*future internet*

MDPI

*Article*

# Chinese Event Extraction Based on Attention and Semantic Features: A Bidirectional Circular Neural Network

**Yue Wu and Junyi Zhang ***

School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China; zhangjy1214@sina.cn

* Correspondence: z_kismet@shu.edu.cn; Tel.: +86-137-9526-6257

Received: 6 September 2018; Accepted: 24 September 2018; Published: 26 September 2018

**Abstract:** Chinese event extraction uses word embedding to capture similarity, but suffers when handling previously unseen or rare words. From the test, we know that characters may provide some information that we cannot obtain in words, so we propose a novel architecture for combining word representations: character–word embedding based on attention and semantic features. By using an attention mechanism, our method is able to dynamically decide how much information to use from word or character level embedding. With the semantic feature, we can obtain some more information about a word from the sentence. We evaluate different methods on the CEC Corpus, and this method is found to improve performance.

**Keywords:** event extraction; attention mechanism; feature representation; Bi-LSTM

## 1. Introduction

Event extraction [1] is one of the tasks involved in Automatic Content Extraction (ACE). The National Institute of Standards and Technology (NIST) sponsors ACE and aims to promote and develop relevant technologies for information extraction (IE). In 2005, ACE included event extraction tasks. The process of defining events in ACE is composed of event-trigger words (trigger) and event arguments (argument). The event extraction tasks include event recognition and event argument recognition. Event recognition is done to determine whether the event-trigger word relates to real-world events. Event argument recognition is done to recognize time, location, and participants. Table 1 is an example of the event arguments in the ACE Chinese Annotation Guidelines [2] for events.

**Table 1.** An example of event arguments.

| | |
|---|---|
| Trigger | 鄂州一司机将**[购得]**的货车假扮成军车。 |
| Participants | 云南大学法学院大三的男生**[陈俊耕]**在 2004 年这个暑假成立了自己的公司。 |
| Time | 联邦政府 **[2005年1月6日]**决定向东南亚遭受海啸袭击的受灾地区提供 5 亿 欧元的援助 |
| Location | 美英飞机轰炸**[伊拉克北部地区]**至少炸死 4 名平民。 |

Chinese event extraction has had some success in recent years.Numerous articles have proved the great role of word embedding in natural language processing, for example [3], which uses word embedding to represent the caption of an image. In addition, the application of deep learning also makes event extraction more effective. However, it still has two major problems:

1. Chinese has a lot of polysemy and cannot accurately represent word meaning with a single word embedding.
2. Chinese grammar is more complex and the same word has different affects in different sentences.

We followed the state-of-the-art methods of event extraction and proposed a new method according to the character of Chinese documents. The contributions of our study are as follows:

1. We use character-level embedding to generate word representation. By using an attention mechanism, our method can dynamically decide how much information to use from a word or character level embedding.
2. We represent the word semantic feature by multiple dimensions to obtain more information from the sentence and combine it with word embedding. This can improve the effect of the word representation.

The experimental results show that our method outperforms the state-of-the-art Chinese event extraction method.

## 2. Related Work

The existing event extraction methods are mainly based on rules and machine learning.

### *2.1. Rules*

Regarding rules, almost all of the state-of-the-art methods focus on processing one sentence at a time. They use a set of elaborately designed features that are extracted by textual analysis and linguistic knowledge.

Ji [4] was inspired by the hypothesis of One Sense Per Discourse [5]; they extended the scope from a single document to a cluster of topic-related documents and employed a rule-based approach to propagate a consistent trigger classification and event arguments across sentences and documents. By combining global evidence from related documents with local decisions, they obtained an appreciable improvement in both event and event argument identification.

Liao [6] proposed document level cross-event inference to improve event extraction. This work does not limit itself to time information for events, but rather uses related events and event-type consistency to make predictions or resolve ambiguities regarding a given event.

Li [7] proposed a joint framework based on structured prediction, which extracts triggers and arguments together so that the local predictions can be mutually improved.

### *2.2. Machine Learning*

Machine learning focuses on the discovery, selection, and combination of classifier construction and features, and regards event identification as a classification problem.

Chieu [8] first introduced the maximum entropy classifier into event extraction, and used the classifier to complete the identification of events and their arguments.

Ahn [9] proposed classifying event factor identification as a multivariate classification problem, and adopted the method of classification learning to realize event identification and event factor identification on an ACE English corpus.

Fu [10] proposed an algorithm of event factor identification based on feature weighting. This algorithm firstly improves the Relief feature selection algorithm in the classification algorithm and then applies it to the clustering algorithm. The improved Relief algorithm (FWA) assigns different weights to different contributions of clustering according to each feature, and then USES the KMeans algorithm to cluster the event arguments.

Chen [11] proposed a dynamic multi-pool convolutional neural network using a dynamic multi-pool layer to capture the sentence-level information and the information that was missing from the multi-event sentence.

Zhang [12] proposed a deep belief network model with mixed supervision to identify the trigger words.

Nguyen [13] proposed that event extraction be done in a joint framework with bidirectional recurrent neural networks. This framework can not only automatically learn the hidden features from the data, but also can reduce the error propagation of the pipeline method.

Previous works have only considered the effect of word embedding. In addition, most studies have not considered the semantic features of words.

Therefore, we propose a Chinese event extraction model, which uses a bidirectional circular neural network and word–character embedding based on attention and semantic features. It explore the semantic feature generalization of event extraction and studies the role of deep semantic information in event extraction. In this paper, we represent the word semantic feature by multiple dimensions and combined with the word itself to generate a feature vector using a bidirectional neural network for feature extraction and using the CRF model to obtain the best prediction results of sentences.

## 3. Event Extraction Task

We focused on the event extraction task of the Automatic Context Extraction (ACE) evaluation. ACE defines an event as something that happens or leads to some change of state. We employed the following terminology:

Event mention: A phrase or sentence in which an event occurs, including one trigger and an arbitrary number of arguments.

Event trigger: The main word that most clearly expresses an event occurrence.

Event argument: An entity mention, temporal expression or value that servers as a participant or attribute with a specific role in an event mention.

ACE annotates 8 types and 33 subtypes (e.g., attack, die, start position) for event mentioned that also correspond to the types and subtypes of the event triggers. Each event subtype has its own set of roles to be filled by the event arguments.

Although an event extraction system needs to recognize event triggers with specific subtypes and their corresponding arguments with the roles for each sentence, in this paper, we only wanted to recognize the event trigger and three event arguments (event participants, event location and event time).

## 4. Methodology

We formalized the EE task as follows:

Let $W = w_1w_2w_3, \ldots, w_n$ be a sentence where $n$ is the sentence length and $w_i$ is the $i$-th token. Additionally, let $E = e_1e_2e_3, \ldots, e_n$ be the labels in this sentence. We use the BIO annotation mode (“B-X” means the beginning of the X element, “I-X” means the middle (including the end) position of the X element, and “O” means not belonging to any type of elements).

For every token $w_i$ in the sentence, we need to predict its label (trigger, time, location, participator or O) and we must make the prediction as accurate as possible. Our method considers character embedding and word features in an effort to improve word embedding. Figure 1 presents the architecture of event trigger and arguments recognition, which primarily involves the following four components: (a) character–word embedding based on attention; (b) feature representation; (c) Bi-LSTM (Bidirectional Long Short-Term Memory network); and (d) CRF (Conditional Random Field).

### *4.1. Word Representation*

### Character Level Embedding

Most methods use the external context of the words in the large corpus to learn the embedding of words. However, in Chinese, a word is usually composed of multiple characters, each containing rich internal information, for example, the word “自行车” (bicycle). The CBOW model can be used to learn the meaning of a word, but it can also be learned by characters: “自” (auto), “行” (walk) and “车” (car). Because of the nature of semantics, the semantic meaning of internal characters may play an important role in semantic meaning. Therefore, an intuitive idea is to consider the internal characters for the embedding of learning words.

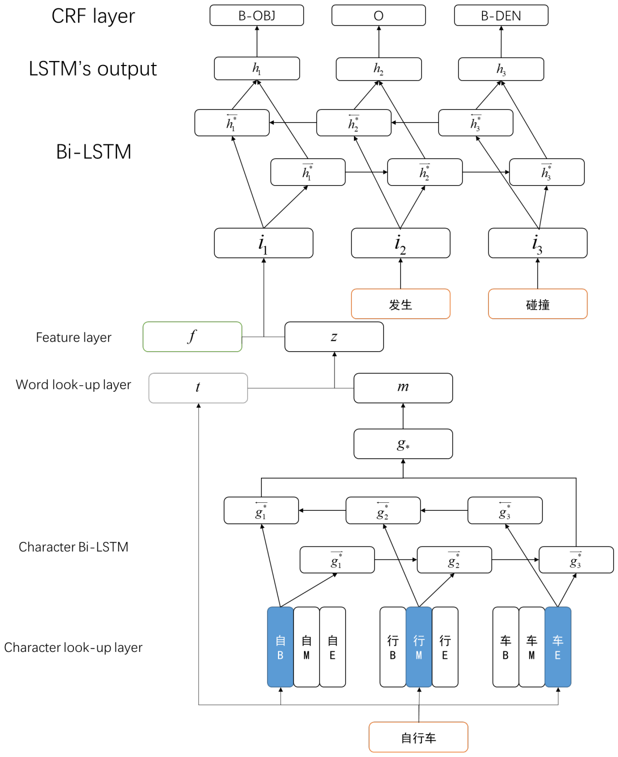

**Figure 1.** The architecture for the event trigger and argument recognition in the event extraction.

Because the word polysemy is more serious in Chinese, Chen [14] proposed the Position-based Character Embedding model. There are different representations according to the position of each character in the word (begin:(c$^B$); middle:(c$^M$); end:(c$^E$)).

For a word $x$, ($x = c_1c_2c_3, \ldots, c_j$) is the character embedding. We use the last hidden vectors from each of the LSTM [15] components and concatenate them together, as shown in Figure 2:

$$\overrightarrow{h_k^*} = LSTM(c_k^*, \overrightarrow{h_{k-1}^*}); \overleftarrow{h_k^*} \quad = LSTM(c_k^*, \overleftarrow{h_{k+1}^*}) \tag{1}$$

$$h^* = [\overrightarrow{h_j^*}, \overleftarrow{h_1^*}]; m = tanh(W_m h^*). \tag{2}$$

### Character–Word Embedding Based on Attention

We found that not all words are similar in meaning to their characters, and some foreign words are created from pronunciation, such as “巧克力” (chocolate), which is completely different from “巧” (artful), “克” (gram) and “力” (force). They only sound similar. At the same time, some characters may have different meanings but are in the same place.

Therefore, we need to jointly consider the weight relation between the word level embedding and the character level embedding. Based on Marek’s model, we let the last word embedding $z$ be:

$$z = \sigma(W_z^2 t + W_z^1 m) \tag{3}$$

where $t$ is the word embedding, and $W_z^1$, $W_z^2$ are weight matrices for calculating $z$ . They decide how much information to use from the character level embedding or from the word embedding. $\sigma()$ is the logistic function with values in the range [0;1]. $z$ has the same dimensions as $t$ or $m$ . This operation can also expand the word embedding dictionary. For words that have regular suffixes (or prefixes), we can share some character level features; words that did not previously have word vectors can also use character level embedding.

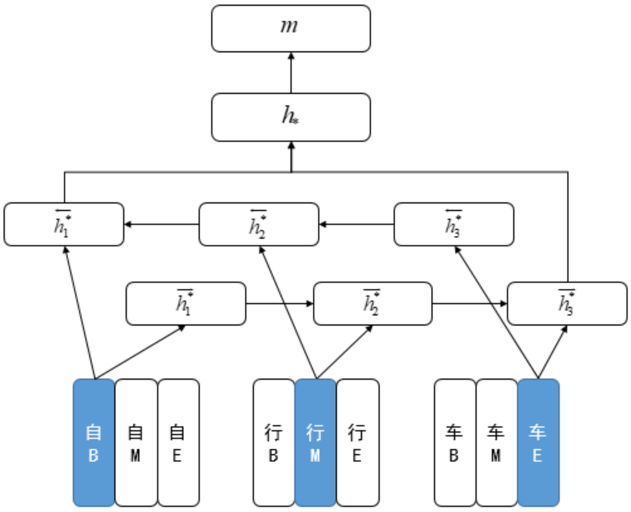

**Figure 2.** First, select the appropriate character embedding, then input it into LSTM, resulting in the last hidden vectors from each of the LSTM components and concatenate them

## 4.2. *Feature Representation*

Based on the results of semantic feature analysis, we defined three different feature abstraction layers. Feature representation mainly adopts the binary representation model. The characteristics of the three abstraction layers are the same as those of the value generation. The following describes the way in which each abstraction layer is represented and the way in which the eigenvalue is generated.

### Part of Speech

POS (part of speech), as a generalization of the word, plays an important role in language recognition, syntactic analysis, and event extraction. In a sentence, the grammatical component has a strong restriction on the speech component. For example, a noun or pronoun can act as a subject in a sentence, but an interjection cannot. Therefore, POS, as an abstract feature, conforms to the expression characteristics of text semantic information and the basic cognition of human beings in the recognition of events. Based on the analysis of the part of speech in CEC (Chinese Emergency Corpus), we found that the event-trigger word is usually a verb or noun, while the location and the participant argument are both nouns. Therefore, it is possible to improve the accuracy of recognition based on the abstract feature layer.

### Dependency Grammar

As DP (dependency grammar) exactly describes the semantic role relationships between words, it has extremely high semantic performance. DP does not pay attention to the meaning of the word itself, but expresses the word through the grammatical relation of the word in the sentence, which can directly obtain the deep semantic information.

Containing the most information and having the clearest expression in a sentence, trigger words, in a certain sense, play the role of predicate verbs of dependency grammar. Based on the analysis of the trigger in the CEC corpus, we found that 62% of the trigger words in the sentence play the dependent grammatical role of the HEAD and 18% play the role of the verb object. The predicate verbs in the dependent grammar and the trigger words in the sentences are mostly consistent.

### Distance from the HEAD

HEAD is the root node in dependency grammar, usually used to express the most important content in a sentence. The trigger is also used to express the core content in the event. Through the analysis of the CEC corpus, they are very close. Therefore, the distance from the HEAD can determine whether the word is a trigger or can recognize the event argument better.

### Feature Representation

Feature representation mainly uses a binary representation model. We describe all feature representations in detail below.

Feature representation of POS (part of speech): The 28 dimensions of POS feature vectors represent 28 types of POS. If the POS of a candidate word corresponds with that of a certain dimension representation of a feature vector, then the feature vector value of the word at the dimension is 1, whereas the feature vector value at the remaining 27 dimensions is 0.

Feature representation of DP (dependency grammar): On this feature layer, 15 vector dimensions exist, thus representing 15 kinds of DP.

Feature representation of DIS (distance from the HEAD): The vector dimension is 7, representing distance from 0 to 6. If it is greater than 6, then each feature vector value of the candidate word is 6.

Based on the above three features, we constructed the feature representation of a 50-dimensional vector. Table 2 shows some examples of feature representation.

**Table 2.** Examples of feature representation.

| Word | Type of Feature | Feature Vector Value |
|---|---|---|
| 学习 (learn) | POS<br>DP<br>DIS | 0000000000001000000000000000<br>000000000001000<br>1000000 |
| 汽车 (car) | POS<br>DP<br>DIS | 0000000000000000100000000000<br>000000000001000<br>0000100 |
| 上海市 (Shanghai) | POS<br>DP<br>DIS | 0000000000000000000001000000<br>000000100000000<br>0010000 |

The feature vector and the previous word vector are concatenated. The final vector is used to input the neural network:

$$i = [z; f]. \tag{4}$$

## 4.3. Bidirectional LSTM

We now describe the Bidirectional Long Short-Term Memory network for this paper. The model receives a sequence of tokens $(w_1, w_2, w_3, \ldots, w_n)$ as input and predicts a label corresponding to each of the input tokens.

First, a sequence of tokens $(w_1, w_2, w_3, \ldots, w_n)$ is obtained. The tokens are mapped to a distributed vector space, which we have explained, resulting in a sequence of word embeddings $(i_1, i_2, i_3, \ldots, i_n)$.

Next, the embeddings are given as input to two LSTM components moving in opposite directions through the text, creating context-specific representations. The respective forward- and backward-conditioned representations are concatenated for each word position, resulting in representations that are conditioned on the whole sequence:

$$\overrightarrow{h_i^*} = LSTM(c_i, \overrightarrow{h_{i-1}^*}); \overleftarrow{h_i^*} \quad = LSTM(c_i, \overleftarrow{h_{i+1}^*}) \tag{5}$$

$$h^* = [\overrightarrow{h^*_{N_j}}, \overleftarrow{h^*_1}] \tag{6}$$

Then, the results are mapped to the m dimension, resulting in a sequence of probability $(p_1, p_2, p_3, \ldots, p_n)$. $M$ is the number of entity types, and $p_{ij}$ describes how confident the network is that the label on the $w_i$ word is $j$.

Finally, a CRF is used, which conditions each prediction on the previously predicted label. In this architecture, the last hidden layer is used to predict the confidence scores for the word with each of the possible labels. A separate weight matrix is used to learn the transition probabilities between different labels, and the Viterbi algorithm is used to find an optimal sequence of weights. Given that $y$ is a sequence of labels $[y_1, y_2, y_3, \ldots, y_n]$, then the CRF score for this sequence can be calculated as

$$score(x,y) = \sum_{i=1}^{n} P_{i,y_i} + \sum_{i=1}^{n+1} A_{y_{i-1},y_i}. \tag{7}$$

$P_{i,y_i}$ shows how confident the network is that the label on the $w_i$ word is $y_i$. $A_{y_{i-1},y_i}$ shows the likelihood of transitioning from label $y_{i-1}$ to label $y_i$, and these values are optimized during training. The output from the model is the sequence of labels with the largest $score(x,y)$, which can be found efficiently using the Viterbi algorithm. In order to optimize the CRF model, the loss function maximizes the score for the correct label sequence, while minimizing the scores for all other sequences:

$$E = -score(x,y) + log \sum_{y \in Y} e^{s(y)}. \tag{8}$$

$Y$ is the set of all possible label sequences.

## 5. Experiments

### 5.1. Corpus and Evaluation Metric

We used Chinese Corpus CEC [16] as the corpus (https://github.com/shijiebei2009/CEC-Corpus). CEC is an event ontology corpus developed by the Semantic Intelligence Laboratory of Shanghai University. It has 332 articles. Compared with the ACE and TimeBank corpus, CEC is smaller, but more comprehensive in the annotation of events and event arguments. Therefore, we used CEC as the experimental data for the event extraction. The text was divided into five categories: earthquake, fire, traffic accident, terrorist attack, food poisoning. In the experiment, the word segmentation and grammar analysis tools adopted the module provided by the LTP language technology platform of Harbin Institute of Technology. After pretreatment, there were 8804 sentences, 5853 trigger words, 1981 event participants, 1633 event locations, and 1388 event times. For the experiment, we divided the corpus into 20% test set, 64% training set, and 16% verification set.

This experiment mainly recognized the event trigger, event participants, event time, and event location. We used recall (R), precision (P) and the F-measure (F) to evaluate the results.

### 5.2. Experiment Preparation

In the process of experimental preprocessing, we used Jieba (https://pypi.org/project/jieba/) as the tool for word segmentation and semantic analysis, and used the Sogou news data (http://www.sogou.com/labs/resource/ca.php) as the corpus to train the word embedding and the character embedding. There are many methods of training word embedding, which are introduced in the next chapter.

The unit number of LSTM was chosen to be 128, the size of word embedding was 300, that of character embedding was 300, that of feature representation embedding was 15, the maximum epoch was 20, the dropout was 0.5, and the batch size was 60. We used Adam (Kingma and Ba, 2014), with the learning rate of 0.001 for optimization.

### *5.3. Experiment Result*

We used the following methods as comparative experiments:

SKIPGRAM: Mikolov [17] first proposed this model of training word embedding. The SKIPGRAM model is intended to predict the surrounding words in the sentence given the current word.

CWE: Chen [15] proposed Character-Enhanced Word Embedding. CWE considers character embedding in an effort to improve word embedding.

CWEP: Chen [15] proposed the Position-based Character Embedding model. In the position-based CWE, various embeddings of each character are differentiated by the character position in the word, and the embedding assignment for a specific character in a word can be automatically determined by the character position.

Char-BLSTM-CRF: Misawa [18] proposed a neural model for predicting a tag for each character using word and character information. This model is like Chen's CWE but with a different method to concatenate the word embedding and the character embedding.

Segment-Level Neural CRF: Sato [19] presented Segment-level Neural CRF, which combines neural networks with a linear chain CRF for segment-level sequence modeling tasks such as named entity recognition (NER) and syntactic chunking. According to this article, the effect is slightly better than xuezhe's LSTM-CNN-CRF model. The experimental parameters of the model refer to this article. In addition, this experiment also provides three methods, including the use of two additional dictionary features. However, due to the lack of a dictionary in Chinese, dictionary features cannot be added.

Table 3 compares several advanced methods with our methods. It is divided into two parts: Word-based and Word–Character-based.

**Table 3.** Performance complexities.We used recall (R), precision (P) and the F-measure (F) to evaluate the results.

| | **(SKIPGRAM) BLSTM-CRF** | **Segment-Level Neural CRF** | **(CWE) BLSTM-CRF** | **(CWEP) BLSTM-CRF** | **Char-BLSTM-CRF** | **Our Method** |
|---|---|---|---|---|---|---|
| **Input** | **Word-Based** | | **Word–Character-Based** | | | |
| R | 74.963 | 78.153 | 75.855 | 77.231 | 75.279 | 80.574 |
| P | 70.455 | 76.864 | 73.668 | 74.495 | 73.597 | 79.856 |
| F | 72.709 | 77.509 | 74.762 | 75.863 | 76.438 | 80.215 |

**Word-Based Models**: Segment-level Neural CRF improved the CRF layer on the basis of LSTM-CNN-CRF, and then achieved a better effect. However, due to the relative complexity of the CRF model and the CNN layer, the time required was much longer than other models.

**Word–Character-Based Models**: The combination of word and character embedding is one of the main research directions at present. The four different methods are mainly due to the different combinations of word and character embedding. According to the last chapter, not all words are similar in meaning to their characters, so our method can achieve better performance. In addition, we considered the semantic feature of words in sentences, so the effect could be better than other methods. Of course, we used attention mechanisms when combining word embedding and character embedding and added a BLSTM layer when combining character embedding, so our method took about 10% more time than the Segment-level Neural CRF method.

Table 4 mainly analyzes the F1 value of each event argument in several methods. As we can see, the Segment-level Neural CRF method is the best method for identifying the location, while our method has the best effect on several dimensions: event time, trigger, and event participants.

**Table 4.** F-measure of each event argument.

| | Segment-Level Neural CRF | (CWEP) BLSTM-CRF | Our Method |
|---|---|---|---|
| Trigger | 76.143 | 76.837 | 81.647 |
| Time | 79.391 | 75.713 | 79.512 |
| Location | 79.487 | 74.521 | 79.339 |
| Participants | 77.509 | 76.379 | 80.363 |
| Average | 77.509 | 75.863 | 80.215 |

Table 5 shows the averaged word length after splitting an event into event arguments. As we can see, trigger is the shortest and location is the longest. Therefore, CNN can extract more information from location. This is the reason why Segment-level Neural CRF (BLSTM-CNNs-CRF) performs better than BLSTM-CRF in location.

**Table 5.** Average length of Chinese event arguments.

| | Trigger | Time | Location | Participants |
|---|---|---|---|---|
| length | 2.007 | 4.931 | 5.981 | 4.019 |

We also conducted a statistical analysis to mark error information. It was found that these problems are mainly due to the following reasons:

**The effect of incorrect participles**, for example, “住” (in hospital). This is an intransitive verb. However, according to traditional semantics, “住” (live) is a verb and “院” (hospital) is a noun. The presence of different semantic analysis results leads to mistakes. In this regard, the performance of the word segmentation tool needs to be improved.

**The effect of semantic ambiguity**. As we all know, a word can have different meanings in different contexts. This can be resolved by generating multiple word representations for the word. In additional, a word may become a different part of an event argument. For example, “酒店” (hotel) may be part of a location or a participant. Such problems require further integration of the dependencies between words and content. Our method provides an idea, but it may not be enough.

## 6. Conclusions

This paper proposes an effective Chinese event argument extraction model. The results show that our model is better than other models, but it uses much more time (about 10% more time). We propose an event extraction model that uses a bidirectional circular neural network and word–character embedding based on attention and semantic features. RNN has a natural sequence structure, so BLSTM-CRF is more widely used in natural language processing. Our future work will examine the reduction of redundancy in a character-based model by preparing and combining different BLSTMs for word and character inputs. In addition, we will focus more on marking the number of training sets with fewer problems in the future.

**Author Contributions:** J.Z. wrote the paper; Y.W. revised the paper.

**Funding:** This research was funded by [the National Key Scientific Instrument and Equipment Development Projects of National Natural Science Foundation of China] grant number [51227803].

**Acknowledgments:** We would like to thank our colleges for their suggestions and help.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Doddington, G.R.; Mitchell, A.; Przybocki, M.A.; Ramshaw, L.A.; Strassel, S.; Weischedel, R.M. The Automatic Content Extraction (ACE) Program-Tasks, Data, and Evaluation. In Proceedings of the LREC, Centro Cultural de Belem, Lisbon, Portugal, 26–28 May 2004; Volume 2, p. 1.

Tourille, J., Ferret, O., Neveol, A., & Tannier, X. (2017, July). Neural architecture for temporal relation extraction: a Bi-

2. Linguistic Data Consortium t Extraction) Chinese Annotation Guidelines for Events. 2005-05-05. Version 5.5. Av ww.ldc.upenn.edu/collaborations/past-projects/ace/ annotation-tasks-and-spec
3. Almgren, K.; Kim, M.; Lee, from the geometric shape of social network data using topological data analysis. *Entropy* **2017**, *19*, 360. [CrossRef]
4. Ji, H.; Grishman, R. Refining event extraction through cross-document inference. In Proceedings of the ACL-08: HLT, Columbus, OH, USA, 16–18 June 2008; pp. 254–262.
5. Yarowsky, D. Unsupervised word sense disambiguation rivaling supervised methods. In Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, Cambridge, MA, USA, 26–30 June 1995; Association for Computational Linguistics: Stroudsburg, PA, USA, 1995; pp. 189–196.
6. Liao, S.; Grishman, R. Using document level cross-event inference to improve event extraction. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; Association for Computational Linguistics: Stroudsburg, PA, USA, 2010; pp. 789–797.
7. Li, Q.; Ji, H.; Huang, L. Joint event extraction via structured prediction with global features. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; Volume 1, pp. 73–82.
8. Chieu, H.L.; Ng, H.T. A maximum entropy approach to information extraction from semi-structured and free text. *AAAI/IAAI* **2002**, 786–791.
9. Ahn, M.D. The stages of event extraction. In Proceedings of the Workshop on Annotating and Reasoning about Time and Events, Sydney, Australia, 23 July 2006; Association for Computational Linguistics: Stroudsburg, PA, USA, 2006; pp. 1–8.
10. Fu, J.; Liu, Z.; Zhong, Z.; Shan, J. Chinese event extraction based on feature weighting. *Inf. Technol. J.* **2010**, *9*, 184–187. [CrossRef]
11. Chen, Y.; Xu, L.; Liu, K.; Zeng, D.; Zhao, J. Event extraction via dynamic multi-pooling 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, Beijing, China, 26–31 July 2015; Volume 1, pp. 167–176.
12. Zhang, Y.; Liu, Z.; Zhou, W. Event recognition based on deep learning in Chinese texts. *PLoS ONE* **2016**, *11*, e0160147. [CrossRef] [PubMed]
13. Nguyen, T.H.; Cho, K.; Grishman, R. Joint event extraction via recurrent neural networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 300–309.
14. Chen, X.; Xu, L.; Liu, Z.; Sun, M.; Luan, H.B. Joint Learning of Character and Word Embeddings. In Proceedings of the IJCAI 2015, Buenos Aires, Argentina, 25 July–1 August 2015; pp. 1236–1242.
15. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
16. Liu, W.; Wang, X.; Zhang, Y.; Liu, Z. An Automatic-Annotation Method for Emergency Text Corpus. *J. Chin. Inf. Process.* **2017**, *2*, 012.
17. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
18. Misawa, S.; Taniguchi, M.; Miura, Y.; Ohkuma, T. Character-based Bidirectional LSTM-CRF with words and characters for Japanese Named Entity Recognition. In Proceedings of the First Workshop on Subword and Character Level Models in NLP, Copenhagen, Denmark, 7 September 2017; pp. 97–102.
19. Sato, M.; Shindo, H.; Yamada, I.; Matsumoto, Y. Segment-Level Neural Conditional Random Fields for Named Entity Recognition. In Proceedings of the Eighth International Joint Conference on Natural Language Processing, Taipei, Taiwan, 27 November–1 December 2017; Volume 2, pp. 97–102.