Exploiting Document Level Information to Improve Event Detection via Recurrent Neural Networks

Shaoyang Duan[1,2] and **Ruifang He**[1,2] and **Wenli Zhao**[1,2]
[1]School of Computer Science and Technology, Tianjin University, Tianjin, China
[2]Tianjin Key Laboratory of Cognitive Computing and Applications, Tianjin, China
{syduan,rfhe}@tju.edu.cn, wlzhao@gmail.com

Abstract

This paper tackles the task of event detection, which involves identifying and categorizing events. The previous work mainly exists two problems: (1) the traditional feature-based methods apply cross-sentence information, yet need taking a large amount of human effort to design complicated feature sets and inference rules; (2) the representation-based methods though overcome the problem of manually extracting features, while just depend on local sentence representation. Considering local sentence context is insufficient to resolve ambiguities in identifying particular event types, therefore, we propose a novel document level Recurrent Neural Networks (DLRNN) model, which can automatically extract cross-sentence clues to improve sentence level event detection without designing complex reasoning rules. Experiment results show that our approach outperforms other state-of-the-art methods on ACE 2005 dataset neither the external knowledge base nor the event arguments are used explicitly.

1 Introduction

Event detection is a crucial subtask of event extraction, which aims to extract event triggers (most often a single verb or noun) and classify them into specific types in text. For instance, according to the ACE 2005 annotation guideline[1], in the sentence "central command says **troops** were involved in a **gun battle yesterday**", an event detection system should be able to detect an **Attack** event with the trigger word "battle". However, this task is very challenging, as the same event might appear with various trigger words and a trigger expression might evoke different event types in different context.

Most of the existing methods either employed feature-based models with cross-sentence level information (Ji and Grishman, 2008)(Liao and Grishman, 2010)(Hong et al., 2011)(Huang and Riloff, 2012) or followed representation-based architectures with sentence level context (Chen et al., 2015)(Nguyen and Grishman, 2015)(Liu et al., 2016)(Nguyen and Grishman, 2016)(Nguyen et al., 2016)(Liu et al., 2017)(Chen et al., 2017). Both models have some inherent flaws: (1) feature-based approaches not only need to elaborately design rich features and often suffer error propagation from the existing natural language processing tools (i.e part of speech tags and dependency), but also the cross-sentence clues are embodied by devising complex inference rules, which is difficult to cover all the semantic laws; (2) though representation-based models can effectively alleviate the problem of manually extract features, local sentence context information may be insufficient for event detection models or even humans to classify events from isolated sentences. For example, consider the following sentences from ACE2005 dataset:

S1: Saba hasn't delivered yet.[2]

S2: I knew it was time to leave.[3]

It is very difficult to identify S1 as a Be-Born event with the trigger "delivered", which means that a person entity is given birth to. Similarly, we have low confidence to tag "leave" as a trigger for End-Position event in the S2, which means that a person entity stops working for an organization. However, the wider context that "She wants to cal-

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

[2]Selected from the file "CNN_IP_20030414.1600.04".

[3]Selected from the file "CNN_CF_20030303.1900.05".

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 352–361,
Taipei, Taiwan, November 27 – December 1, 2017

l her pregnant daughter Saba in Sweden to see if she has delivered." would give us more confidence to tag "delivered" as a Be-Born event in the S1. It is easy to identify the "leave" as a trigger for End-Position event in the S2, if we know the previous information that " this is when you were in the Senate – less and less information was new, fewer and fewer arguments were fresh, and the repetitiveness of the old arguments became tiresome. I was becoming almost as cynical as my constituents".

In fact, each document often has a main content in ACE 2005 English corpus. For example, if the content of a document is about terrorist attack, the document is more likely to contain Injure events, Die events, Attack events, and is unlikely to describe Be-Born events. In other words, there is a strong association between events appearing in a document. In addition, event types contained in documents with the related topics are also consistent. Therefore how to use intra and inter document information becomes particularly important. Although there have been already some work to capture the clues beyond sentence to improve sentence level event detection (Ji and Grishman, 2008)(Liao and Grishman, 2010)(Hong et al., 2011), they still exist the following disadvantages: (1) inherent defects in feature-based models; (2) document level information was used by a large number of inference rules, this is not only complicated and time-consuming, but also difficult to cover all of the semantic laws.

In this paper, we propose a document level recurrent neural networks (**DLRNN**) model for event detection to solve the above problems. Firstly, to capture lexical-level clues and minimize the dependence on supervised tools and resources for features, we introduce a distributed word representation model (Mikolov et al., 2013a), which has been proved very effective for event detection (Chen et al., 2015)(Nguyen and Grishman, 2015)(Nguyen and Grishman, 2016). Secondly, we employ bidirectional recurrent networks to encode sentence level clues, which can effectively reserve the history clues and the following information of the current word. Thirdly, to capture document level and cross-document level clues without complicated inference rules. We introduce a document representation, which uses a distributed vector to represent a document and has been showed to be able to get better performance on text classification and sentiment analysis tasks (Le and Mikolov, 2014). Finally, we use BILOU labeling method to solve the problem that a trigger contains multiple words.

In summary, our main contributions are as follows: (1) we prove the importance of document level information for event detection. (2) to capture document level clues, we devise a document level Recurrent Neural Networks (DLRNN) model for event detection, which can automatically learn features beyond sentence. (3) moreover, to solve the problem that a trigger word contains multiple words, we introduce BILOU labeling method. (4) finally, we improve the performance and achieve the best performance on ACE 2005 dataset neither the external knowledge base nor the event arguments are used explicitly.

2 Task Description

This paper focuses on addressing event detection task, which is a crucial subtask of event extraction. According to Automatic Context Extraction (ACE) evaluation[4], which annotates 8 types and 33 subtypes for event mention. An event is defined as a specific occurrence involving one or more participants. Firstly, we introduce some ACE terminologies to facilitate the understanding of event extraction task:

Entity: an object or a set of objects in one of the semantic categories of interests.

Entity mention: a reference to an entity (typically, a noun phrase).

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

Event arguments: the mentions that are involved in an event (participants).

Event mention: a phrase or sentence within which an event is described, including the trigger and arguments.

Given an English document, an event extraction system should identify event triggers and their corresponding arguments with specific subtypes or the roles for each sentence, but an event detection system only needs to identify event trigger and their subtype. For instance, for the sentence "central command says **troops** were involved in a **gun battle yesterday**", an event extraction system is supposed to detect the word "battle" as the event trigger of **Attack** event and identify the word

[4] https://project.ldc.upenn.edu/ace

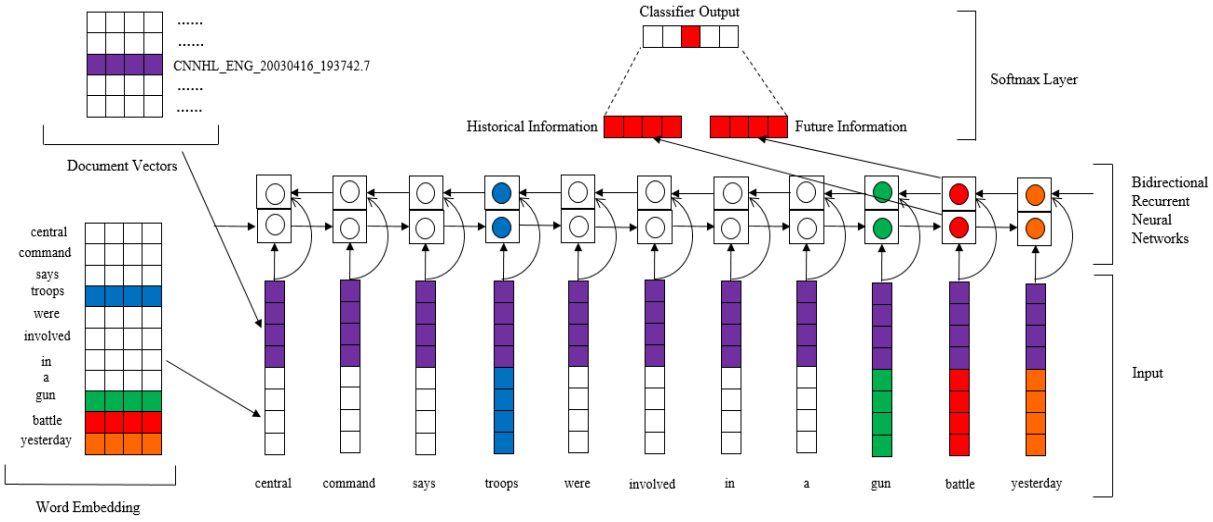

Figure 1: An illustration of our DLRNN model for detecting the trigger word "battle" in the input sentence "central command says troops were involved in a gun battle yesterday".

"troops", "gun" and "yesterday" as event argument whose roles are **Attacker**, **Instrument** and **Time-Within**. However, for an event detection system, identifying the word "troops", "gun" and "yesterday" as event argument whose roles are **Attacker**, **Instrument** and **Time-Within** is not involved. Following previous work, we treat these simply as 33 separate event types and ignore the hierarchical structure among them.

3 Model

In this section, we give the details for the DLRNN model (show in Figure 1). First of all, we formalize the event detection task as a multi-classes classification problem following previous work. More precisely, for each word in a sentence, our goal is to classify them into one of 34 classes (33 trigger types and None class).

Our DLRNN model primarily includes four parts: (i) word embedding, which contains lexical information for each word and is trained from external corpus in an unsupervised manner; (ii) document vector, which reveals the topic of a document is trained in an unsupervised mechanism; (iii) bidirectional recurrent neural networks encoding, which can learn the historical and future abstractive representation of a candidate trigger; (iv) trigger prediction, which calculates a confidence score for each event subtype candidate.

3.1 Word Embedding

The representation of the words as continuous vectors (word embedding) are proved more powerful than discrete representation (Bengio et al., 2003)(Mikolov et al., 2013b). Word embedding not only addresses the problem of dimension disaster, but also makes the word vector contain richer semantic information. The closer the vector space, the closer the semantic. In addition, word embedding can automatically learn lexical-level clues in the process of pre-training. Not only does not require human ingenuity, but also effectively alleviates the error propagation brought by other NLP lexical analysis toolkits. Recent work has demonstrated that using word embedding can enhance the robustness of event detection model (Nguyen and Grishman, 2015)(Chen et al., 2015)(Nguyen and Grishman, 2016).

In this paper, we pre-trained word embedding via skip-gram model (Mikolov et al., 2013b) and New York Times corpus[5]. Given a sequence of training words w_1,w_2,w_3,...,w_T, the skip-gram model trains the embedding by maximizing the average log probability:

$$\frac{1}{T}\sum_{t=k}^{T-k}\log p(w_{t-k},...,w_{t+k}|w_t) \quad (1)$$

where w_{t-k},...,w_{t+k} is the context of w_t and the window size is k, usually it is expressed by the concatenation or sum of all word vectors in the

[5] https://catalog.ldc.upenn.edu/LDC2008T19

context; $p(w_{t-k},...,w_{t+k}|w_t)$ is calculated via softmax. There, we have:

$$p(w_{t-k}, ..., w_{t+k}|w_t) = \frac{e^{y_{w_{t-k},...,w_{t+k}}}}{\sum_i e^{y_i}} \quad (2)$$

where each of y_i is un-normalized log-probability for each output context of the word *i*, computed as

$$y = b + Uw_t \quad (3)$$

where *U*,*b* are the sofmax parameters.

3.2 Document Vector

In order to illustrate the importance of the document vector for event detection in terms of disambiguity. we propose three hypotheses from intra and inter document context perspectives.

H1: As we all know, the same word in different context often has different meanings. For instance, the word "delivered" in S1 can mean that someone is born or bring something to a destination, when given different context.

H2: Events in a document exist consistency. For example, Die events and Marry events almost never appear in the same document, but Die events often occur with Attack events and Injure events in a document.

H3: The event types in documents with the related topics exist consistency. For instance, if the document that describing a financial crisis contains End-Position events and End-Org events, and then another document related to the financial crisis topic is more likely to happen End-Position events and End-Org events.

Based on the above three assumptions, we introduced an advanced document representation method. Documents are represented by the distributed vector like word embedding, which not only contains the main content of a document, but also the more relevant documents, the closer the document vector. For all the words in a document, the document vector is shared and is concatenated with word embedding, serving as the semantic representation of a word, as shown in Figure 1. Concatenating the document vector to word embedding has the following advantages: (i) a word is no longer represented by a unique word vector, but expressed by different vector in different documents. This can help event detection model to disambiguate event type; (ii) the consistency of events in a document is guaranteed. Since all the words in a document share a document vector, which passes the identified event subtype information. For example, if some candidate triggers containing a particular document vector are mostly identified as Attack events, Die events, and Injuries events, and then the other candidate triggers that containing the document vector will be less likely to be identified as Marry events or Be-Born events. (iii) documents with related topic almost contain the same event types. Due to the fact that the more relevant topic of the documents, the closer document vectors, the model will be given high confidence to label candidate trigger in a document as the types that appearing in the relevant topic of the documents.

In this paper, we trained document vectors by using the PV-DM model (Le and Mikolov, 2014), which is very similar to the CBOW model that is another word embedding model (Mikolov et al., 2013a). Unlike the skip-gram model, given a document that contains training words w_1,w_2,w_3,...,w_T, document vector is trained by maximizing the average log probability:

$$\frac{1}{T}\sum_{t=k}^{T-k} \log p(w_t|w_{t-k}, ..., w_{t+k}, doc) \quad (4)$$

where w_{t-k},...,w_{t+k} is the context of w_t and the window size is *k*; *doc* is the document vector containing the training words, which can be randomly initialized to a fixed dimension of vector like word embedding, see (Le and Mikolov, 2014) for details.

3.3 Bidirectional Recurrent Neural Networks Encoding

Recurrent neural networks (RNN) has been shown to perform considerably better than standard feedforward architecture (Hammerton, 2003)(Sutskever et al., 2011)(Liu et al., 2014)(Sundermeyer et al., 2014). In this paper, we used RNN to encode word level information and document level clues. In the following, we describe our encoding model in detail.

The traditional RNN predicts the current tag with the consideration of the current input and history information before the current input. It loses the following information after the current input. In order to address this problem, we ran two RNNs, one of the RNNs is responsible for encoding the history information, and the other one is responsible for encoding the future information. In addition, the standard RNN often suf-

fers from gradient vanishing or gradient exploding problems during training via backpropagation (Bengio et al., 1994). To remedy this problem, we used long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) that is a variant of RNN to replace the standard RNN.

Formally, given candidate input sequence $X = \{x_1, x_2, ..., x_n\}$. We run LSTM1 to get the hidden representation $\{h_{f_1}, h_{f_2}, ..., h_{f_n}\}$ and run LSTM2 to get the hidden representation $\{h_{b_1}, h_{b_2}, ..., h_{b_n}\}$. Each h_{f_i} and h_{b_i} are computed by:

$$h_{f_i} = \overrightarrow{LSTM}(x_i, h_{f_{i-1}}) \quad (5)$$

$$h_{b_i} = \overleftarrow{LSTM}(x_i, h_{b_{i+1}}) \quad (6)$$

where x_i is the concatenation of the word embedding of token i in candidate sentence and document vector that contains token i, as shown in Figure 1; $h_{f_{i-1}}$ contains the historical information before x_i; $h_{b_{i+1}}$ contains the future clues after x_i. Eventually, we obtain the context information over the whole sentence $\{h_1, h_2, ..., h_n\}$ with a greater focus on the position i by concentrating $\{h_{f_1}, h_{f_2}, ..., h_{f_n}\}$ and $\{h_{b_1}, h_{b_2}, ..., h_{b_n}\}$, where $h_i = [h_{f_i}, h_{b_i}]$.

3.4 Trigger Prediction

In the actual situation, due to the fact that a trigger may contain multiple words, we introduce the BILOU labeling method, which has been shown to be able to achieve better results than BIO labeling in entity recognition tasks (Gupta et al., 2016). In the BILOU labeling method, B represents the beginning of a trigger word, I indicates that the word is inside a trigger word, L represents that the word is the last word for a trigger word, O signifies that the word is not a trigger word, U denotes the trigger word contains unique word.

After bidirectional long short-term memory (BiLSTM) encoding, we get the global abstract representation h_i that encapsulates all context of the input sentence (see in section 3.3). And then, we feed h_i into a feed-forward neural network with a softmax layer (as shown in Figure 1). In the end, we get a 34 dimensions vector [6], where the k-th term o_k is the probability value for classifying x_i to the k-th event type.

Given all of our (suppose T) training samples $(x^{(i)}; y^{(i)})$, we can then define the loss function as the average negative log-likelihood:

$$J(\theta) = -\frac{1}{T}\sum_{i=1}^{T} \log p(y^{(i)}|x^{(i)}, \theta) \quad (7)$$

In order to compute the network parameter θ, we minimize the average negative log-likelihood $J(\theta)$ via stochastic gradient descent (SGD) over shuffled mimi-batches with Adam update rule (Kingma and Ba, 2014) and the dropout regularization (Zaremba et al., 2014).

4 Experiments

4.1 Dataset and Experimental Setup

We evaluate our DLRNN model on the ACE2005 English corpus. For fair comparisons, the same with (Ji and Grishman, 2008)(Liao and Grishman, 2010)(Hong et al., 2011)(Liu et al., 2017)(Chen et al., 2017), we select the same 40 newswire documents as the test set, the same 30 documents from different genres as development set and the remaining 529 documents are used as training set. Furthermore, we also follow the criteria of the previous work (Ji and Grishman, 2008)(Liao and Grishman, 2010)(Hong et al., 2011)(Li et al., 2013)(Liu et al., 2017)(Chen et al., 2017) to judge the correctness of the predicted event mentions and use *Precision* (*P*), *Recall* (*R*), *F-measure* (F_1) as the evaluation metrics.

We set the the dimension of word embedding to 200, the dimension of document vectors to 100, the size of hidden layer to 300, the size of mini-batch to 100, the dropout rate to 0.5, the learning rate to 0.002. All of the above hyper-parameter are adjusted on the development set.

4.2 Baseline Methods

In order to validate our DLRNN model, we choose the following models as our baselines, which are the state-of-the-art methods in sentence level and cross-sentence level event detection models.

Cross-Sentence Level Baselines:

1) **Cross-Document Inference:** It is the feature-based model proposed by (Ji and Grishman, 2008), which is the first time to use document information to assist in sentence level event detection. They employed document theme clustering and designed a lot of reasoning rules to ensure event consistency within the scope of the document and clustering.

2) **Cross-Event Inference:** This is the feature-based method proposed by (Liao and Grishman,

[6] As a result of the BILOU tag, the actual dimension is more than 34 dimensions, but it is described as 34 dimensions for ease of understanding.

Methods	P	R	F_1
Ji's cross-document†	60.2	76.4	67.3
Liao's cross-event†	68.7	68.9	68.8
Hong's cross-entity†	72.9	64.3	68.3
Li's joint model	73.7	62.3	67.5
Ngyuen's JRNN	66.0	73.0	69.3
Chen's DMCNN	75.6	63.6	69.1
Chen's DMCNN+DS	75.7	66.0	70.5
Liu's ANN	79.5	60.7	68.8
Liu's ANN+Attention	78.0	66.3	71.7
Our DLRNN†	77.2	64.9	70.5

Table 1: Overall Performance on Blind Test Data. "†" designates the model that employs the evidences beyond sentence level. The boldface indicates that the model is representation-based model.

2010), which not only used the consistency information of the same type events in a document, but also explored the clues from the co-occurrence of different event types in the same document.

3) **Cross-Entity Inference:** It is the feature-based approach proposed by (Hong et al., 2011), which used the entity co-occurrence as a key feature to predict event mention.

Sentence Level Baselines:

4) **Joint Model:** It is the feature-based model proposed by (Li et al., 2013), which exploited argument information implicitly and captured the dependencies between two triggers within the same sentence.

5) **Joint RNN:** It is the representation-based method proposed by (Gupta et al., 2016), which exploited the inter-dependence of event trigger and event argument.

6) **DMCNN + Distant Supervision:** It is the representation-based method proposed by (Chen et al., 2017), which used the Freebase and FrameNet to extend the training corpus through distant supervision.

7) **ANN + Attention:** It is the representation-based approach proposed by (Liu et al., 2017), which exploited argument information explicitly for event detection via supervised attention mechanisms.

4.3 Performance Comparison

Table 1 are the comparisons of experimental results of our method with the baseline methods on the same blind test dataset. Seen from Table 1, we make the following observations:

1) The performance of representation-based models is better than that of feature-based models. It indicates the artificially well-designed features are not sufficient for event detection, and automatically extracting features based on neural networks can capture richer semantic clues. In detail, the F_1 score of our DLRNN model is higher than state-of-the-arts feature-based model (Liao's cross-event) by 1.7%; the other three representation-based models achieved better experimental results than that of Liao's cross-event model, which gain 0.5%,1.7% and 2.9% improvement, respectively.

2) The feature-based models that using cross-sentence information is more advantageous than the sentence level model. More accurately, in the cross-sentence models, only the performance of the Ji's cross-document method is slightly lower than Li's joint model (-0.2%), but the performance of the remaining models is better than Li's joint model (an improvement of 0.8% and 1.3% in F_1 score). It proves the clues beyond sentence are very important for event detection.

3) Our DLRNN method outperforms all cross-sentence level feature-based event detection models. In detail, DLRNN gains 3.2% improvement on F_1 score than Ji's cross-document, gains 1.7% improvement on F_1 score than Liao's cross-event and gains 2.2% improvement on F_1 score than Hong's cross-entity. The reasons are as follows: on the one hand, artificially constructed inference rules are difficult to cover all semantic laws; on the other hand, our DLRNN is better able to capture document level clues (including intra and inter-document context).

4) In spite that the performance of our DLRNN model does not improve the F_1 score compared with Chen's DMCNN+DS model, even the performance is not as good as Liu's ANN+Attention model. However, our method neither explicitly utilized event argument information, nor extended training data through using world knowledge (Freebase) and linguistic knowledge (FrameNet). If removed the event argument information and the knowledge base (Chen's DMCNN and Liu's ANN), the F_1 score of our DLRNN model is superior to the DMCNN and ANN methods, which are -1.4% and -1.7% lower, respectively. This not only illustrates that document level clues are very effective for the representation-based model, but

also prove that the effectiveness of the proposed method.

4.4 The Effectiveness of Document Vector

In order to verify the effectiveness of the document vector trained by PV-DM model for event detection, we design four experiments as baselines for comparison with our DLRNN (as shown in Table 2): BiLSTM, BiLSTM+TF-IDF, BiLSTM+AVE and BiLSTM+LDA.

1) **BiLSTM:** BiLSTM is similar to DLRNN except for removing the document vectors, only uses word embedding as the input of model.

2) **BiLSTM+TF-IDF:** Selected the word vector of the most important word for each document as the document vector for the document.

3) **BiLSTM+AVE:** The document vector is obtained by averaging the vector of each word in the document.

4) **BiLSTM+LDA:** The probability that each document corresponds to each topic is the document vector of the document.

5) **DLRNN:** DLRNN model uses the document vector, which is trained by PV-DM approach instead of averaging the word vector in the document[7].

Methods	P	R	F_1
BiLSTM	76.1	63.5	69.3
BiLSTM+TF-IDF†	74.2	64.6	69.1
BiLSTM+AVE†	75.4	64.7	69.6
BiLSTM+LDA†	74.3	66.1	70.0
DLRNN†	77.2	64.9	70.5

Table 2: Overall Performance on Blind Test Data. "†" designates the model that employs the evidences beyond sentence level."+TF-IDF" represents the document vector was obtained by TF-IDF."+LDA" represents the document vector was obtained by LDA."+AVE" represents the document vector was obtained by averaging the word vector in the document.

Seen from Table 2, we get the following observations: 1) in addition to BiLSTM+TF-IDF, the event detection models with the document vector can achieve better experimental results. In detail, BiLSTM+AVE, BiLSTM+LDA and DLRNN are 0.3%, 0.7% as well as 1.2% better than BiLSTM on F_1 score, respectively. This indicates that document level clues can contribute to sentence level event detection model. 2) compared to BiLSTM+TF-IDF, BiLSTM+AVE, BiLSTM+LDA, DLRNN gains 1.4%, 0.9%, 0.5% on F_1 score. This illustrates PV-DM model is able to capture richer semantic information.

In addition, in order to illustrate the documents that their vectors are similar contain the consistent event types. We visualize the document vectors. In detail, we randomly selected a document containing the events from ACE2005 English corpus, and found a document that is most similar to the selected document by calculating the cosine similarity of document vectors. Finally, we systematically compared the events contained in the two documents.

We randomly selected the document CNNHL_ENG_20030624_133331.33 as a source document, and found the document CNNHL_ENG_20030624_230338.34 is most similar to it by computing the cosine similarity of document vectors[8]. Seen from Figure 2, we observe that the two documents contain the same event types, except that the document CNNHL_ENG_20030624_133331.33 does not contain Attack event. Event type overlapping rate is up to 80%. This proves that there is correlation between the documents of similar document vectors.

Selected document	Most similar document
CNNHL_ENG_20030624_133331.33	CNNHL_ENG_20030624_230338.34
SUBTYPE="Transport"	SUBTYPE="Transport"
SUBTYPE="Transport"	SUBTYPE="Transport"
SUBTYPE="Transport"	SUBTYPE="Transport"
SUBTYPE="Transport"	SUBTYPE="Transport"
SUBTYPE="Die"	SUBTYPE="Die"
SUBTYPE="Die"	SUBTYPE="Attack"
SUBTYPE="Die"	N/A
SUBTYPE="Charge-Indict"	SUBTYPE="Charge-Indict"
SUBTYPE="Phone-Write"	SUBTYPE="Phone-Write"
SUBTYPE="Injure"	SUBTYPE="Injure"

Figure 2: The comparison of event types on the most similar documents.

4.5 The Event Consistency in a Document

Seen from the Table 3, we observe that the Injure event often appears along with the Attack events, the Die events, and the Transport events

[7] we clean the documents up by converting everything to lower case and removing punctuation and the stop words.

[8] The cosine similarity is 0.992

Event Subtype	Conditional Probability
Attack	0.4399
Die	0.2018
Transport	0.1555
Meet	0.0287
Demonstrate	0.0221
.....	.....
Nominate	0.0
Elect	0.0

Table 3: The ranking probability of events co-occurrence with Injure events.

in the same document. The total probability of the above three types of events concurrence with Injure event is about 0.797. Furthermore, the Nominate events, the Elect events, and so on, have never been appeared in the same document containing the Injure events. This indicates that only certain types of events can occur in the same document, therefore the introduction of the document vector will help to predict event types in a document. Thus, the inter-document information reflected in document vector is useful to event detection.

4.6 The Effectiveness of BILOU Labeling

According to statistics, ACE2005 English corpus contains 235 trigger words, which are composed of multiple words, about 4.39% of the total trigger words. It is not appropriate to treat identifying the triggers that contains multiple words as a word classification task, because most of the triggers of multiple words contain prepositions. However, the prepositions in such triggers do not trigger event independently. Therefore, using BILOU encoding helps to treat the multiple words trigger as a whole. Table 3 demonstrates the effectiveness of the BILOU encoding (an improvement of 0.2% on F_1 score).

Methods	P	R	F_1
DLRNN-BILOU	78.8	63.5	70.3
DLRNN	77.2	64.9	70.5

Table 4: Overall Performance on Blind Test Data. "-BILOU" indicates that the model has not the BILOU labeling.

5 Related Work

Event detection is a challenging task in the field of natural language processing, which has attracted more and more researchers' attention in recent years. The current event detection models can roughly be divided into: (1) the sentence level event detection models and (2) the cross-sentence level event detection models.

(1) The sentence level event detection models: they are designed to use the sentence information for event classification. According to the differences in how to use sentence information, they can be divided into two categories: the feature-based models and the representation-based models. The early event detection models are almost all feature-based models, which transformed lexical features, syntactic features and semantic features into one-hot vectors by other natural language processing toolkits, and then sended these well-designed features into the classifiers (eg: structure perceptron or support vector machine) and eventually completed the event classification (Ahn, 2006)(Li et al., 2013). With the success of deep learning in entity identification and relationship classification (Collobert and Weston, 2008)(Zeng et al., 2014), many event detection researchers turned to focus on the representation-based models. This kind of models do not need to extract the features manually. They used the distributed word vector as the input and encoded the word vector into low-dimensional abstractive representation by the neural network to complete event detection (Nguyen and Grishman, 2015)(Chen et al., 2015)(Nguyen et al., 2016)(Nguyen and Grishman, 2016)(Liu et al., 2016)(Liu et al., 2017)(Chen et al., 2017).

(2) The cross-sentence level event detection models: they aim to explore the clues beyond sentence to improve sentence level event detection. Remarkable researches are cross-document inference (Ji and Grishman, 2008), cross-event inference (Liao and Grishman, 2010), cross-entity inference (Hong et al., 2011) and modeling textual cohesion (Huang and Riloff, 2012). There mainly have two disadvantages: 1) The existing cross-sentence event detection models are feature-based models, which not only need to construct complex manual features and lack generalization ability; 2) utilizing the clues beyond sentence through designing complex and numerous reasoning rules, is not only complex, but also can not cover all semantic phenomenon. Different from the above methods, our approach makes the machine automatically learn the document level information by the representation based way to improve the per-

formance of event detection.

6 Conclusion

In this paper, we propose a novel model (DLRNN) to automatically extract cross-sentence level clues for event detection by concatenating word vector and document vector. Moreover, we use BILOU encoding to solve the problem that contains multiple words in a trigger word. In order to prove the effectiveness of the proposed method, we systematically conduct a series of experiments on ACE2005 dataset. Experimental results show that the proposed method is better than state-of-the-arts cross-sentence level feature-based models and the sentence level representation-based models without using argument information and external corpus, such as Freebase and FrameNet (Liu et al., 2017)(Chen et al., 2017), which demonstrates that intra and inter-document context is effective for event detection.

Acknowledgments

This work was supported by the National Science Foundation of China (No. 61472277), the National Key Basic Research and Development Program of China (973 Program, No. 2013CB329301).

References

David Ahn. 2006. The stages of event extraction. *In Proceedings of ACL,* pages 1–8.

Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research* 3(6):1137–1155.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *The Journal of IEEE transactions on neural networks* 5(2):157–166.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. *In Proceedings of ACL* .

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. *In Proceedings of IJCNLP,* pages 167–176.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing:deep neural networks with multitask learning. *In Proceedings of ICML,* pages 160–167.

Pankaj Gupta, Hinrich Schtze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. *In Proceedings of COLING,* pages 2537–2547.

James Hammerton. 2003. Named entity recognition with long short-term memory. *In Proceedings of NAACL,* pages 172–175.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *The Journal of Neural Computation,* 9(8):1735–1780.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. *In Proceedings of ACL,* pages 1127–1136.

Ruihong Huang and Ellen Riloff. 2012. Modeling textual cohesion for event extracion. *In Proceedings of AAAI,* pages 1664–1670.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. *In Proceedings of ACL,* pages 254–262.

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

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *In Proceedings of ICML,* pages 1188–1196.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. *In Proceedings of ACL,* pages 73–82.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. *In Proceedings of ACL,* pages 789–797.

Shujie Liu, Nan Yang, Mu Li, and Ming Zhou. 2014. A recursive recurrent neural network for statistical machine translation. *In Proceedings of ACL,* pages 1491–1500.

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016. Leveraging framenet to improve automatic event detection. *In Proceedings of ACL* pages 2134–2143.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. *In Proceedings of ACL* .

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *In Proceedings of Advances in Neural Information Processing Systems,* pages 3111–3119.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. *In Proceedings of NAACL,* pages 300–309.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolution neural networks. *In Proceedings of IJCNLP,* pages 365–371.

Thien Huu Nguyen and Ralph Grishman. 2016. Modeling skip-grams for event detection with convolution neural networks. *In Proceedings of EMNLP,* pages 886–891.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. 2014. Translation modeling with bidirectional recurrent neural networks. *In Proceedings of EMNLP,* pages 14–25.

Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. *In Proceedings of ICML,* pages 1017–1024.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329* .

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. *In Proceedings of COLING,* pages 2335–2344.