LSTM-Based End-to-End Framework for Biomedical Event Extraction

Xinyi Yu, Wenge Rong, Jingshuang Liu, Deyu Zhou, Yuanxin Ouyang, and Zhang Xiong,

Abstract—Biomedical event extraction plays an important role in the extraction of biological information from large-scale scientific publications. However, most state-of-the-art systems separate this task into several steps, which leads to cascading errors. In addition, it is complicated to generate features from syntactic and dependency analysis separately. Therefore, in this paper, we propose an end-to-end model based on long short-term memory (LSTM) to optimize biomedical event extraction. Experimental results demonstrate that our approach improves the performance of biomedical event extraction. We achieve average F1-scores of 59.68%, 58.23% and 57.39% on the BioNLP09, BioNLP11 and BioNLP13's Genia event datasets, respectively. The experimental study has shown our proposed model's potential in biomedical event extraction.

Index Terms—Biomedical event extraction, end-to-end, Bi-LSTM, Tree-LSTM

1 Introduction

For several years, biology and information communities have been working towards the goal of extracting complex information from large volumes of biological publications, especially information related to the behavior of bio-molecules containing event information in a structured form [1]. The structured descriptions of biomedical events consisting of "event category," "trigger," and "argument" [2]. Figure 1 presents a typical instance describing the phosphorylation and negative regulation behaviors between proteins "TRAF2" and "CD40." The trigger for the phosphorylation event is "phosphorylation" and the argument is "TRAF2." The word "inhibits" indicates that a negative regulation event is also included. These two events can be annotated as follows: 1) Event1: Phosphorylation, Trigger: phosphorylation, Theme1: TRF2; 2) Event2: Negative Regulation, Trigger: inhibits, Theme1: CD40, Cause1: Event1.

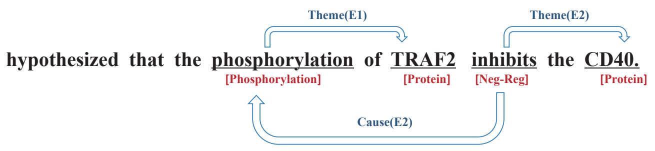

Fig. 1. An example of event extraction.

Recently, a lot of methods toward biology event extraction have been proposed and most of them are based on three steps: 1) event trigger detection, 2) argument detection, and 3) post-processing. Although this pipeline is easy to implement, it might cause cascading errors [3]. For example, if there is no trigger detected in the first step, the argument will never be detected and thus the entire event will be missed. To address this problem, Poon and Vanderwende [4] adopted a Markov logic network to create logically joint statements to extract triggers and arguments simultaneously. However, their Markov logic network method does not significantly outperform previous systems because the Markov logic network cannot make good use of a large number of features. Riedel and McCallum [5] proposed three combined models with a prediction-based passive-aggressive (PA) online learning algorithm to overcome this problem. The first model performs joint trigger and argument extraction. The second model captures correlations between events. The third model ensures consistency between arguments of the same event. Furthermore, to solve the problem of one-hot encoding being unable to represent rich semantic information, Li et al. [6] proposed a method that adopted dual decomposition and rich features by integrating word embedding to detect events jointly.

Although these methods have shown some potential, certain challenges still remain: 1) Cascading errors are not eliminated because these models still utilize at least three steps that separate trigger detection and argument detection into two individual models (without any common parameters), as such errors emerging from the trigger detection step will be passed to the argument detection step and eventually affect the final result because models for trigger detection and argument detection are trained separately with their own parameters. 2) Few models can capture useful features automatically because they are not good at handling sequence information. 3) Representing words purely with word embedding is not sufficient as semantic information and dependency information should also be included [7].

In light of these challenges, we tried to adopt end-to-end oriented approach to solve the challenges. Our motivation is inspired by the study of Bhattacharyya et. al [8], who proposed a novel approach for end-to-end relationship extraction by using a module based on a neural network. Similarly, Miwa et al. presented an end-to-end neural model that can capture both word sequence and dependency tree substruc-

- *X. Yu, W. Rong, J. Liu, Y. Ouyang and Z. Xiong are with State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China. They are also with School of Computer Science and Engineering, Beihang University, Beijing 100191, China. E-mail: {yuxinyi, w.rong, jingshuangl, xiongz}@buaa.edu.cn.*
- *D. Zhou is with School of Computer Science and Engineering, Southeast University, Nanjing 210096, China. E-mail: d.zhou@seu.edu.cn*

Manuscript received October 19, 2018; revised .

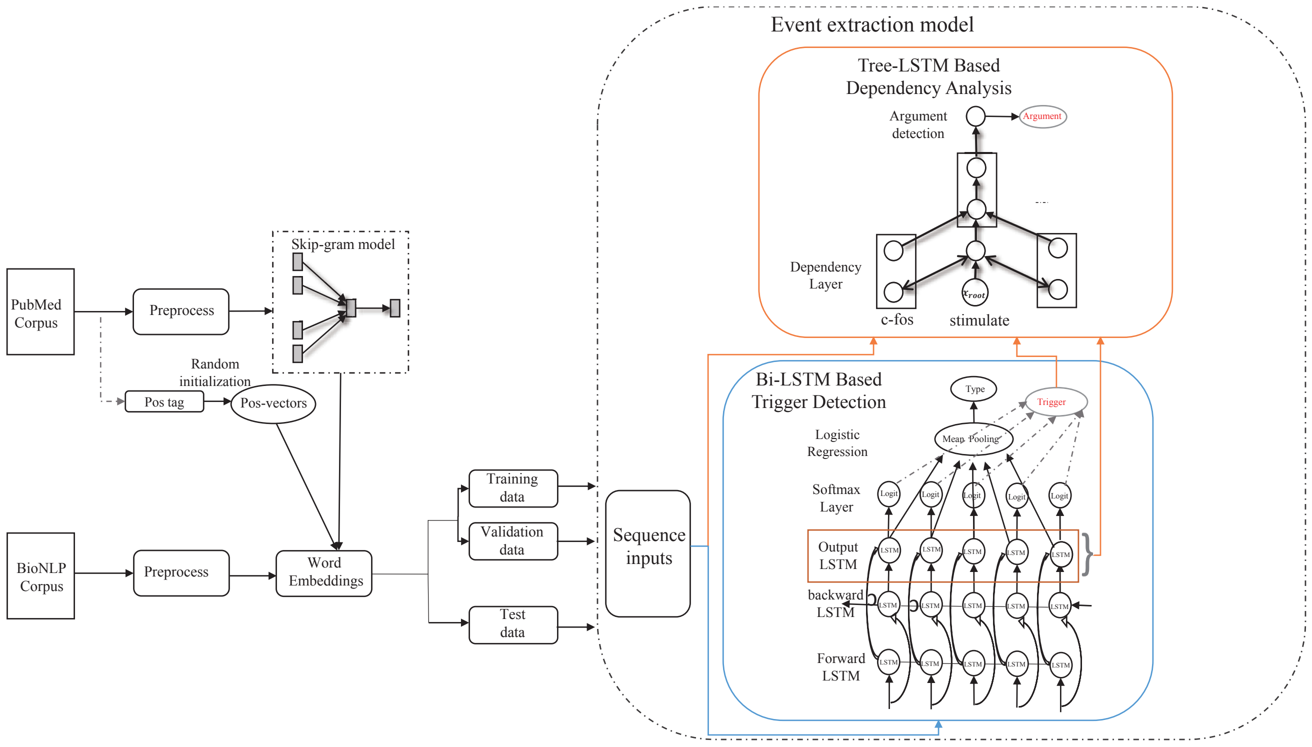

Fig. 2. Overall Framework: Consists of Bi-LSTM, dependency layer, and Tree-LSTM

ture information by stacking bidirectional tree-structured long short-term memory (LSTM) recurrent neural networks (RNNs) with bidirectional sequential LSTM-RNNs to extract entities and the relationships between them. This method outperformed a state-of-the-art feature-based system on end-to-end relationship extraction [9]. Based on this idea, Li et. al [10] introduced a new model called DET-BLSTM for event extraction. DET-BLSTM used a dynamic extended tree (DET) instead of the original sentences as the input, as well as constructed a bidirectional LSTM model to get the different information from forwards and backwards. However, DET-BLSTM was a single model and was applied to a relatively simple event extraction task of BioNLP'16 Shared Task on Bacteria Biotope. Thus, it is still a challenge to design a model which can achieve co-training between two sub-models.

In this paper, we present an end-to-end model based on bidirectional LSTM (Bi-LSTM) networks and tree LSTM (Tree-LSTM) networks. Bi-LSTM has been proven to be beneficial in other natural language processing tasks, such as multi-feature extraction [11] and named entity recognition [12]. Because of their superior ability to preserve sequence information over time, Bi-LSTM networks have achieved excellent results on a variety of sequence modeling tasks. Regarding Tree-LSTM, it is a generalization of LSTMs into tree-structured network topologies [13], which can simultaneously optimize both a composition function and parser in order to eliminate the need for external parse trees [14]. Therefore, we first represent words as a linear sequence using word embedding. We then build a Bi-LSTM model to detect triggers. Next, we employ a bidirectional Tree-LSTM to assist argument detection. Finally, in order to reduce the risks caused by cascading errors, we implement an end-to-end framework to train our model.

The main contributions of our work are as follows: 1) We employ Bi-LSTM to extract words in order to represent syntactic interpretations of sentence structure and preserve sequence information over time. 2) We employ Tree-LSTM to derive the long-term dependencies of text, instead of local features. 3) The proposed end-to-end model makes it possible to jointly perform trigger detection and argument detection with shared parameters. 4) We employ word embedding to better solve the semantic sparsity problem of short text compared to the one-hot representation.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. The proposed end-to-end model is detailed in Section 3. Section 4 shows the details of our experimental study. Section 5 concludes this paper and discusses future research directions.

2 Related Work

Extracting information from unstructured text is one of the most important goals in the biomedical event extraction domain. Biomedical event extraction can be regarded as the task of assigning labels to corresponding words and determining the relationships between target pairs. Typical examples of such processes have been analyzed in [1], where events were expressed using three different types of entities: T-entities, including event triggers (localization, binding, etc.), protein references (protein), and references to other entities (entity). Typically, the goal of extraction is to detect the types of triggers and relationships between triggers and arguments [39].

Traditionally, there are two types of approaches for this task in the community, i.e., rule-based approaches and machine learning based approaches [39]. Rule-based approaches generate a dictionary and patterns from annotated events and then apply the dictionary and patterns to extract events from input text [40]. Therefore, because of poor generalization ability resulting from stable rules, this approach typically has high precision, but low recall during prediction. Machine Learning(ML) approach is currently the most common approach to biomedical event extraction. It considers each task as a classification problem. The common pipeline for event extraction includes event trigger identification and event argument detection. ML approaches, especially those using SVMs, have achieved the best performance on previous BioNLP-ST Genia event extraction tasks [39]. For example, Björne et al.'s system adopted an SVM for event extraction and extracted a wide array of features based on dependency parsing graphs. Lever et al. (VERSE Team) extracted semantic features and adopted feature selection for SVM classification through the use of a feature-based approach [41]. Despite the advantages of ML approaches in terms of extracting rich features from text [39], these approaches still have some drawbacks. One-hot features are restricted by the issues of semantic gaps and dimensionality disasters. Additionally, selecting an optimal subset of features may require excessive experiments [10]. Furthermore, pipeline-based systems suffer from cascading errors.

To overcome these issues, deep neural networks have been considered as an effective method for event extraction tasks [42]. Mehryary et al. proposed a deep-learning-based approach for event extraction by combining several LSTM networks through syntactic dependency graphs [43]. Li et al. proposed a novel bidirectional LSTM-based RNN called the dynamic extended tree for bacteria biotope event extraction [10]

Although these methods have shown some promising results, challenges still exist. Cascading errors are not eliminated because these methods separate trigger detection and argument detection into two individual models without any common parameters shared. This means that the errors in any subtask will negatively affect subsequent subtasks [8]. Therefore, if we can create an end-to-end model to perform trigger detection and argument detection simultaneously, it will be much easier to eliminate cascading errors. Recently, joint-neural-network-based models have achieved superior performance compared to separate predictive models in event extraction tasks. For example, Li and Ji presented an incremental joint framework to simultaneously extract entity mentions and relationships using a structured perceptron combined with an efficient beam-search [44]. Similarly, Pawar et al. proposed an approach that combines the advantages of neural networks and Markov logic networks to jointly address all three subtasks of end-to-end relationship extraction [8]. Additionally, Miwa et al. stacked bidirectional tree-structured LSTM-RNNs on top of bidirectional sequential LSTM-RNNs to capture both word sequence and dependency tree substructure information. This allowed their model to jointly represent both entities and relationships using shared parameters in a single model [9].

3 Methodology

In this study, we developed an end-to-end method based on Bi-LSTM and Tree-LSTM to extract biomedical events. The framework of our method is illustrated in Fig. 2. The framework is composed of a Bi-LSTM layer and Tree-LSTM layer. First, we preprocess the dataset for model training and employ word embedding to represent the raw corpus. Next, Bi-LSTM is utilized to predict trigger types and find trigger words. We then represent the dependency information of sentences by using Tree-LSTM which can synthesize other semantic and syntactic information from the hidden layer of Bi-LSTM. Finally, an argument is detected by Tree-LSTM and the entire event extraction process is completed.

3.1 Word Embedding

The primary purpose of word embedding is to form a lookup table to represent each word in a vocabulary with dense, low dimensional, and real-valued vectors. Although there are several methods available for word embedding, we adopt Word2Vec, which is a standard tool used in various natural language processing tasks, because of its excellent model performance with a wide range of parameters and hardware configurations [15]. In this study, we used a large number of unlabeled texts to train the word embedding model. First, we downloaded abstract texts from PubMed to build a corpus. Then we split the abstracts into sentences and tokenized each sentence into tokens. Finally, we used Word2Vec [16] to process the sentences and derive vectors via the skip-gram language model [17], as shown in Fig. 2.

3.2 Long Short-Term Memory

Using artificial neural networks in language modeling was first proposed by Bengio et al. [18], who noted that architectures such as RNNs [19] can make use of information in arbitrarily long sequences. This approach was further investigated by Mikolov et al. [20], who demonstrated that RNNs perform better than other methods in language models. However, in practice, this model is limited to looking back only a few steps because of the vanishing gradient problem. Therefore, a novel architecture called LSTM [21] has been developed over recent years. LSTM has overcome the vanishing gradient problem and solved complex chronological lag issues by using the qualities discussed previously [22].

LSTM [22] is composed of a cell, input gate, output gate, and forget gate. The input gate can ensure that information added to the cell state is important and non-redundant. The output gate can create a filter using the values of h_{t-1} and x_t (x_t is the input for the current time step. h_{t-1} is the output from the previous LSTM unit). Then, it can regulate the values that are outputted from the aforementioned vector. The forget gate can remove information that is no longer required for the LSTM from the cell state via multiplication with a filter. Each of the three gates can be considered as a multi-layer neural network that can compute an activation based on a weighted sum. The gates block or pass information based on their own sets of weights. These weights are adjusted through the recurrent network learning process. Figure 3 illustrates how data flows through a memory cell and is controlled by gates.

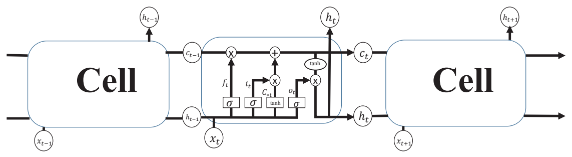

Fig. 3. LSTM cell [21]

Accoring to Fig. 3, the given inputs are multiplied by the weight matrices and a bias is added. A sigmoid function, which generates an output ranging from 0 to 1, is responsible for deciding which values to keep and which to discard. c_{t-1} is the "memory" of the previous unit. Regarding the outputs, h_t is the output of the current network. c_t is the memory of the current unit.

3.3 Bi-LSTM-Based Trigger Detection

Trigger detection, which involves the identification of event triggers and their types, is the core operation in biomedical event extraction. In our model, we employ a Bi-LSTM network [23] to detect triggers. Most previous studies treated trigger detection as a separate classification problem and trained support vector machines (SVMs) for each event type based on a rich set of features. To avoid the complexity of building a dictionary from training data and syntactic dependence, we implement LSTM to learn the information among sentences automatically during the training process.

A Bi-LSTM network [24] can access both preceding and subsequent contexts. It aims to map every word in a dictionary to a numerical vector such that the distance and relationship between vectors reflects the semantic information between words. Therefore, in this step, inputs are composed of vectors corresponding to the words in a sentence. During training, the Bi-LSTM network is able to learn and improve representations for words automatically [25]. A memory cell can be implemented as follows:

$$i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) \tag{1}$$

$$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \tag{2}$$

$$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \tag{3}$$

$$u_t = \tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \tag{4}$$

$$c_t = i_t \odot u_t + f \odot c_{t-1} \tag{5}$$

$$\vec{h}_t = o_t \odot \tanh(c_t) \tag{6}$$

where σ is the logistic sigmoid function, and i, f, o, and c are the input gate, forget gate, output gate, and cell vectors respectively, all of which are of the same size as the hidden vector h. x_t, which denotes t-th word input, is composed of word embedding and part of speech (*POS*) embeddings (word embedding for POS tags) and can be represented as $[v_t^{(w)};v_t^{(p)}]$.

Using Bi-LSTM will move inputs in two directions–from past to future and from future to past. From the above equations, the memory cells will produce a representation sequence $\vec{h}_1$, $\vec{h}_2$, $\vec{h}_3$, ..., $\vec{h}_n$ after the LSTM unit finishes recurrent computation of all tokens from left to right. Then, the counterpart $\overleftarrow{h}_t$ of $\vec{h}_t$ will be generated by another LSTM unit computing in the opposite direction. The final representation sequence $h = \{h_1, h_2, ...h_n\}$ is a concatenation of $\vec{h}_t$ and $\overleftarrow{h}_t$.

In our model, we first use Bi-LSTM to capture the syntactic and semantic information in a sentence. We then predict the trigger type by employing average pooling and logistic regression layers following the Bi-LSTM network, as illustrated in Fig. 2. To train a good classifier, we require a wide range of features, such as word embeddings, POS tags, and semantic information. Additionally, we were inspired by the one-against-all [26] strategy to solve the problem that one sentence may include more than one trigger of different types. We construct one classifier for each class and each classifier is trained to distinguish between samples of its corresponding class and samples of all other classes. The classifiers determine whether or not the current sentence includes triggers belonging to one or more classes.

If the sentence does include triggers belong to certain classes, we continue to extract trigger words from the sentence, as shown in Figure 2. We employ a softmax after the output layer of Bi-LSTM, to find which vector has achieved the highest score. Then the corresponding input of the selected vector can be considered as the trigger. This process can be represented by the following equations:

$$p(y_t = 1|h_{\mathrm{t}}) = \frac{\exp(Wh_{\mathrm{t}})}{\sum_{t=1}^{N} \exp(Wh_t)} \tag{7}$$

Here, W represents weight matrices and b represents bias vectors.

3.4 Tree-LSTM-Based Argument Detection

In this part, Tree-LSTM is used to denote the type of relationship between each target pair. Target pair candidates for argument detection are built using all possible combinations of the trigger word and the remaining words in the sentence. For example, we first detect the word "transcription" as the trigger for the "Transcription" type. We then create a target pair candidates for further analysis. Each target pair is composed of "transcription" and another word from the sentence, such as ("transcription", "stimulates"), ("transcription", "c-fos"). If the detected triggers are wrong or there is no relationship between the pair, this pair will be labeled as the "Other" type. Otherwise, this pair can be considered as the argument of the trigger word. Usually, the argument exist between one trigger words and another trigger words, or between trigger words and the Proper noun of biomedical.

First, we create candidate pairs according to the trigger word found in previous step. Then we build a dependency tree layer by employing Tree-LSTM. According to [13], features of nodes are mainly constructed based on the shortest dependency path. Next, for each candidate pair, its new vector can be represented by the combination of its corresponding hidden layer in both Bi-LSTM network and Tree-LSTM network. The softmax layer then receives the target candidate vectors and will make a prediction.

As for dependency tree layer, it represents the relationships between two words in the dependency tree [13]. Typically, this layer largely focuses on the shortest dependency path (SDP) between two entities. This method was shown to be effective at learning sentence embeddings and syntax jointly [14] as well as relations between words in sentences [27], where the shortest dependency paths were able to retain the most relevant information and eliminate irrelevant words in a sentence.

In addition, in order to capture more information of the target word pair, we were inspired to adopt bidirectional Tree-LSTM to represent the relationships between words [9]. First, we employ bottom-up LSTM to propagate information from the leaves to each of the nodes. We then use the nodes at the top of the tree to propagate information from the root via top-down LSTM. In this section, we refer to the tree-structured LSTM-RNN proposed by Miwa et al. [9], which overcomes the limitations of the two variants of tree-structured LSTM-RNNs proposed by Tai et al.

Similar to standard LSTM units, each Tree-LSTM unit (indexed by t) contains input and output gates i_t and o_t, respectively, a memory cell c_t, and hidden state h_t [28]. The difference is that Tree-LSTM units are dependent on the states of potentially numerous child units for gating vectors and updating memory cells. The Tree-LSTM used in this study can defined as follows:

$$i_t = \sigma(W^{(i)}x_t + \sum_{l\in C(t)} U_l^{(i)} h_{tl} + b^{(i)}) \tag{8}$$

$$f_{tl} = \sigma(W^{(f)}x_t + \sum_{l\in C(t)} U_l^{(f)} h_{tl} + b^{(f)}) \tag{9}$$

$$o_t = \sigma(W^{(o)}x_t + \sum_{l\in C(t)} U_l^{(o)} h_{tl} + b^{(o)}) \tag{10}$$

$$u_t = \tanh(W^{(u)}x_t + \sum_{l\in C(t)} U_l^{(u)} h_{tl} + b^{(u)}) \tag{11}$$

$$c_t = i_t \odot u_t + \sum_{l\in C(t)} f_{tl} \odot c_{tl} \tag{12}$$

$$h_t = o_t \odot \tanh(c_t) \tag{13}$$

where $C(t)$ is the number of children of the t-th node, subscript tl means t-th node and its l-th child and same-type children share a common weight matrix U.

Intuitively, one can interpret each parameter matrix in these equations as an encoding of the correlations between the component vectors of the Tree-LSTM units, input x_t, and hidden states h_t of a unit's children. Additionally, because the Tree-LSTM unit contains one forget gate f_{tl} for each child l, it can selectively incorporate information from each child.

In order to detect argument based on the dependency analysis, we represent the t-th candidate vector constructed by the dependency tree layer as $o_t^d = [h_t^{root}; h_t^{trigger}; h_t^{entity}]$, where h_t^{root} is the vector from the lowest common ancestor of the target pair t, which is generated by bottom-up LSTM. $h_t^{trigger}$ denotes the hidden state vector of the trigger word in terms of Tree-LSTM units and h_t^{entity} represents the vector of another entity in the target pair candidate. The argument prediction process can be defined as follows:

$$h_{\mathrm{t}}^{(d)} = \tanh(W^{(dh)} \mathrm{o}_t^d + \mathrm{b}^{(dh)}) \tag{14}$$

$$y_t = soft\max(W^{(dy)} h_{\mathrm{t}}^{(d)} + \mathrm{b}^{(dy)}) \tag{15}$$

where W represents weight matrices and b represents bias vectors.

3.5 End-to-End Model Training

In this research, we combine Tree-LSTM (corresponding to argument detection) and Bi-LSTM (corresponding to trigger detection), as the inputs of the argument detection are composed of vectors from the Bi-LSTM layer and Tree-LSTM layer.

In previous studies, researchers often constructed two models to separate trigger detection and argument detection with two independent loss functions. In contrast, we adopt an end-to-end model with a single loss function in this study. First, we apply a dropout operation to the embedding layers and hidden layers in the model. We then choose a training set to train the model and calculate the total loss from trigger detection and argument detection. The total loss can be defined as $loss = l_1 + l_2$, where l_1 is the loss from trigger detection and l_2 is the loss from argument detection. The details of l_1 and l_2 can be represented by the following equation:

$$l_1 = -\sum_{j=1}^{N} [y_j log\hat{y}_j + (1 - y_j) log(1 - \hat{y}_j)] \tag{16}$$

$$l_2 = -\sum_{j=1}^{N} \sum_{i=1}^{C_j} y_{ji} log(\hat{y}_{ji}) \tag{17}$$

where N is the length of a sentence; C_j represents the number of children of each node; y_j and y_{ji} are the actual labels; $\hat{y}_j$ and $\hat{y}_{ji}$ represent the prediction results.

Next, we adopt back propagation through time (BPTT) and the Adam algorithm with gradient clipping, parameter averaging, and L2-regularization to update the parameters of the model[1].

Additionally, we trained n models (n corresponds to the number of event types in the study, where all datasets for each model were the same) during our experiment to solve the problem of one sentence potentially including an arbitrary number of triggers for different types of events. In this manner, a binary classifier for trigger type detection was created for each of the event types. This classifier can determine whether or not a sentence contains one of the trigger types. If the sentence contains a trigger type, we continue with the remaining steps to find specific triggers and detect arguments. Otherwise, we continue to prediction for the next sentence and repeat the previous steps until all sentences in the current model have been tested.

1. The source code is publicly available at https://github.com/deardelia/LSTM-based-end-to-end-for-biomedical-event-extraction.git

4 Experimental Study

4.1 Experiment Configuration

In this study, two datasets were employed. The first is prepared for training Word2Vec and is composed of unlabeled abstract texts from a public database called PubMed, which is approximately 6 GB in size, including biological publications from 2010 to 2017. These abstracts were prepared for training Word2Vec because some rare biological terms require more data to construct more accurate word embeddings. The other dataset is downloaded from BioNLP'09, BioNLP'11 and BioNLP'13, which consist of training sets, development sets, and test sets. Table 1 lists the event types addressed in the BioNLP'09 and BioNLP'11 Genia task. There is a modification of the event types defined in BioNLP'13's Genia task, which adds a new type called "Protein modification" including "Phosphorylation" type. The details of training data has been showed in 2.

The raw corpus mined from PubMed can be processed by Word2Vec to extract embeddings. Texts downloaded from BioNLP'09, BioNLP'11 and BioNLP'13 must be preprocessed before they can be sent into the training model. First, we split them into sentences using the Genia Sentence Splitter [29]. The tokenized sentences are then parsed using the StanfordParser [30] and tagged using the Stanford-postagger [31].

During the Bi-LSTM and Tree-LSTM process, we add dropout [32] into the embedding layer and hidden layers in the model to avoid overfitting. Dropout can be regarded as a way to regularize neural network and the core idea of dropout is to randomly set some of the neurons to zero during the forward pass of network.

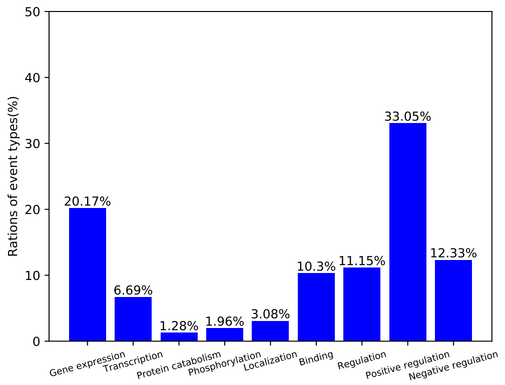

Fig. 4. Statistics for BioNLP'11 Genia Corpus

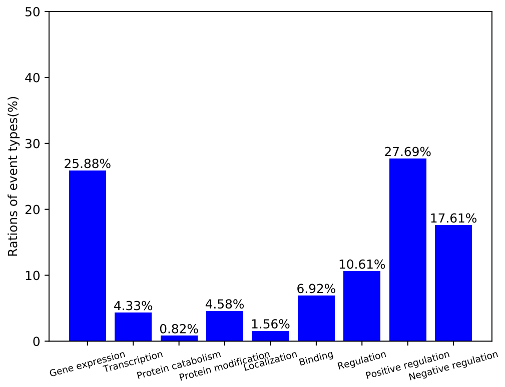

Fig. 5. Statistics for BioNLP'13 Genia Corpus

4.2 Evaluation Metrics and Baselines

The purpose of this task is to denote the event types, the trigger words, types between arguments.In this study, we adopted the Strict Equality[2] evaluation criteria defined by BioNLP Shared Task to judge whether we have extracted correct biomedical events. to define the c. All experiments were conducted on the corpora provided by BioNLP'09 BioNLP'11 and BioNLP'13 and the statistics for the BioNLP'11 and BioNLP'13 corpus are listed in Fig. 4 and Fig. 5, respectively (task definition in BioNLP'09 remains the same as BioNLP'11).

We adopted commonly used evaluation metrics in BioNLP task for our study: precision (P), recall (R), and F1 score [1]. In this study, we compared our method to several previously proposed and recently developed methods to investigate our method's efficiency. These baseline methods are summarized below:

1) Björne et al. [33] used SVMs to detect event-defining words, followed by the detection of their relationships. Their method is characterized by heavy reliance on efficient, state-of-the-art machine learning (ML) techniques and a wide array of features derived from full dependency analysis of each sentence.

2. http://www.nactem.ac.uk/tsujii/GENIA/SharedTask/evaluation.shtml

2) Riedel and McCallum [34] adopted a PA online learning algorithm to predict the confidence of triggers and arguments, then extracted the events with the highest confidence under certain constraints using dual decomposition.

3) Li et al. [6] adopted a combination of dual decomposition and rich features integrating word embedding to detect events. Their model not only extracts rich features based on dependency parsing, but can also jointly extract events to alleviate cascading errors using dual decomposition.

4) Hakala et al. [35] tested the availability of the large scale text mining resource EVEX to provide supporting information for an existing event extraction system. The extraction was carried out using a combination of the BANNER named entity detector [36] and the TEES event extraction system [33].

5) Björne and Salakoski [37] integrated their convolutional neural network into the open source Turku Event Extraction System (TEES) framework [33], which used a linear representation of the input text and could encode the information with various vector space embeddings.

4.3 Results and Discussion

We summarize our experimental setting as follows. The Bi-LSTM used for trigger detection are built with 2 hidden layers while the length of dependency tree has been set 20. Besides, the batch size is set as 32 and epoch is set as 500.

TABLE 1
Event Types

Event type	Corpora	Core arguments	Optional arguments
Gene expression	GE, ID	Theme (Protein, Regulon/Operon)	
Transcription	GE, ID	Theme (Protein, Regulon/Operon)	
Protein catabolism	GE, ID	Theme (Protein)	
Phosphorylation	GE, ID,EPI	Theme (Protein)	Site(Entity)
Localization	GE, ID,BB	Theme (Protein, Core entity), Bacterium(Bacterium) Localization (Host, HostPart...)	AtLocGE (Entity), ToLocGE(Entity)
Binding	GE, ID	Theme (Protein, Core entity)	Site(Entity)
Regulation	GE, ID	Theme (Protein, Core entity, Event), Cause (Core entity, Event)	Site (Entity), CSite (Entity)
Positive regulation	GE, ID	Theme (Protein, Core entity, Event), Cause (Core entity, Event)	Site (Entity), CSite (Entity)
Negative regulation	GE, ID	Theme (Protein, Core entity, Event), Cause (Core entity, Event)	Site (Entity), CSite (Entity)

TABLE 2
Details of training data

Corpus	Number of documents	Number of annotatioins
Genia Event (GE) in BioNLP 2009	800 Abstracts	9300 protein, 8,597 events
Genia Event (GE) in BioNLP 2011	1210 abstracts, 14 full-text	21616 protein, 24967 events
Genia Event (GE) in BioNLP 2013	34 full-text	12068 protein, 9364 events

For the sake of analysis, Table 3 compares our method with Li et al.'s based on the development set of BioNLP'09 (as Li et al. only provides the experimental results based on the development set in BioNLP'09). The size of development set and test set are similar to each other, however, there is no gold annotation on test set. Then Table 4 lists the experimental results of our method compared with other baselines based on the test set of BioNLP'09. In both Table 3 and Table 4, SIMPLE represents all simple events including "Gene expression", "Transcription", "Protein catabolism" and "Localization", BIND equals "Binding" event just as mentioned above, REG refers to complex events which are composed of "Regulation", "Positive Regulation" and "Negative Regulation", TOTAL means the overall performance results.

TABLE 3
Detailed results based on the development set of BioNLP'09

Event Category	Method	Recall(%)	Precision(%)	F1(%)
SIMPLE	Li et al.	76.74	83.59	80.02
	Ours	**83.59**	**83.67**	**83.62**
BIND	Li et al.	42.74	61.27	50.36
	Ours	**44.69**	**65.47**	**53.12**
REG	Li et al.	43.58	57.30	49.51
	Ours	**48.76**	**58.33**	**53.12**
TOTAL	Li et al.	53.83	67.18	59.77
	Ours	**55.09**	**68.18**	**60.94**

From Table 3 and Table 4, it is found that in almost all the classes, our method performs better in terms of recall compared to the baseline methods. According to the precision metric, we achieved better performance in most categories, which indicates that our method can efficiently extract biomedical events from text. In terms of F1 score, which is a synthesized evaluation metric considering both precision and recall, our model achieved better performance compared to the three baseline methods on all types. In particular, our method achieved satisfactory performance for the "Binding" events, which are difficult to extract completely because of uncertainty regarding the number of themes. One reason for this result is that our method is able to capture rich syntactic and semantic information via Bi-LSTM and Tree-LSTM, and continuously optimize the relationships between target pairs based on loss, which can efficiently reduce incomplete event extractions. Improvement on simple events, such as "Gene expression," "Transcription," "Phosphorylation," and "Localization," is inconspicuous because these events are relatively easy to extract because of their simple structure, which includes only one theme and one trigger.

Although our method can improve complex event extraction (including "Regulation," "Positive Regulation," and "Negative Regulation") to some degree, it is still difficult to achieve an ideal result, because complex events sometimes have themes or triggers spread across multiple sentences. Additionally, the themes of complex events not only contain proteins, or a specific domain or region of those proteins, but may also contain triggers of other events.

To further prove the effectiveness of our method, we compared it to the three baseline methods based on the test set of BioNLP'11 and BioNLP'13. The experimental results are listed in Table 5 and Table 6. The results also reveal that the proposed method outperforms the baseline methods in terms of recall, precision, and F1 Score.

In order to prove the efficiency of the Tree-LSTM, Table 7compare the results achieved by different methods used in argument detection. The results show that Tree-LSTM is good at dealing with argument detection and it can also capture more useful semantic information than the traditional Bi-LSTM can.

Besides, it is also important to study the suitable hyperparameters because selecting optimal parameters for a neural network architecture can often make the difference between mediocre and state-of-the-art performance [38]. In order to find proper values for the parameters used in our model, we conducted experiments on several hyperparameters, including embedding dimensionality and learning rate.

For embedding dimensionality, we tested values of 50, 100, 150, 200, and 300 to find the optimal value for event extraction. From the results listed in Table 8, one can see

TABLE 4
Detailed results based on the test set of BioNLP'09

Event Category	Method	Recall(%)	Precision(%)	F1(%)
Gene expression	Björne et al.	69.81	78.50	73.90
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	–
	Ours	**79.26**	**85.34**	**82.19**
Transcription	Björne et al.	39.42	69.23	50.23
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	–
	Ours	**62.73**	**86.88**	**72.88**
Protein catabolism	Björne et al.	42.86	66.67	52.17
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	–
	Ours	**77.66**	**85.61**	**81.44**
Localization	Björne et al.	49.43	**81.90**	61.65
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	–
	Ours	**90.97**	81.76	**86.21**
Binding	Björne et al.	40.46	49.82	44.41
	Björne and Salakoski(2018)	–	–	–
	Riedel and McCallum	–	–	48.0
	Ours	**44.69**	**65.47**	**53.12**
Regulation	Björne et al.	25.43	38.14	30.52
	Björne and Salakoski			
	Riedel and McCallum	–	–	–
	Ours	**45.33**	**61.33**	**52.13**
Positive regulation	Björne et al.	38.76	48.72	43.17
	Björne and Salakoski			
	Riedel and McCallum	–	–	–
	Ours	**41.47**	**62.16**	**49.75**
Negative regulation	Björne et al.	35.36	43.46	38.99
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	–
	Ours	**45.36**	**59.12**	**51.34**
SIMPLE	Björne et al.	64.21	77.45	70.21
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	72.6
	Ours	**80.36**	**83.67**	**81.98**
BIND	Björne et al.	40.46	49.82	44.41
	Björne and Salakoski	–	–	–
	Riedel and McCallum	–	–	52.6
	Ours	**44.69**	**65.47**	**53.12**
REG	Björne et al.	35.63	45.87	40.11
	Björne and Salakoski			
	Riedel and McCallum	–	–	46.9
	Ours	**48.76**	**58.33**	**53.12**
TOTAL	Björne et al.	46.73	58.48	51.95
	Björne and Salakoski	49.34	**69.87**	57.84
	Riedel and McCallum	–	–	57.4
	Ours	**54.33**	66.12	**59.68**

that 200 is the best choice for embedding dimensionality. For learning rate, which tells the optimizer how far to move the weights in the direction of the gradient for a mini-batch, the experimental results are illustrated in Fig. 6 and as a result, we set learning rate to 0.001. Besides, the dropout rate is set 0.5 because a dropout rate of 0.5 has been shown to be effective in the Fig. 7.

TABLE 7
Comparing the efficiency of different argument detection methods (based on the development set from BioNLP'09)

Event Category	Method	Recall(%)	Precision(%)	F1(%)
SIMPLE	Bi-LSTM	78.32	80.17	79.23
	Tree-LSTM	**83.59**	**83.67**	**83.62**
BIND	Bi-LSTM	40.33	62.98	49.17
	Tree-LSTM	**44.69**	**65.47**	**53.12**
REG	Bi-LSTM	43.07	52.99	47.52
	Tree-LSTM	**48.76**	**58.33**	**53.12**
TOTAL	Bi-LSTM	50.22	63.14	55.93
	Tree-LSTM	**55.09**	**68.18**	**60.94**

5 Conclusions and Future Work

In this paper, we proposed a system to extract structured events from biomedical publications. As previous models continue to face challenges in terms of cascading errors between trigger detection and argument detection that will affect the final results, we proposed an end-to-end method based on Bi-LSTM and Tree-LSTM to extract events. This allows us to predict triggers and detect arguments using a single model. Semantic and syntactic information between sentences can be learned automatically using Bi-LSTM. Tree-LSTM is then able to detect relationships between target pairs based on dependency analysis. In addition, word embedding features are introduced at the beginning of the model to improve performance. Furthermore, in order to to

TABLE 5
Detailed results based on the test set of BioNLP'11

Event Category	Method	Recall(%)	Precision(%)	F1(%)
SIMPLE	Björne et al.	68.22	76.47	73.50
	Björne and Salakoski	–	–	–
	Riedel and McCallum	67.01	81.40	78.40
	Li et al.	73.47	83.30	78.07
	Ours	**69.87**	**88.57**	**81.98**
BIND	Björne et al.	42.97	43.60	43.28
	Björne and Salakoski	–	–	–
	Riedel and McCallum	42.97	56.42	48.79
	Li et al.	42.63	**60.23**	49.92
	Ours	**47.10**	**60.23**	**52.86**
REG	Björne et al.	38.72	47.64	42.72
	Björne and Salakoski	–	–	–
	Riedel and McCallum	37.52	**52.67**	43.82
	Li et al.	39.10	51.46	44.44
	Ours	**48.76**	42.42	**45.37**
TOTAL	Björne et al.	49.56	57.65	53.30
	Björne and Salakoski	49.94	**69.45**	58.10
	Riedel and McCallum	48.49	64.08	55.20
	Li et al.	51.25	64.40	57.08
	Ours	**53.07**	64.51	**58.23**

TABLE 6
Detailed results based on the test set of BioNLP'13

Event Category	Method	Recall(%)	Precision(%)	F1(%)
SIMPLE	Björne and Salakoski	–	–	–
	Hakala et al.	–	–	76.59
	Li et al.	**76.11**	83.31	79.55
	Ours	75.98	**84.27**	**79.91**
Protein modification	Björne and Salakoski	–	–	–
	Hakala et al.	–	–	65.37
	Li et al.	68.06	81.25	74.07
	Ours	**69.28**	**83.44**	**75.70**
BIND	Björne and Salakoski	–	–	–
	Hakala et al.	–	–	42.88
	Li et al.	46.25	45.43	45.83
	Ours	**47.28**	**45.76**	**46.51**
REG	Björne and Salakoski	–	–	–
	Hakala et al.	–	–	38.41
	Li et al.	34.21	**47.81**	39.88
	Ours	**38.97**	46.31	**42.32**
TOTAL	Björne and Salakoski	**65.78**	44.38	53.30
	Hakala et al.	58.03	45.44	50.97
	Li et al.	47.96	**59.71**	53.19
	Ours	56.12	58.73	**57.39**

TABLE 8
The influence of dimensionality on event extraction (based on the development set from BioNLP'09)

Feature	Trigger(F1)%	Argument(F1)%	Event(F1)%
Baseline50	72.48	66.97	58.03
Baseline100	72.57	67.31	58.97
Baseline150	72.65	67.43	59.76
Baseline200	**73.02**	**67.64**	**60.75**
Baseline300	72.55	67.59	59.34

implement an end-to-end framework, we make the hidden layer from Bi-LSTM as well as the hidden layer from Tree-LSTM serve as two input for argument detection, and then we can train the model using an overall loss function and reduce cascading errors to some degree.

Despite its promising performance, some limitations exist in our model that deserve further consideration. As shown in the experiments section, the extraction of complex events remains a significant challenge. In addition, an imbalanced data distribution among different types will also affect final performance. Therefore, in the future, we will attempt to apply co-reference resolution [45] in our model to solve the issue of some complex events potentially featuring themes or causes occurring in other sentences. Finally, handling imbalanced datasets is another problem we will attempt to resolve.

ACKNOWLEDGEMENTS

This work was partially supported by the State Key Laboratory of Software Development Environment of China (No. SKLSDE-2017ZX-16) the National Natural Science Foundation of China (No. 61772132).

REFERENCES

[1] J. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii, "Overview of BioNLP'09 shared task on event extraction," in *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, 2009, pp. 1–9.

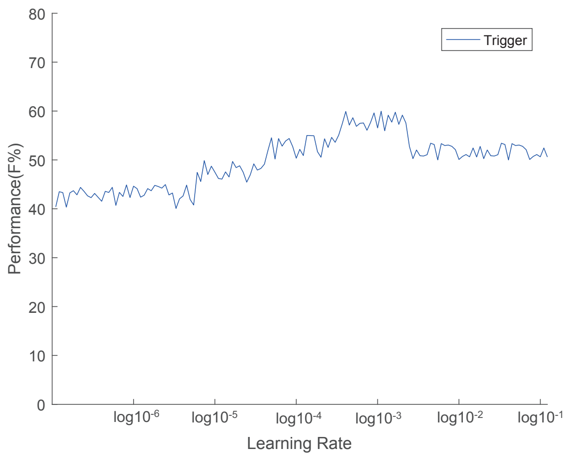

Fig. 6. The influence of learning rate on event extraction(in BioNLP09).

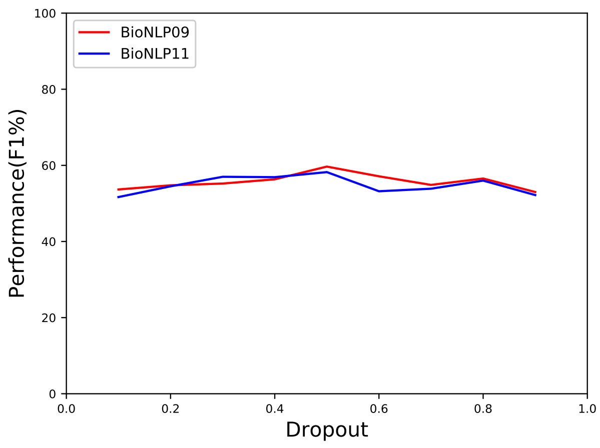

Fig. 7. The influence of dropout value on event extraction.

[2] G. Gonzalez, T. Tahsin, B. C. Goodale, A. C. Greene, and C. S. Greene, "Recent advances and emerging applications in text and data mining for biomedical discovery," *Briefings in Bioinformatics*, vol. 17, no. 2, pp. 33–42, 2016.

[3] S. Riedel, H. Chun, T. Takagi, and J. Tsujii, "A Markov logic approach to bio-molecular event extraction," in *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, 2009, pp. 41–49.

[4] H. Poon and L. Vanderwende, "Joint inference for knowledge extraction from biomedical literature," in *Proceedings of 2010 Conference of the North American Chapter of the Association of Computational Linguistics*, 2010, pp. 813–821.

[5] S. Riedel and A. McCallum, "Fast and robust joint models for biomedical event extraction," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011, pp. 1–12.

[6] L. Li, S. Liu, M. Qin, Y. Wang, and D. Huang, "Extracting biomedical event with dual decomposition integrating word embeddings," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 13, no. 4, pp. 669–677, 2016.

[7] A. Komninos and S. Manandhar, "Dependency based embeddings for sentence classification tasks," in *Proceedings of 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1490–1500.

[8] P. Bhattacharyya, S. Pawar, and G. K. Palshikar, "End-to-end relation extraction using neural networks and Markov logic networks," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017, pp. 818–827.

[9] M. Miwa and M. Bansal, "End-to-end relation extraction using LSTMs on sequences and tree structures," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 1105–1116.

[10] L. Li, J. Zheng, J. Wan, D. Huang, and X. Lin, "Biomedical event extraction via long short term memory networks along dynamic extended tree," in *IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2016, Shenzhen, China, December 15-18, 2016*, 2016, pp. 739–742. [Online]. Available: https://doi.org/10.1109/BIBM.2016.7822612

[11] A. Zhao, L. Qi, J. Dong, and H. Yu, "Dual channel LSTM based multi-feature extraction in gait for diagnosis of neurodegenerative diseases," *Knowledge-Based Systems*, vol. 145, pp. 91–97, 2018.

[12] J. P. C. Chiu and E. Nichols, "Named entity recognition with bidirectional LSTM-CNNs," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 357–370, 2016.

[13] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, 2015, pp. 1556–1566.

[14] J. Maillard, S. Clark, and D. Yogatama, "Jointly learning sentence embeddings and syntax with unsupervised Tree-LSTMs," *CoRR*, vol. abs/1705.09189, 2017.

[15] L. Ma and Y. Zhang, "Using word2vec to process big text data," in *2015 IEEE International Conference on Big Data, Big Data 2015, Santa Clara, CA, USA, October 29 - November 1, 2015*, 2015, pp. 2895–2897. [Online]. Available: https://doi.org/10.1109/BigData.2015.7364114

[16] K. W. Church, "Word2vec," *Natural Language Engineering*, vol. 23, no. 1, pp. 155–162, 2017.

[17] Y. Nie, W. Rong, Y. Zhang, Y. Ouyang, and Z. Xiong, "Embedding assisted prediction architecture for event trigger identification," *Journal of Bioinformatics and Computational Biology*, vol. 13, no. 3, 2015.

[18] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[19] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[20] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of 11th Annual Conference of the International Speech Communication Association*, 2010, pp. 1045–1048.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] B. Cortez, B. Carrera, Y. Kim, and J. Jung, "An architecture for emergency event prediction using LSTM recurrent neural networks," *Expert Systems with Applications*, vol. 97, pp. 315–324, 2018.

[23] R. Ghaeini, S. A. Hasan, V. V. Datla, J. Liu, K. Lee, A. Qadir, Y. Ling, A. Prakash, X. Z. Fern, and O. Farri, "Dr-bilstm: Dependent reading bidirectional LSTM for natural language inference," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018, pp. 1460–1469.

[24] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *CoRR*, vol. abs/1508.01991, 2015.

[25] I. J. Unanue, E. Z. Borzeshi, and M. Piccardi, "Recurrent neural networks with specialized word embeddings for health-domain named-entity recognition," *Journal of Biomedical Informatics*, vol. 76, pp. 102–109, 2017.

[26] R. K. Eichelberger and V. S. Sheng, "Does one-against-all or one-against-one improve the performance of multiclass classifications?" in *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, 2013, pp. 1609–1610.

[27] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin, "Classifying relations via long short term memory networks along shortest dependency paths," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1785–1794.

[28] A. Katiyar and C. Cardie, "Investigating LSTMs for joint extraction of opinion entities and relations," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 919–929.

[29] R. Satre, K. Yoshida, A. Yakushiji, Y. Miyao, Y. Matsubayashi, and T. Ohta, "AKANE system: Protein-protein interaction pairs in the

BioCreAtIvE2 challenge, PPI-IPS subtask," in *Proceedings of the 2nd BioCreative Challenge Evaluation Workshop*, 2007, pp. 209–212.

[30] H. Xu, S. AbdelRahman, M. Jiang, J. Fan, and Y. Huang, "An initial study of full parsing of clinical text using the stanford parser," in *Proceedings of 2011 IEEE International Conference on Bioinformatics and Biomedicine Workshops*, 2011, pp. 607–614.

[31] K. Yordanova, "A simple model for improving the performance of the stanford parser for action detection in textual instructions," in *Proceedings of the 2017 International Conference Recent Advances in Natural Language Processing*, 2017, pp. 831–838.

[32] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[33] J. Björne, J. Heimonen, F. Ginter, A. Airola, T. Pahikkala, and T. Salakoski, "Extracting contextualized complex biological events with rich graph-based feature sets," *Computational Intelligence*, vol. 27, no. 4, pp. 541–557, 2011.

[34] S. Riedel and A. McCallum, "Robust biomedical event extraction with dual decomposition and minimal domain adaptation," in *Proceedings of BioNLP Shared Task 2011 Workshop*, 2011, pp. 46–50.

[35] K. Hakala, S. V. Landeghem, T. Salakoski, Y. V. de Peer, and F. Ginter, "EVEX in ST'13: Application of a large-scale text mining resource to event extraction and network construction," in *Proceedings of the BioNLP Shared Task 2013 Workshop*, 2013, pp. 26–34.

[36] R. Leaman and G. Gonzalez, "BANNER: an executable survey of advances in biomedical named entity recognition," in *Biocomputing 2008, Proceedings of the Pacific Symposium, Kohala Coast, Hawaii, USA, 4-8 January 2008*, 2008, pp. 652–663.

[37] J. Björne and T. Salakoski, "Biomedical event extraction using convolutional neural networks and dependency parsing," in *Proceedings of the BioNLP 2018 workshop, Melbourne, Australia, July 19, 2018*, 2018, pp. 98–108.

[38] N. Reimers and I. Gurevych, "Optimal hyperparameters for deep LSTM-networks for sequence labeling tasks," *CoRR*, vol. abs/1707.06799, 2017.

[39] A. Wang, J. Wang, H. Lin, J. Zhang, Z. Yang, and K. Xu, "A multiple distributed representation method based on neural network for biomedical event extraction," *BMC Medical Informatics and Decision Making*, vol. 17, no. S-3, pp. 59–66, 2017.

[40] Q. Bui, D. Campos, E. M. van Mulligen, and J. A. Kors, "A fast rule-based approach for biomedical event extraction," in *Proceedings of the BioNLP Shared Task 2013 Workshop*, 2013, pp. 104–108.

[41] J. Lever and S. J. Jones, "VERSE: event and relation extraction in the BioNLP 2016 shared task," in *Proceedings of the 4th BioNLP Shared Task Workshop*, 2016, pp. 42–49.

[42] A. Wang, J. Wang, H. Lin, J. Zhang, Z. Yang, and K. Xu, "Biomedical event extraction based on distributed representation and deep learning," in *Proceedings of 2016 IEEE International Conference on Bioinformatics and Biomedicine*, 2016, p. 775.

[43] F. Mehryary, J. Björne, S. Pyysalo, T. Salakoski, and F. Ginter, "Deep learning with minimal training data: TurkuNLP entry in the BioNLP shared task 2016," in *Proceedings of the 4th BioNLP Shared Task Workshop*, 2016, pp. 73–81.

[44] Q. Li and H. Ji, "Incremental joint extraction of entity mentions and relations," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014, pp. 402–412.

[45] K. B. Cohen, A. Lanfranchi, M. J. Choi, M. Bada, W. A. B. Jr., N. Panteleyeva, K. Verspoor, M. Palmer, and L. E. Hunter, "Coreference annotation and resolution in the colorado richly annotated full text (CRAFT) corpus of biomedical journal articles," *BMC Bioinformatics*, vol. 18, no. 1, pp. 372:1–372:14, 2017.