SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2018, Vol. 61 092106:1–092106:12
https://doi.org/10.1007/s11432-017-9359-x

A language-independent neural network for event detection

Xiaocheng FENG, Bing QIN* & Ting LIU

Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, *China*

Received 16 October 2017/Revised 8 December 2017/Accepted 7 February 2018/Published online 3 August 2018

Abstract Event detection remains a challenge because of the difficulty of encoding the word semantics in various contexts. Previous approaches have heavily depended on language-specific knowledge and pre-existing natural language processing tools. However, not all languages have such resources and tools available compared with English language. A more promising approach is to automatically learn effective features from data, without relying on language-specific resources. In this study, we develop a language-independent neural network to capture both sequence and chunk information from specific contexts and use them to train an event detector for multiple languages without any manually encoded features. Experiments show that our approach can achieve robust, efficient and accurate results for various languages. In the ACE 2005 English event detection task, our approach achieved a 73.4% F-score with an average of 3.0% absolute improvement compared with state-of-the-art. Additionally, our experimental results are competitive for Chinese and Spanish.

Keywords nature language processing, event detection, neural networks, representation learning

Citation Feng X C, Qin B, Liu T. A language-independent neural network for event detection. Sci China Inf Sci, 2018, 61(9): 092106, https://doi.org/10.1007/s11432-017-9359-x

1 Introduction

Event extraction is an important and fundamental task in nature language processing (NLP) and computational linguistics [1, 2]. Event extraction is crucial in understanding user generated text on social networks or breaking news [3, 4]. In this study, we focus on event detection [5–7], which performs as a vital role in the overall task of event extraction and serves as an intermediate step for the subsequent event extraction sub-tasks (e.g., argument identification and argument typing). Following the statement of automatic content extraction (ACE)[1], every event can be clearly expressed by some event triggers, which can be a verb or a phrase. Therefore, our goal is to extract these event triggers and precisely classify them into specific types.

Event detection or event trigger extraction is a crucial and challenging sub-task of event extraction, because the same event might appear in the form of various trigger expressions and an expression might represent different event types in different contexts. Figure 1 shows two examples. In S1, “release” is a verb concept and a trigger for “Transfer-money” event, while in S2, “release” is a noun concept and a trigger for “Release-parole” event.

Event detection is typically regarded as a type of text classification problem in literature. Most of the previous methods [8–12] built an event detector with numerous lexical and syntactic features. Although

* Corresponding author (email: qinb@ir.hit.edu.cn)

[1] https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-events-guidelines-v5.4.3.pdf.

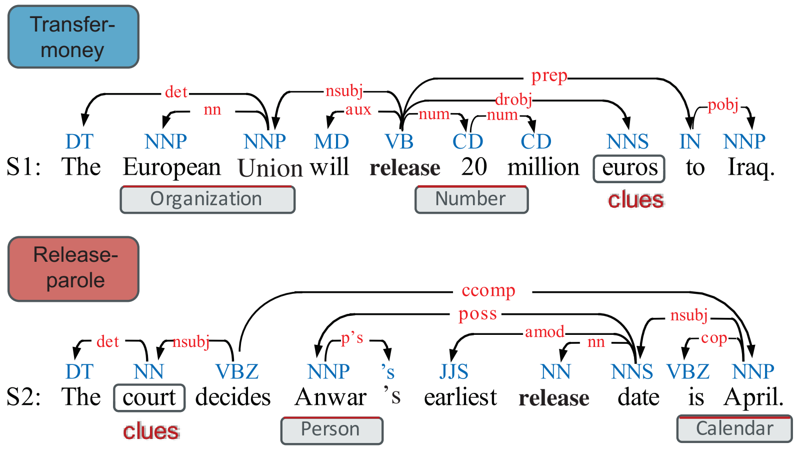
Transfer-money
det
nn
nsubj
aux
prep
drobj
num
num
pobj
DT NNP NNP MD VB CD CD NNS IN NNP
S1: The European Union will **release** 20 million euros to Iraq.
Organization
Number
clues
Release-parole
ccomp
poss
det
nsubj
p's
amod
nn
nsubj
cop
DT NN VBZ NNP 's JJS NN NNS VBZ NNP
S2: The court decides Anwar 's earliest **release** date is April.
clues
Person
Calendar

Figure 1 (Color online) Event type and syntactic parser results of an example sentence.

Table 1 Top 8 most similar words (in three clusters)

Injure	Score	Fight	Score	Fire	Score
Injures	0.602	Fighting	0.792	Fires	0.686
Hurt	0.593	Fights	0.762	Aim	0.683
Harm	0.592	Battle	0.702	Enemy	0.601
Maim	0.571	Fought	0.636	Grenades	0.597
Injuring	0.561	Fight	0.610	Bombs	0.585
Endanger	0.543	Battles	0.590	Blast	0.566
Dislocate	0.529	Fighting	0.588	Burning	0.562
Kill	0.527	Bout	0.570	Smoke	0.558

such approaches perform reasonably well, features are often derived from language-specific resources and output of pre-existing natural language processing toolkits (e.g., name tagger and dependency parser), thereby making these methods difficult to apply to other languages. Additionally, feature engineering is labor intensive and prone to error propagation. For example, in S2, when predicting the type of a trigger candidate “release”, the clue word “court” can help the classifier label “release” as a trigger of a “Release-parole” event. However, for feature engineering methods, establishing a relation between “court” and “release” is difficult, because no direct dependency path exists between them.

We overcome these problems by developing a neural network based approach that focuses on learning a sufficient representation of each word with the whole sequence information for trigger detection. Specifically, distributed representation learning has been widely used in modeling complex structures and has proven to be effective for many NLP tasks, such as machine translation [13,14], relation extraction [15,16] and sentiment analysis [17]. We argue that the distributed representation also helps to improve event detection results. Following the distributional hypothesis [18], we obtain similar words like those shown in Table 1, when we simply learn general word embeddings from a large corpus (e.g., Wikipedia) for each word. We can see that similar words, such as those centered around “injure”, “fight” and “fire”, converge to similar types.

In this paper, we also find that sequence and chunk are two types of meaningful language-independent structures for event detection. For example, considering S2 again, when extracting the trigger “release”, the detector can utilize the forward sequence information to capture the semantic of the clue word “court”. In addition, in S1, “European Union” and “20 million euros” are two chunks indicating that this sentence is related to an organization and financial activities, respectively. These clues greatly help in inferring “release” as a trigger of a “Transfer-money” event. Therefore, our hybrid neural network (HNN) incorporates two types of neural networks (i.e., bidirectional LSTM (Bi-LSTM) and convolutional neural network (CNN)) to model both sequence and chunk information from free contexts.

We evaluate our system on the event detection task for various languages with available ground-truth event detection annotations. Our approach achieved a 73.4% F-score with an average of 3.0% absolute improvement compared with state-of-the-art in English event detection task. The results are also competitive for Chinese and Spanish. We demonstrate that our combined model outperforms traditional feature-based methods with respect to generalization performance across languages because of (i) its capacity to model the semantic representations of each word by capturing both sequence and chunk information, and (ii) the use of word embeddings to induce a more general representation for trigger candidates.

The major contributions of this work are as follows:

- We present a novel neural network approach by integrating Bi-LSTM and CNN for sentence-level event detection.
- We report empirical results on ACE datasets and demonstrate that our approach outperforms the state-of-the-art event trigger extraction model for English.
- Our model is language independent without any supervised NLP tools and resources, and shows promising performance in Chinese and Spanish event trigger extraction.

The remainder of this paper is organized as follows. Section 2 introduces the related work. Section 3 presents an overview of the neural architecture, including the trigger detector and training process. Section 4 demonstrates the experiments and Section 5 concludes this study.

This paper is a substantial extension of our earlier work in [19]. We herein add new experimental results, a comprehensive description of the models, more details about our method, and an comprehensive analysis of the results.

2 Related work

Event detection is a fundamental problem in information extraction and NLP [11, 20], which aims to detect the event trigger of a sentence [9]. The majority of the existing methods regard this problem as a classification task and use machine learning methods with hand-crafted features, such as lexical (e.g., full word, pos tag), syntactic (e.g., dependency features) and external knowledge (WordNet) features [21]. Ji and Grishman [9] combined global evidence from related documents with local decisions for event extraction. Some studies leveraged richer evidence, such as cross-entity [8] and joint inference [22]. Other researchers treat event extraction as the task of predicting the structure of an event in a sentence. McClosky [23] portrayed the problem of biomedical event extraction as a dependency parsing problem. Goyal et al. [24] tried to use a distributional semantic model (DSM) to represent events. However, the DSM ignores the structure within the context, thus reducing the distribution to a bag of words. Li [22] presented a joint framework for ACE event extraction based on structured perceptron with beam search. To derive more information from the sentence, they proposed to extract entity mentions, relations and events in the ACE task based on the unified structure.

Despite the effectiveness of the feature-based methods, we argue that manually designing feature templates is typically labor intensive. Moreover, feature engineering requires expert knowledge and rich external resources, which may not be available for low-resource languages. Furthermore, a desirable approach should automatically learn informative representations from data so that it could be easily adapted to different languages. Neural network has recently emerged as a powerful method of automatically learning text representation from data and performed well various NLP tasks. For event detection, two recent studies [5, 20] explored the neural network to learn continuous word representation and regard it as the feature to infer whether a word is a trigger. Nguyen [5] presented a CNN with entity type information and word position information as extra features. However, their system limits the context to a fixed window size, which leads to the loss of word semantic representation for long sentences.

Here, we introduce a HNN to learn continuous word representation. Compared with the feature-based approaches, our method does not require feature engineering and can be directly applied to different languages. Compared with the previous neural models, we retain the advantage of CNN [5] in capturing

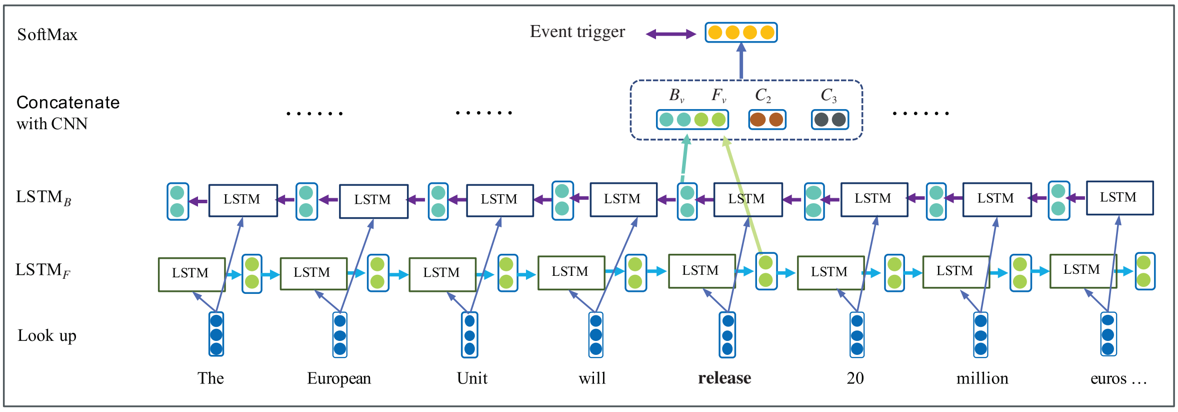

Figure 2 (Color online) Illustration of our model for the event trigger extraction (the trigger candidate here is "release"). $\boldsymbol{F}_v$ and $\boldsymbol{B}_v$ are the output of Bi-LSTM, while $\boldsymbol{C}_2$ and $\boldsymbol{C}_3$ are the output of CNN with convolutional filters with widths of 2 and 3, respectively.

local contexts. Besides, we also incorporate a Bi-LSTM to model the preceding and following information of a word because studies demonstrate that LSTM excels in capturing long-term dependencies in a sequence [17, 25].

3 Our approach

In this section, we introduce our neural network, which combines Bi-LSTM and CNN to learn a continuous representation for each word in a sentence (Figure 2). This representation is used to predict whether the word is an event trigger. We first used a Bi-LSTM to encode the semantics of each word with its preceding and following information. We then added a CNN to capture the structure information from local contexts. Taking advantage of the word semantic representation, our approach did not rely on any language-specific resources or complex features (syntactic or dependency parsing) and thus was easily adapted to multiple languages.

3.1 Bi-LSTM

In this subsection, we describe the Bi-LSTM model for event detection. The Bi-LSTM is a type of bidirectional recurrent neural network (RNN), which can simultaneously model the word representation with its preceding and following information. Word representations can be considered as features to detect triggers and their event types.

The power of Bi-LSTM lies in its ability to capture long-term dependencies in a sequence from both directions. Bi-LSTM has been successfully used in many natural language processing tasks [13, 17, 26]. This successful usage makes Bi-LSTM for event extraction contain richer sentence information than majority of the previous methods, which only used event arguments [11, 22] or dependence information [27]. Additionally, sequence information is a general and reliable structure for most languages.

Figure 2 presents the details of Bi-LSTM for event trigger extraction. We utilize a similar approach [20] and take every words of the sentence as the input. Each token is transformed by looking up word embeddings. We specifically use the skip-gram model to pre-train the word embeddings to represent each word. This model is one of the state-of-the-art models used to capture the distributed representation for many NLP tasks [28, 29]. All word vectors are stacked in a word embedding matrix $\boldsymbol{L}_w \in \mathbb{R}^{d\times|V|}$, where d is the dimension of the word vector and $|V|$ is the vocabulary size.

Figure 2 shows that Bi-LSTM comprises two LSTM neural networks, namely a lower one LSTM$_F$ and an upper one LSTM$_B$, to model the preceding and following contexts, respectively. The input of LSTM$_F$ is the preceding contexts plus the word as the trigger candidate. The input of LSTM$_B$ is the following contexts plus the word as the trigger candidate. We run LSTM$_F$ from the beginning to the end of a sentence, and run LSTM$_B$ from the end to the beginning of a sentence. Afterwards, we concatenate the

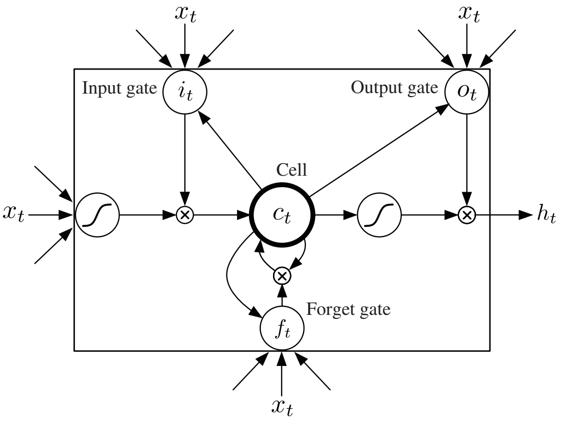

Figure 3 LSTM cell.

output $\boldsymbol{F}_v$ and $\boldsymbol{B}_v$ of LSTM$_F$ and LSTM$_B$, respectively and feed them into a softmax layer to classify the event type of the current word. Alternatively, the last hidden vectors of LSTM$_F$ and LSTM$_B$ could be averaged or summed as well.

Additionally, compared with the standard RNN, Bi-LSTM does not face the problem of gradient vanishing or exploding [30]. The reason lies in LSTM's usage of a more sophisticated and powerful LSTM cell as the transition function, such that long-distance semantic correlations in a sequence could be better modeled.

Figure 3 illustrates a single LSTM memory cell. We can see that the LSTM cell contains three multiplicative gates: the input $\boldsymbol{i}$, output $\boldsymbol{o}$ and forget gates $\boldsymbol{f}$, which provide continuous analogs of writing, reading and resetting operations for the cells, respectively. More precisely, the input to the cells is multiplied by the activation of the input gate; the output to the net is multiplied by that of the output gate; and the previous cell values are multiplied by the forget gate. The net can only interact with the cells via the gates. The LSTM cell is calculated as follows:

$$\begin{aligned}
\boldsymbol{g}^{(t)} &= \phi(W_{gx}\boldsymbol{x}^{(t)} + W_{gh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_g),\\
\boldsymbol{i}^{(t)} &= \sigma(W_{ix}\boldsymbol{x}^{(t)} + W_{ih}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i),\\
\boldsymbol{f}^{(t)} &= \sigma(W_{fx}\boldsymbol{x}^{(t)} + W_{fh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f),\\
\boldsymbol{o}^{(t)} &= \sigma(W_{ox}\boldsymbol{x}^{(t)} + W_{oh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o),\\
\boldsymbol{s}^{(t)} &= \phi(\boldsymbol{g}^{(t)} \odot \boldsymbol{i}^{(t)} + \boldsymbol{s}^{(t-1)} \odot \boldsymbol{f}^{(t)}),\\
\boldsymbol{h}^{(t)} &= \boldsymbol{s}^{(t)} \odot \boldsymbol{o}^{(t)}.
\end{aligned}$$

The value of the hidden layer at time t is the vector $\boldsymbol{h}^{(t)}$, whereas $\boldsymbol{h}^{(t-1)}$ is the value output by each memory cell in the hidden layer at the previous time. $\boldsymbol{x}$ is the input word vector. We use tangent ϕ as the input node $\boldsymbol{g}$ following the state-of-the-art design of [31]. The activation function of gates $\boldsymbol{i}$, $\boldsymbol{o}$ and $\boldsymbol{f}$ is the sigmoid σ. W_{gx}, W_{gh}, W_{ix}, W_{ih}, W_{fx}, W_{fh}, W_{ox} and W_{oh} are the parameters of input node $\boldsymbol{g}$ and gates $\boldsymbol{i}$, $\boldsymbol{o}$ and $\boldsymbol{f}$.

3.2 Convolution neural network

In Subsection 3.1, we introduced the way to model a word semantic representation via the Bi-LSTM model using its preceding/following history information. We also argued that local context is extremely useful in detecting the trigger and event type. For example, "take over" and "take off" have the same word "take", but they indicate diverse event types. The former means "transfer-ownership", whereas the latter denotes "transport".

A CNN excels in capturing salient features from a sequence of objects [32]. Hence, we designed a CNN to capture some local chunks. This approach was been used for event detection in previous studies [5,20]. we specifically used multiple convolutional filters with different widths to produce the local context

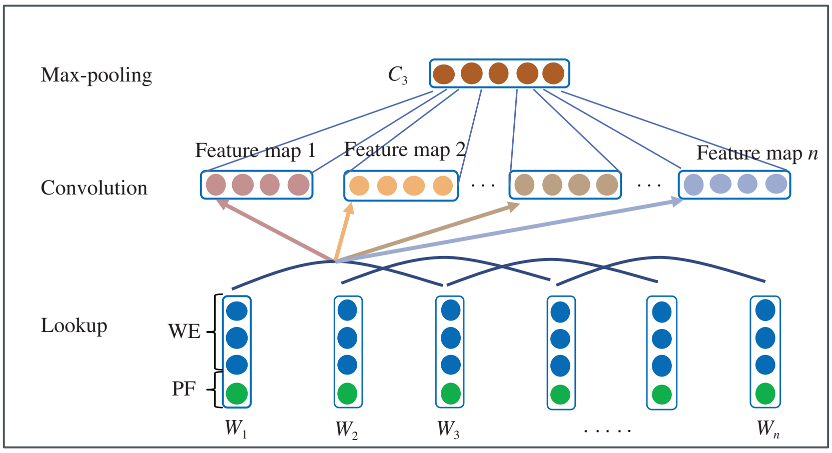

Figure 4 (Color online) CNN structure.

representation because they can capture the local semantics of n-grams of various granularities, which are proven powerful for event detection. In our work, multiple convolutional filters with width of 2 and 3 encode the semantics of bigrams and trigrams in a sentence. This local information can also help our model fix some errors due to lexical ambiguity.

Figure 4 illustrates the CNN with three convolutional filters. Let us consider a sentence comprising n words as $\{w_1, w_2, \ldots, w_i, \ldots, w_n\}$. Each word w_i is mapped to its embedding representation $\boldsymbol{e}_i \in \mathbb{R}^d$. Additionally, we add a position feature (PF) defined as the relative distance between the current word and the trigger candidate. A convolutional filter is a list of linear layers with shared parameters. Let l_{cf} be the width of a convolutional filter, and let $\boldsymbol{W}_{\mathrm{cf}}$ and $\boldsymbol{b}_{\mathrm{cf}}$ be the shared parameters of the linear layers in the filter. The input of a linear layer is the concatenation of the word embeddings in a fixed-length window size l_{cf}, which is denoted as $\boldsymbol{u}_{\mathrm{cf}} = [\boldsymbol{e}_i; \boldsymbol{e}_{i+1}; \ldots; \boldsymbol{e}_{i+l_{\mathrm{cf}}-1}] \in \mathbb{R}^{d \times l_{\mathrm{cf}}}$. The output of a linear layer is calculated as follows:

$$\boldsymbol{O}_{\mathrm{cf}} = \boldsymbol{W}_{\mathrm{cf}} \cdot \boldsymbol{u}_{\mathrm{cf}} + \boldsymbol{b}_{\mathrm{cf}},$$

where $\boldsymbol{W}_{\mathrm{cf}} \in \mathbb{R}^{d \times l_{\mathrm{cf}}}$, $\boldsymbol{b}_{\mathrm{cf}} \in \mathbb{R}^{\mathrm{len}}$. We denote len as the output length of linear layer. We feed the output of a convolutional filter to a MaxPooling layer and obtain an output vector with a fixed length to capture the semantics of local contexts.

3.3 Output

Finally, we concatenate the bidirectional sequence features: $\boldsymbol{F}_v$ and $\boldsymbol{B}_v$, which are learned from the Bi-LSTM, and the local context features: $\boldsymbol{C}_2$ and $\boldsymbol{C}_3$, which are the output of the CNN with convolutional filters having widths of 2 and 3, as a single vector $\boldsymbol{O} = [\boldsymbol{F}_v, \boldsymbol{B}_v, \boldsymbol{C}_2, \boldsymbol{C}_3]$. We then exploit a softmax approach to identify the trigger candidates and classify each trigger candidate as a specific event type. The softmax function is calculated as follows, where C is the number of event types:

$$\mathrm{softmax}_i = \frac{\exp(x_i)}{\sum_{i'=1}^{C} \exp(x_{i'})}.$$

3.4 Training

We train all aforementioned models using a generic stochastic gradient descent (SGD) forward and backward training procedure. The loss function in our model is the cross-entropy errors of the event trigger identification and trigger classification.

$$\mathrm{loss} = -\sum_{s \in T} \sum_{w \in s} \sum_{c=1}^{C} P_c^g(w) \cdot \log(P_c(w)),$$

Table 2 Hyperparameters used in our experiments on three languages

Language	Word embedding		Gradient learning method	
	Embedding corpus	Embedding dimension	Learning method	Parameters
English	NYT	300	SGD	learning rate $r = 0.03$
Chinese	Gigword	300	Adadelta	$p = 0.95$, $\delta = 1\mathrm{e}^{-6}$
Spanish	Gigword	300	Adadelta	$p = 0.95$, $\delta = 1\mathrm{e}^{-6}$

Table 3 # of documents

Data set	English ACE2005	Chinese ACE2005	Spanish ERE
Train set	529	513	93
Dev set	30	60	12
Test set	40	60	12

where T is the training data; s is a sentence and w is a word in the sentence. We regard each sentence in each epoch as a batch. In the event trigger identification task, C is the binary value (1 indicates a word is a trigger, while 0 indicates it is not a trigger). In the trigger classification task, C is the number of event types. $P_c(w)$ is the probability of predicting w as type c given by the softmax layer, whereas $P_c^g(w)$ indicates whether class c is the correct classification result with a value of 0 or 1. We differentiate the loss function through back-propagation with all the related parameters. We initialize all parameters to form a uniform distribution $U(-0.01, 0.01)$. We set the widths of the convolutional filters as 2 and 3. The number of feature maps is 300, and the PF dimension is 5. Table 2 illustrates the setting parameters used for the three languages in our experiments [33].

4 Experiments

We applied the developed approach for event detection on various data sets and evaluated the effectiveness separately [1, 2]. Here, we focused on the event trigger identification and event trigger classification tasks defined in the ACE evaluation, where an event is defined as a specific occurrence. More precisely, our first was to extract the event triggers without the need to classify them. The second task involved identifying the event triggers and classifying them into specific types. In this section, we describe the detailed experimental settings and discuss the results.

4.1 Dataset and evaluation

We evaluated the proposed approach on various languages (i.e., English, Chinese, and Spanish) with Precision (P), Recall (R) and F-measure (F), respectively. Table 3 shows the detailed description of the data sets used in our experiments. We utilized the ACE2005 corpus and followed the settings in the previously reported studies [10, 20, 34, 35] for English[2] and Chinese[3]. We used ERE (annotation of entities, relations, and events)[4] as the benchmark corpus for Spanish, since there are no previous work on event evaluation. As the first event extraction system on this corpus, we used a 10-fold cross validation as the evaluation metric. We abbreviate our model as HNNs.

4.2 Baseline methods

We compared our approach with the following baseline methods.

(1) MaxEnt, a baseline feature-based method, which trains a maximum entropy classifier with some lexical and syntactic features [9].

(2) Cross-event [12], using document-level information to improve the performance of the ACE event extraction.

[2] English data set. https://catalog.ldc.upenn.edu/LDC2008T19.

[3] Chinese data set. https://catalog.ldc.upenn.edu/Chinese Gigaword Fifth Edition.

[4] Spanish data set. https://catalog.ldc.upenn.edu/LDC96S35.

Table 4 Comparison of different methods on the English event detection

Model	Trigger identification			Trigger classification		
	Precision	Recall	F-score	Precision	Recall	F-score
MaxEnt	76.2	60.5	67.4	74.5	59.1	65.9
Cross-event	N/A	N/A	N/A	68.7	68.9	68.8
Cross-entity	N/A	N/A	N/A	72.9	64.3	68.3
Joint model	76.9	65.0	70.4	73.7	62.3	67.5
PSL	N/A	N/A	N/A	75.3	64.4	69.4
PR	N/A	N/A	N/A	68.9	**72.0**	70.4
CNN	80.4	67.7	73.5	75.6	63.6	69.1
RNN	73.2	63.5	67.4	67.3	59.9	64.2
LSTM	78.6	67.4	72.6	74.5	60.7	66.9
Bi-LSTM	80.1	69.4	74.3	81.6	62.3	70.6
FN	N/A	N/A	N/A	77.6	65.2	70.7
ANN	N/A	N/A	N/A	76.8	67.5	71.9
HNN	**80.8**	**71.5**	**75.9**	**84.6**	64.9	**73.4**

(3) Cross-entity [8], extracting events using cross-entity inference.

(4) Joint model [22], a joint structured perception approach incorporating multi-level linguistic features to simultaneously extract event triggers and arguments such that local predictions can be mutually improved.

(5) PSL [36], Liu's probabilistic soft logic model employs both latent local and global information for event detection reported to be the best feature-based system.

(6) Pattern recognition [27], using a pattern expansion technique to extract the event triggers.

(7) CNN [20], which exploits a dynamic multi-pooling CNN for event trigger detection.

(8) FN [37], Liu's FN-Based approach leverages the annotated corpus of FrameNet to alleviate data sparseness problem of event detection based on the observation that frames in FrameNet are analogous to events in ACE.

(9) ANN [38], a system explicitly exploiting argument information for event detection via supervised attention mechanisms.

4.3 Comparison on English

Table 4 shows the overall performance of all methods on the ACE2005 English corpus. Our approach significantly outperformed all previous methods. The better performance of HNN can be further explained by the following points:

(1) Compared with the feature-based methods, such as MaxEnt, Cross-event, Cross-entity, and Joint model, neural network-based methods, including CNN, Bi-LSTM and HNN, performed better because they can better utilize of word semantic information and avoid the errors propagated from the NLP tools, which may hinder the performance for event detection.

(2) Bi-LSTM can capture both preceding and following sequence information, which are much richer than the dependency path. For example, in S2, the semantic of "court" can be delivered to release by a forward sequence in our approach, which is an important clue that can help to predict "release" as a trigger for "release-parole". Explicit feature-based methods cannot establish a relation between "court" and "release" because they belong to different clauses, and no direct dependency path exists between them. However in our approach, the semantics of "court" can be delivered to release by a forward sequence.

(3) CNN can capture structured context information, which is useful for predicting the event type of a trigger candidate. For example, "take over" and "take off" have the same word "take", but they indicate diverse event types. The former means "transfer-ownership" whereas the latter denotes "transport". Therefore, combining Bi-LSTM and CNN, we can achieve a 5.4% performance improvement on trigger

Table 5 Case study for English event detection

English sentence examples	Li [11]	Chen [20]	Our method
Davies is **leaving** (end-position) to become chairman of the London school of economics, one of the best-known parts of the University of London.	Missing error	Classification error	Correct
Palestinian security forces returned Monday to the positions they held in the Gaza Strip before the outbreak of the 33-month Palestinian **uprising** (attack) as Israel removed all major checkpoints in the coastal territory, a Palestinian security source said.	Missing error	Correct	Correct
U.S. and British troops were moving on the strategic southern port city of Basra Saturday after a massive aerial assault **pounded** (attack) Baghdad at dawn.	Missing error	Missing error	Correct
Thousands of Iraq's majority Shiite Muslims **marched** (transport) to their main mosque in Baghdad to mark the birthday of Islam's founder Prophet Mohammed.	Classification error	Correct	Correct

identification's F-measure over the joint model and 3% gain on trigger classification F-measure over the state-of-the-art (pattern recognition).

(4) Cross-entity system achieves a higher recall because it uses not only sentence-level information but also document-level information. It utilizes event concordance to predict a local trigger's event type based on cross-sentence inference. For example, an "attack" event is more likely to occur with a "killed" or "die" event rather than a "marry" event. However, this method heavily relies on lexical and syntactic features, thus the precision is lower than neural-based methods.

(5) RNN and LSTM perform marginally worse than Bi-LSTM. An obvious reason is that RNN and LSTM only consider the preceding sequence information of the trigger, which may result in missing some important following clues. Considering S1 again, when extracting the trigger "releases", both models will miss the following sequence "20 million euros to Iraq", which may seriously hinder the performance of RNN and LSTM for event detection.

Table 5 lists some real cases of different methods for event detection. "Missing error" means the system fails to detect the word as an event trigger, whereas the "classification error" means the event type is not correct though the system can identify the word as a trigger. Neural network-methods, including [20] and our method can extract more event triggers. The reason might be that neural network could take advantage of lexical probabilities in a manner that is difficult to capture with limited size of the training corpus. Additionally, we find that our model is good at capturing long distance information. For example, in the first sentence, London School and University of London are clues for predicting the event type of "leaving", both of which contribute much to inferring the tag of "leaving".

4.4 Comparison on Chinese

We followed a previous work [34] and employed language technology platform [39] to perform word segmentation for Chinese.

Table 6 shows the comparison results between our model and the state-of-the-art methods [11, 34]. MaxEnt [11] was a pipeline model, which employed human-designed lexical and syntactic features. Rich-C was developed by [34] and incorporated Chinese-specific features to improve Chinese event detection. Our method outperformed the other methods based on human designed features for the event trigger identification and achieved comparable F-score for the event classification.

4.5 Spanish extraction

Table 7 presents the performance of our method on the Spanish ERE corpus. The results showed that the HNN approach performed better than LSTM and Bi-LSTM. This finding indicated that our proposed model could achieve the best performance in multiple languages compared with the other neural network

Table 6 Results on the Chinese event detection

Model	Trigger identification			Trigger classification		
	Precision	Recall	F-score	Precision	Recall	F-score
MaxEnt	50.0	**77.0**	60.6	47.5	73.1	57.6
Rich-C	62.2	71.9	66.7	58.9	**68.1**	**63.2**
HNN	**74.2**	63.1	**68.2**	**77.1**	53.1	63.0

Table 7 Results on the Spanish event detection

Model	Trigger identification			Trigger classification		
	Precision	Recall	F-score	Precision	Recall	F-score
LSTM	62.2	52.9	57.2	56.9	32.6	41.6
Bi-LSTM	76.2	63.1	68.7	61.5	42.2	50.1
HNN	**81.4**	**65.2**	**71.6**	**66.3**	**47.8**	**55.5**

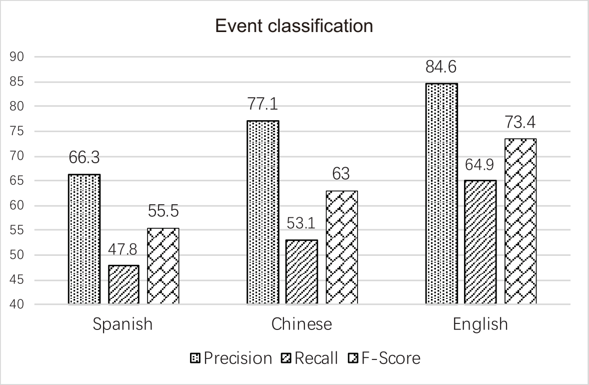

Figure 5 Comparison of the three languages.

methods. We did not compare our system with other systems [40], because they reported the results on a non-standard data set.

4.6 Comparison of the three languages

We take our method (HNN) as an example and conduct a horizontal comparison on English, Chinese, and Spanish.

Figure 5 shows the experimental results of HNN on the three languages. The model achieved better precision, recall and F-score on English than on Chinese and Spanish. On the Chinese corpus, the model needed a Chinese word segmentation model, which might induce some noise. Meanwhile, on the English corpus, the words were naturally separated by space. For Spanish, we believe that the low performance was due to the lack of training data. The performance of a machine learner depends on the feature representation and the size of training data. Our method introduced a method of learning a language-independent feature representation. However, for Spanish, the training data only includes 93 documents, which was much less than those for English (529) and Chinese (513).

5 Conclusion

We introduced a language-independent neural network model that incorporates both Bi-LSTM and CNN to capture sequence and structure semantic information from specific contexts for event detection. Compared with the traditional event detection methods, our approach does not rely on any linguistic resources and thus can be easily applied to any language. Moreover, our model can be effectively trained end-to-end with supervised event trigger identification and classification objects. We conducted experiments using various languages (i.e., English, Chinese and Spanish). The empirical results showed that our approach achieved a state-of-the-art performance in English and competitive results in Chinese. We also found that the Bi-LSTM was powerful for trigger extraction, specifically in capturing the preceding and following contexts in a long distance. As future work, we plan to incorporate discourse information into our neural network, which might help our system model the event structure better. This information has proven to be successful in sentiment classification [17].

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61632011, 61772156, 61702137).

References

1 Jurafsky D, Martin J H. Speech & Language Processing. London: Pearson Education India, 2000

2 Manning C D. Foundations of Statistical Natural Language Processing. Cambridge: MIT Press, 1999

3 Gao Y, Zhang H W, Zhao X B, et al. Event classification in microblogs via social tracking. ACM Trans Intel Syst Technol, 2017, 8: 35

4 Zhao S C, Gao Y, Ding G G, et al. Real-time multimedia social event detection in microblog. IEEE Trans Cybern, 2017. doi: 10.1109/TCYB.2017.2762344

5 Nguyen T H, Grishman R. Event detection and domain adaptation with convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, Beijing, 2015. 365–371

6 Peng H R, Song Y Q, Roth D. Event detection and co-reference with minimal supervision. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, 2016. 392–402

7 Wang Z Q, Zhang Y. A neural model for joint event detection and summarization. In: Proceedings of Conference on Empirical Methods in Natural Language Processing, Copenhagen, 2017

8 Hong Y, Zhang J F, Ma B, et al. Using cross-entity inference to improve event extraction. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, 2011

9 Ji H, Grishman R. Refining event extraction through cross-document inference. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL) with the Human Language Technology Conference, Columbus, 2008. 254–262

10 Li J W, Luong M, Jurafsky D. A hierarchical neural autoencoder for paragraphs and documents. 2015. ArXiv:1506.01057

11 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, 2013. 73–82

12 Liao S 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, 2010. 789–797

13 Dzmitry B, Kyunghyun C, Yoshua B. Neural machine translation by jointly learning to align and translate. 2014. ArXiv:1409.0473

14 Feng X C, Tang D Y, Qin B, et al. English-Chinese Knowledge base Translation with Neural Network. In: Proceedins of the 26th International Conference on Computational Linguistics: Technical Papers, Osaka, 2016. 2935–2944

15 Feng X C, Guo J, Qin B, et al. Effective deep memory networks for distant supervised relation extraction. In: Proceeding of the 26th International Joint Conference on Artificial Intelligence, Melbourne, 2017. 4002–4008

16 Zeng D J, Liu K, Lai S W, et al. Relation classification via convolutional deep neural network. In: Proceedings of the 25th International Conference on Computational Linguistics, Dublin, 2014. 2335–2344

17 Tang D Y, Qin B, Liu T. Document modeling with gated recurrent neural network for sentiment classification. In: Proceedings of Conference on Empirical Methods in Natural Language Processing, Lisbon, 2015. 1422–1432

18 Harris Z S. Distributional structure. Word, 1954, 10: 146–162

19 Feng X C, Huang L F, Tang D Y, et al. A language-independent neural network for event detection. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, 2016. 66–71

20 Chen Y B, Xu L H, Liu K, et al. 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, 2015. 167–176

21 David A. The stages of event extraction. In: Proceedings of the Workshop on Annotating and Reasoning about Time and Events, Sydney, 2006

22 Li Q, Ji H. Incremental joint extraction of entity mentions and relations. In: Proceedings of the Association for Computational Linguistics, Baltimore, 2014. 402–412

23 McClosky D, Surdeanu M, Manning C D. Event extraction as dependency parsing. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, 2011. 1626–1635

24 Goyal K, Jauhar S K, Li H Y, et al. A structured distributional semantic model for event co-reference. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, 2013. 467–473

25 Li J W, Jurafsky D, Hovy E. When are tree structures necessary for deep learning of representations? 2015. ArXiv:1503.00185

26 Graves A. Supervised Sequence Labelling with Recurrent Neural Networks. Berlin: Springer, 2012

27 Cao K, Li X, Fan M, et al. Improving event detection with active learning. In: Proceedings of Recent Advances in

Natural Language Processing, Hissar, 2015. 72–77

28 Baroni M, Dinu G, Georgiana K. Dont count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, 2014. 238–247

29 Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality. In: Proceedings of the 27th Annual Conference on Neural Information Processing Systems, Lake Tahoe, 2013

30 Hochreiter S, Schmidhuber J. LSTM can solve hard long time lag problems. In: Proceedings of Conference on Neural Information Processing Systems, Denver, 1997. 473–479

31 Zaremba W, Sutskever I, Vinyals O. Recurrent neural network regularization. 2014. ArXiv:1409.2329

32 Liu Y, Wei F R, Li S J, et al. A dependency-based neural network for relation classification. 2015. ArXiv:1507.04646

33 Zeiler M D. ADADELTA: an adaptive learning rate method. 2012. ArXiv:1212.5701

34 Chen C, Ng V. Joint modeling for chinese event extraction with rich linguistic features. Citeseer, 2012, 290: 529–544

35 Chen Z, Ji H. Language specific issue and feature exploration in Chinese event extraction. In: Proceedings of Annual Conference of the North American Chapter of the Association for Computational Linguistics, Boulder, 2009. 209–212

36 Liu S L, Liu K, He S Z, et al. A probabilistic soft logic based approach to exploiting latent and global information in event classification. In: Proceedings of the 30th AAAI Conference on Artificial Intelligence, Phoenix, 2016. 2993–2999

37 Liu S L, Chen Y B, He S Z, et al. Leveraging framenet to improve automatic event detection. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Berlin, 2016. 2134–2143

38 Liu S L, Chen Y B, Liu K, et al. Exploiting argument information to improve event detection via supervised attention mechanisms. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, 2017. 1789–1798

39 Liu T, Che W X, Li Z H. Language technology platform. J Chinese Inf Proc, 2011, 25: 53–62

40 Tanev H, Zavarella V, Linge J, et al. Exploiting machine learning techniques to build an event extraction system for portuguese and spanish. Linguamática, 2009, 1: 55–66