Dredze M (2015) Improved relation extraction with feature-rich compositional embedding models. In: Proceedings of the 2015 conference on empirical methods in natural language processing (Association for Computational Linguistics, 2015), pp 1774–1784. <https://doi.org/10.18653/v1/D15-1205>. <http://www.aclweb.org/anthology/D15-1205>
44. Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q (2015) LINE: large-scale information network embedding (WWW World Wide Web Consortium (W3C), 2015). <https://www.microsoft.com/en-us/research/publication/line-large-scale-information-network-embedding/>
45. Socher R, Huval B, Manning CD, Ng AY (2012) Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning (Association for Computational Linguistics, 2012), pp 1201–1211. <http://www.aclweb.org/anthology/D12-1110>
46. Kingma D.P, Ba J.L (2015) Adam: A Method for Stochastic Optimization. international conference on learning representations
47. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res* 15(1):1929

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.