Bioinformatics, 36(2), 2020, 637–643
doi: 10.1093/bioinformatics/btz607
Advance Access Publication Date: 8 August 2019
Original Paper

Data and text mining

Context awareness and embedding for biomedical event extraction

Shankai Yan and Ka-Chun Wong *

Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong SAR 999077

*To whom correspondence should be addressed.

Associate Editor: Jonathan Wren

Received on September 23, 2018; revised on July 26, 2019; editorial decision on July 29, 2019; accepted on August 6, 2019

Abstract

Motivation: Biomedical event extraction is fundamental for information extraction in molecular biology and biomedical research. The detected events form the central basis for comprehensive biomedical knowledge fusion, facilitating the digestion of massive information influx from the literature. Limited by the event context, the existing event detection models are mostly applicable for a single task. A general and scalable computational model is desiderated for biomedical knowledge management.

Results: We consider and propose a bottom-up detection framework to identify the events from recognized arguments. To capture the relations between the arguments, we trained a bidirectional long short-term memory network to model their context embedding. Leveraging the compositional attributes, we further derived the candidate samples for training event classifiers. We built our models on the datasets from BioNLP Shared Task for evaluations. Our method achieved the average *F*-scores of 0.81 and 0.92 on BioNLPST-BGI and BioNLPST-BB datasets, respectively. Comparing with seven state-of-the-art methods, our method nearly doubled the existing *F*-score performance (0.92 versus 0.56) on the BioNLPST-BB dataset. Case studies were conducted to reveal the underlying reasons.

Availability and implementation: https://github.com/cskyan/evntextrc.

Contact: kc.w@cityu.edu.hk

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The unbridled growth of publications in biomedical literature databases offers a great opportunity for researchers to stand on the shoulders of giants for cutting-edge advancements. Nonetheless, it is also a challenge to digest the extensive information from the huge volume of textual data in a heterogeneous manner. Information extraction (IE) is an effective approach to summarize the knowledge into expressive forms for management and comprehension; it can be integrated with other knowledge resources for innovative discovery (Rebholz-Schuhmann *et al.*, 2012). Examples include protein–protein interactions (Mallory *et al.*, 2016), drug–drug interaction (Zhao *et al.*, 2016), causal relationships between biological entities (Perfetto *et al.*, 2016) and other topic-oriented association mining systems (Cañada *et al.*, 2017; Lim *et al.*, 2016).

Over the past decades, considerable efforts have been devoted toward rule-based (Bui and Sloot, 2012) and trigger-based (Bjorne *et al.*, 2010; Björne and Salakoski, 2011) detection methods for biomedical event extraction from PubMed abstracts (Ananiadou *et al.*, 2010). In general, trigger detection dominates the whole prediction process whose performance greatly affects the final event detection (Pyysalo *et al.*, 2012). Trigger identification method has been well-studied and improved. The latest trigger-based approach using deep neural network has shown its strength in general event extraction tasks (Nguyen *et al.*, 2016). Combining with lexical and semantic features, word embedding (Mikolov *et al.*, 2013) is proposed to build an advanced trigger classifier (Zhou *et al.*, 2014). Nevertheless, trigger detection is a multiclass classification problem with limited annotation labels. The well-known datasets from BioNLP Shared Task (BioNLPST) include BioNLP'09 (Kim *et al.*, 2009), BioNLP'11 (Kim *et al.*, 2011), BioNLP'13 (Nédellec *et al.*, 2013) and BioNLP'16 (Nédellec *et al.*, 2016). The trigger-based methods are based on the dependency parse tree and character *n*-grams. The dependency parser in natural language processing (NLP) is well-studied (Nivre *et al.*, 2016) and has been developed from empirical techniques to neural network models (Chen and Manning, 2014). However, there is a performance deviation from the traditional applications when applied to biomedical literature due to the contextual variations. The parser that was developed specifically for biomedical text mining (BioNLP) such as McCCJ (McClosky and Charniak, 2008) is necessary for biomedical IE (Luo *et al.*, 2017). Bidirectional long short-term memory (BiLSTM) has been applied to medical event detection in clinical records (Jagannatha and Yu, 2016). Nonetheless, its events are binary relations which are very different from the complex events in BioNLPST.

One of the major concerns behind this is that the trigger prediction errors would propagate along the downstream tasks. The training data for trigger detection is quite limited because the ground truth labels are not even given in the BioNLP Shared Task datasets. In addition, the training samples are not easily selected manually. Consequently, it becomes an unbalanced multiclass classification problem which is the main barrier for performance improvement in the subsequent biomedical text mining tasks.

In this study, we proposed a novel method to detect biomedical events using a different strategy. We do not need the annotation of triggers and the cumbersome dependency parsing for each sentence. We aspire to model the context embedding for each argument. The argument embeddings are adopted to detect directed relations. The proposed neural network model is applicable to general event extraction, thanks to the universality of the underlying neural language models (Bengio *et al.*, 2003). Our method is specially designed for biomedical event extraction while keeping replaceable components (e.g. pretrained word embedding) for general event extraction tasks. The remainder of this article is organized in the following order. First, we briefly introduce the datasets and indicate the defectiveness of the existing approaches. Next, we sketch out the framework of our approach and then elaborate the procedures in detail. After that, we evaluate our method and make a comprehensive comparison with other approaches on the BioNLP Shared Task dataset. Then, we demonstrate the effectiveness of our method by investigating the underlying reasons through experiments.

2 Datasets

In order to ensure fair comparisons among different approaches, we adopted two datasets from the BioNLP Shared Task with 1 (BioNLPST-BB) and 9 (BioNLPST-BGI) event type(s). The datasets contain the events of bacteria localization and the genetic processes concerning the bacterium Bacillus subtilis, respectively. The entities are annotated with entity types in both training and testing set. In each annotated event, the involved entities have been assigned different roles called argument types and the event contains a direction pointing from one to another. We aim to measure how the performance changes with different event types for model generalization estimation. The development set is initially used to validate the prediction model or tune the hyperparameters. However, it only contains 3 out of 10 event types in BioNLPST-BGI. Therefore, we combine the training set and the development set as a single annotated dataset for each task. As shown in Table 1, the event types are extremely imbalanced in BioNLPST-BGI; it means that the event detection is an imbalanced multiclass classification problem.

The events come from the sentences of PubMed abstracts and the biological entities are annotated by curators or name entity recognition (NER) tools. The objective of event detection is to annotate the relationships among the preannotated or recognized entities. For example, the sentence 'We now report that the purified product of gerE (GerE) is a DNA-binding protein that adheres to the promoters for cotB and cotC'. has totally six preannotated entities, 'T1: purified product of gerE', 'T2: GerE', 'T3: DNA-binding protein', 'T4: promoters', 'T5: cotB' and 'T6: cotC'. It contains two 'PromoterOf' events (E1: promoters->cotB; E2: promoters->cotC) and two 'Interaction' events (E3: GerE->cotB; E4: GerE->cotC). The events are different from the traditional binary relations (e.g. gene–gene interaction) due to the difficulty of recognizing their directions and the diversity of the entity types as well as the event types. Under the context of knowledge graph topology, our prediction is a directed edge with a specific type instead of a plain binary relation. The mentioned example can be used to construct a directed graph with six nodes (entities) and four edges (events). We directly adopted the tokenization and NER results (e.g. 'T1: Protein', 'T2: Protein', 'T3: Protein', 'T4: Promoter', 'T5: Gene' and 'T6: Gene') from the annotated datasets.

Besides the event annotations (e.g. E1: T4->T6, E2: T4->T5, E3: T2->T5, E4: T2->T6), the argument labels (e.g. 'T1: Protein', 'T2: Protein', 'T3: Protein', 'T4: Promoter', 'T5: Gene' and 'T6: Gene') within each event type are also used in our method. Table 2 shows the summary of the numbers of argument in each task. It is obvious that the labels for the argument types are also imbalanced. The arguments are all annotated upon the recognized entities. Therefore, we assume that the error rate of the entity recognition is very low and can consider it as known information.

The triggers used in most of the existing approaches are not officially released in the datasets and they are manually annotated by the researchers. However, those trigger words vary across different tasks; it heavily requires manual preprocessing. Furthermore, the classification errors in the trigger detectors can propagate to the argument detection and event detection. In fact, the nonexistence of trigger words does not affirm the absence of events since different authors may have different writing styles and the triggers are not guaranteed to appear in the sentence. Therefore, we do not use any trigger-based method in our study. Instead, the context of the arguments within each event is considered for feature construction.

Table 2. Statistics of the arguments for two tasks in BioNLP Shared Task

Task	Argument type	Training set	Development set
BioNLP Shared Task 2011—Bacteria–gene interactions	Action	92	16
	Agent	125	15
	Entity	15	/
	Gene	36	3
	Member	15	/
	Promoter	38	/
	Protein	29	/
	Regulon	10	/
	Site	29	/
	Target	185	21
	Transcription	31	3
BioNLP Shared Task 2016—bacteria biotopes	Bacteria	168	118
	Location	260	184

Table 1. Statistics of the events for two tasks in BioNLP Shared Task

Task	Event type	Arguments	Training set	Development set
BioNLP Shared Task 2011—bacteria–gene interactions	ActionTarget	Action->Target	108	18
	Interaction	Agent->Target	126	18
	PromoterDependence	Promoter->Protein	32	/
	PromoterOf	Promoter->Gene	36	/
	RegulonDependence	Regulon->Target	11	/
	RegulonMember	Regulon->Member	15	/
	SiteOf	Site->Entity	17	/
	TranscriptionBy	Transcription->Agent	25	3
	TranscriptionFrom	Transcription->Site	14	/
BioNLP Shared Task 2016—bacteria biotopes	Lives_In	Bacteria->Location	327	223

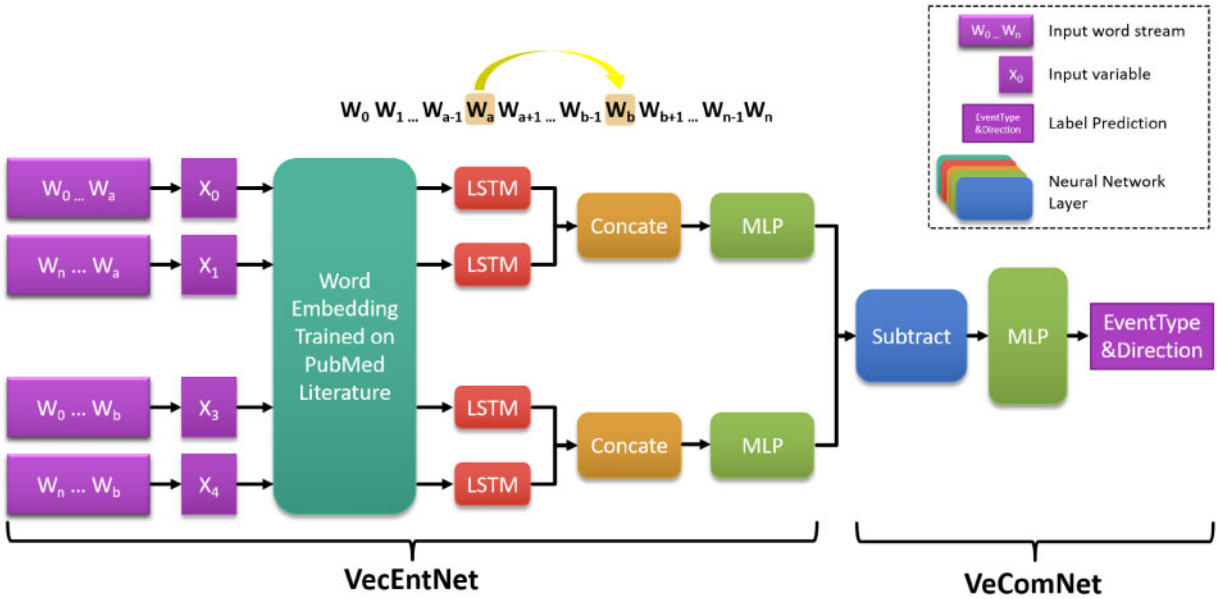

Fig. 1. Overview of the neural network architecture for argument embedding and event detection. VecEntNet is trained for argument detection using the argument annotations in the training set. The parameters of VecEntNet is then fixed and the hidden layer of the MLP in VecEntNet is used as the input of VeComNet. VeComNet is trained for directed event detection using the event annotations in the training set. The testing data are passed to VecEntNet for generating argument embedding which is put into VeComNet for event prediction

3 Methodology

3.1 An overview of the event detection framework

The overall workflow of our proposed event detection method is shown in Figure 1. We take the tokenized words in the dataset as input and transform them into word vectors trained on the PubMed literature. For each event argument W_a and W_b, we input the stream of words on both sides of them to a BiLSTM for constructing the context embeddings (Melamud *et al.*, 2016) of arguments. We train the context embedding model (VecEntNet) using the annotations of arguments in each task. The context embeddings are further used to train the event detection model (VeComNet) for detecting event types and directions.

3.2 Word embedding

To construct robust features for argument recognition, we use the distributed representations of words in a sentence instead of the traditional *N*-gram features (Mikolov *et al.*, 2013). The adopted word vectors are pretrained on a corpus of 10 876 004 biomedical abstracts from PubMed, which contains 1 701 632 distinct words and 200 dimensions (Kosmopoulos *et al.*, 2015). The training is actually a transformation from the one hot encoding of the words to a continuous space with dimension reduction. Such unsupervised training on a large corpus captures the general features of each word and help prevent overfitting.

3.3 Bidirectional LSTM

LSTM (Gers, 1999) is a recurrent neural network (RNN) cell that can be trained to decide which information should be forgotten or kept. BiLSTM is broadly utilized in NLP tasks to learn contextual representations from phrases or sentences (Melamud *et al.*, 2016). Therefore, we use the words surrounding the recognized entities to train the contextual representations. As shown in Figure 1, W_a and W_b are recognized as two biological entities which can be a word or a phrase. The word embedding sequences $W_0 \dots W_a$ and $W_n \dots W_a$ are extracted from two directions as the inputs of a BiLSTM. In practice, we set up a window size u to normalize the sizes of two word sequences and use a dummy word to pad the sequence with length less than u. The inputs are then modified as followed.

$$\begin{aligned}\vec{x}_0 &= \langle X_0, X_1\rangle = \langle W_{a-u:a}, W_{a+u:a}\rangle \\ \vec{x}_1 &= \langle X_2, X_3\rangle = \langle W_{b-u:b}, W_{b+u:b}\rangle\end{aligned} \tag{1}$$

where $\vec{x}_i$ stands for the surrounding words of entity i, $W_{a:b}$ is the sequence of word embeddings from the ath to bth word. We adopt a closed boundary strategy to construct the contextual word sequences because the named entities itself may contain useful information to distinguish the argument context. As for the example mentioned in Section 2, the word 'promoters' itself indicates that it is probably an 'Agent' argument in the event of 'PromoterOf', since it is a general word that is also applicable to other entities. In contrast, the words 'cotB' and 'cotC' do not have any contribution to the context modeling, which will be forgotten in BiLSTM. Given the window size $u = 3$, the inputs for event promoters->cotB are $\vec{x}_0$ = [adheres, to, the, promoters; and, cotB, for, promoters]; $\vec{x}_1$ = [the, promoters, for, cotB; DummyWord, cotC, and, cotB]. The word streams are then input into LSTM cells. On the other hand, the output of BiLSTM is the concatenation output vector of the left-to-right LSTM and the right-to-left LSTM. In the above-mentioned example, the outputs of the BiLSTM layers are represented as LSTM([adheres, to, the, promoters]) concatenations with LSTM([and, cotB, for, promoters]) and LSTM([the, promoters, for, cotB]) concatenations with LSTM([PADDING, cotC, and, cotB]), where LSTM([...]) is the last output of the LSTM layer.

$$\begin{aligned}\text{BiLSTM}(X_0, X_1) &= \text{LSTM}(W_{a-u:a}) \oplus \text{LSTM}(W_{a+u:a}) \\ \text{BiLSTM}(X_2, X_3) &= \text{LSTM}(W_{b-u:b}) \oplus \text{LSTM}(W_{b+u:b})\end{aligned} \tag{2}$$

3.4 Argument embedding

We use multilayer perceptron (MLP) to train the argument classification model. As observed in Table 2, the skewed label distribution is a challenge for argument identification. We separate this multilabel classification problem into several binary classification problems under the one-versus-all strategy. Then we train each argument classifier separately using the estimator formulated in Equation (3). We use Dropout layer (Srivastava *et al.*, 2014) with dropout rate 0.2 in MLP as regularization to prevent overfitting.

$$\hat{y} = \text{MLP}(\vec{x}) = \text{Sigmoid}(F_2(\text{Tanh}(F_1(\text{BiLSTM}(\vec{x}))))) \tag{3}$$

where $F_i(x) = A_i x + b_i$ is a fully connected layer in MLP, Tanh is the hyperbolic tangent activation function and Sigmoid is the

activation function of the last layer of MLP. To tackle the imbalanced problem, we first estimate the distribution of the binary labels from the training dataset, and then use weighted binary cross-entropy in Equation (4) as the loss function to optimize the neural network model.

$$\text{loss} = -\sum_i (z y_i \log \hat{y}_i + (1-z)(1-y_i)\log(1-\hat{y}_i)) \tag{4}$$

where y_i is the true label of sample i and z represents the weight of positive class. The class weight z is estimated as $1-\frac{n}{N}$ with the number of positive class n and the total number of the samples N.

After VecEntNet is trained, we are able to extract the argument embedding R from the first layer of the MLP (Equation 5) for event detection. In particular, the triggers are actually embedded in the context of each argument. The trigger information, as well as their relations to the arguments, is encoded into the argument embedding for event detection.

$$R(\vec{x}) = F_1(\text{BiLSTM}(\vec{x})) \tag{5}$$

All possible pairs of recognized entities within each sentence are considered as candidate samples. For the classifier of an event type $e_i : \langle \arg_s, \arg_t \rangle$, we take the input $\vec{x}^*$ as the concatenation of both argument embeddings for each recognized entity of candidate pairs within one sentence [Equation (6)]. Since we are not aware of the true argument type for each entity, we use both embedding types with different orders for the entity pairs.

$$\vec{x}^* = \langle R_{\arg_s}(\vec{x}_0) \oplus R_{\arg_t}(\vec{x}_0), R_{\arg_t}(\vec{x}_1) \oplus R_{\arg_s}(\vec{x}_1) \rangle \tag{6}$$

VeComNet is designed for detecting the event types as well as the event direction of a candidate pair of recognized entities. To be consistent, we also build the multiclass classifiers under the one-versus-all strategy for event detection. For an event type e_i, we encode the direction $\arg_s \rightarrow \arg_t$ as 1 and others as 0. As a result, the label for a directed event type has two bits, in which one bit encodes the existence of this event type and another one encodes the direction. Therefore, the binary classification problem for each event type is transformed into a multilabel classification problem.

Similar to word vector, argument vector also possess the compositional attribute. To reflect the direction from the model, we use a subtract layer to combine the two input vectors as $\text{VeCom}(\vec{x}^*)$ [Equation (7)] and use it to predict the direction. The subtraction of the two argument vectors can be regarded as the multiplication of the concatenation of them and a factor matrix $\begin{bmatrix} I \\ -I \end{bmatrix}$, where I denotes the identity matrix. We explicitly multiply this factor matrix to conduct the vector composition before proceeding to the fully connected layer. In addition, the subtraction layer can decrease the number of neurons in the MLP, and thus its model generalization. As for the existence, we take the $L^1-\text{Norm}$ of $\text{VeCom}(\vec{x}^*)$ as input to another MLP for existence prediction.

$$\begin{aligned} \text{VeCom}(\vec{x}^*) = {} & R_{\arg_s}(\vec{x}_0) \oplus R_{\arg_t}(\vec{x}_0) \\ & - R_{\arg_t}(\vec{x}_1) \oplus R_{\arg_s}(\vec{x}_1) \end{aligned} \tag{7}$$

The resultant directed event estimator is demonstrated in Equations (8) and (9) representing the existence and direction respectively.

$$\begin{aligned} \hat{y}^* &= \text{MLP}^*(\text{Abs}(\text{VeCom}(\vec{x}^*))) \\ &= \text{Sigmoid}(F_2^*(\text{ReLU}(F_1^*(\text{Abs}(\text{VeCom}(\vec{x}^*)))))) \end{aligned} \tag{8}$$

$$\begin{aligned} \hat{y}' &= \text{MLP}'(\text{VeCom}(\vec{x}^*)) \\ &= \text{Sigmoid}(F_2'(\text{ReLU}(F_1'(\text{VeCom}(\vec{x}^*))))) \end{aligned} \tag{9}$$

where $F_i^*(x) = A_i^* x + b_i^*$ and $F_i'(x) = A_i' x + b_i'$ are fully connected layers, Abs is a layer for absolute value calculation, ReLU is the rectified linear unit activation function. Binary cross-entropy is adopted as loss function and stochastic gradient descent (SGD) is employed as the optimizer to train the classifiers for each event type.

4 Results

The training set and development set are combined to form an annotated dataset. We evaluated our method under 10-fold cross-validation. For the arguments or events in BioNLPST-BGI with less than 20 data instances, we changed to 5-fold cross-validation to ensure that the testing set would not have less than 2 classes. To ensure the training quality of those few labels, we randomly duplicated the samples in the training set so that the prediction model for each event type is trained on balanced data. Only the training samples were duplicated when training the argument embedding. The testing samples were neither duplicated nor used in argument embedding. We trained our models on a Linux machine equipped with a 32-core CPU and 32GB RAM. The hyperparameters used in the experiments are summarized in Supplementary Tables S1 and S2. Parameter analysis is also conducted to demonstrate the robustness of our method. The results shown in the Supplementary Materials indicate that our method are not sensitive to the hyperparameters.

4.1 Performance of VecEntNet and VeComNet during training

We use accuracy and mean-squared error to keep track of iterative training. As depicted in Figure 2 and Supplementary Figures S1–S9, VecEntNet converges roughly at the 10th epoch and keeps stable in the following training. Therefore, we use 20 epochs as the default hyperparameter in the subsequent experiments. Figure 2a shows that only the argument 'Gene' converges slower than others. Nevertheless, the overall performance of training VecEntNet and VeComNet is desirable.

4.2 Performance of VecEntNet and VeComNet under 10-fold cross-validation

We evaluate the overall performance with precision, recall and *F*-score under 10-fold cross-validation experiments. We can observe from Figure 3 and Supplementary Figures S10–S15 that VecEntNet performs very well in most of the argument classifications on BioNLPST-BGI. However, it is expected that VecEntNet can be underestimated on the tasks with limited training samples such as 'Entity', 'Gene' and 'Site'. Nevertheless, VeComNet achieves robust performance by leveraging the argument embedding of VecEntNet. As for the performance on BioNLPST-BB dataset shown in Figure 3c, we can observe that both VecEntNet and VeComNet can be scaled for enhanced performance once sufficient data are given. Our proposed model definitely performs well on balanced data but it is also applicable to imbalanced labels due to the weighted loss function proposed in VecEntNet. The detailed performance is tabulated in Supplementary Tables S3, S4 and Table 3.

Regarding the two worst cases of argument classification, 'Entity' and 'Gene' (*F*-scores = 0.15 and 0.37), their corresponding event detection is still satisfactory (*F*-scores = 0.97 and 0.76) as observed from Supplementary Table S3. We can also observe that the argument with better performance ('Site' and 'Promoter') within the same event type can compensate the weaknesses of the worse one.

4.3 Performance comparison with other top-ranked approaches

We compared our performance with that of the best method in the competition on BioNLPST-BGI dataset with respect to each event type. As tabulated in Table 4, VeComNet and the Uturku's approach Björne and Salakoski (2015) have their own merits on performence. VeComNet performs the best on 'Interaction', 'RegulonMember', 'SiteOf', 'TranscriptionBy' events with significant improvement on the *F*-scores (0.12, 0.32, 0.68, 0.4) compared to the best existing approach; and has competitive performance on 'RegulonDependence' and 'TranscriptionFrom' events. The performance of VeComNet on other events are stable where its average performance is better than the Uturku's approach. The method from Uturku seems to overfit the dataset since, in most of the event types,

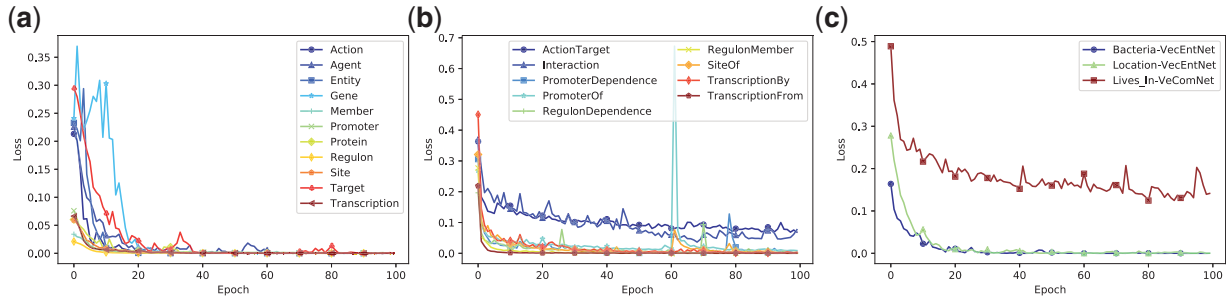

Fig. 2. Training loss of (**a**) VecEntNet as well as (**b**) VeComNet on BioNLPST-BGI dataset and those on (**c**) BioNLPST-BB. More details and high resolution version can be found in Supplementary Figures S1–S9

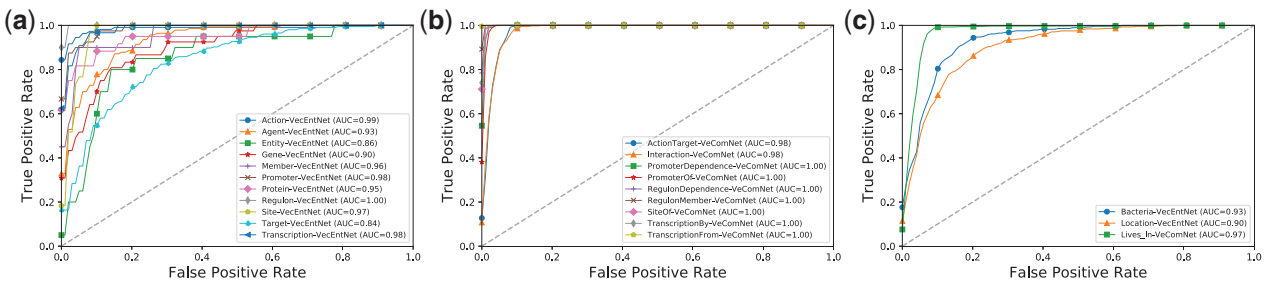

Fig. 3. Microaverage ROC curves of (**a**) VecEntNet as well as (**b**) VeComNet on BioNLPST-BGI and those on (**c**) BioNLPST-BB. More details and high resolution version can be found in Supplementary Figures S10–S15

Table 3. Performance of VecEntNet and VeComNet on BioNLPST-BB dataset

	VecEntNet		VeComNet
	Bacteria	Location	Lives_In
Accuracy	0.88	0.82	0.92
Precision	0.66	0.69	0.89
Recall	0.74	0.77	0.96
F-score	0.69	0.72	0.92
Train time (s)	771.44	757.76	4.83
Test time (s)	0.72	0.74	0.15

Table 4. Performance comparison between VeComNet and the best method (Uturku) on BioNLPST-BGI development dataset

Method	VeComNet			**Uturku** (Björne *et al.*, 2012)		
Event type	Precision	Recall	*F*-score	Precision	Recall	*F*-score
ActionTarget	0.7	0.91	0.79	0.94	0.92	0.93
Interaction	0.73	**0.82**	**0.77**	0.75	0.56	0.64
PromoterDependence	0.82	0.84	0.82	1.00	1.00	1.00
PromoterOf	0.79	0.78	0.76	1.00	1.00	1.00
RegulonDependence	0.97	0.99	0.98	1.00	1.00	1.00
RegulonMember	0.99	**0.99**	**0.99**	1.00	0.50	0.67
SiteOf	0.95	**0.98**	**0.97**	1.00	0.17	0.29
TranscriptionBy	**0.95**	**0.99**	**0.97**	0.67	0.50	0.57
TranscriptionFrom	0.99	0.99	0.99	1.00	1.00	1.00
Micro Average	0.823	**0.835**	**0.813**	0.91	0.83	0.79

Note: The digits in bold means it performs the best compared with other method in each metrics such as precision, recall, f1.

it achieved the ideal *F*-score of 1.0 while our proposed method does not. Our method stands out from other approaches because of its generalization ability.

Table 5. Performance comparison between VeComNet and other top-ranked methods (Delèger *et al.*, 2016) on BioNLPST-BB test dataset

Method	Precision	Recall	*F*-score
VeComNet	**0.89**	**0.96**	**0.92**
VERSE (Lever and Jones, 2016)	0.51	0.62	0.56
TurkuNLP (Mehryary *et al.*, 2016)	0.63	0.45	0.52
LIMSI	0.39	0.65	0.49
HK	0.60	0.39	0.47
whunlpre	0.56	0.41	0.47
DUTIR (Li *et al.*, 2016)	0.57	0.38	0.46
WXU	0.56	0.38	0.46

Note: The digits in bold means it performs the best compared with other method in each metrics such as precision, recall, f1.

From Table 5, we can observe that VeComNet has the strongest power in the single event prediction. The fewer the arguments and event types contained in the detection task, the more powerful VeComNet will be. Furthermore, VeComNet is a generic model that can be used in different event detection tasks without any tuning and modification. The robustness and predictive power of VeComNet enables it to be a promising model in the area of biomedical event extraction.

5 Case studies

To reveal how our method works, we randomly picked some cases from the testing dataset. The sample sentence 'The expression of rsfA is under the control of both sigma(F) and sigma(G).' with ID 'PMID-10629188-S5' in the testing dataset of BioNLPST-BGI has four recognized entities [T_1: 'expression', T_2: 'rsfA', T_3: 'sigma(F)', T_4: 'sigma(G)'] and three events (ActionTarget: [Action]$T_1 \rightarrow$ [Target]T_2, Interaction: [Agent]$T_3 \rightarrow$ [Target]T_2, Interaction: [Agent]$T_4 \rightarrow$ [Target]T_2) as ground true annotations. We obtained 11 argument models by fitting VecEntNet on the training dataset with the argument annotations. We further gained the argument

embeddings for each possible pair of entities [totally $n(n-1)$ pairs given n recognized entities in a sentence] in both training and testing datasets. For the above-mentioned sample, some of the candidate pairs generated are $\langle T_1, T_2\rangle, \langle T_2, T_1\rangle, \langle T_2, T_3\rangle, \langle T_3, T_2\rangle, \langle T_2, T_4\rangle, \langle T_4, T_2\rangle, \langle T_1, T_3\rangle, \langle T_1, T_4\rangle$. The argument models for event type ActionTarget are $\text{arg}_{\text{action}}$ and $\text{arg}_{\text{target}}$. We take them as functions and the candidate pairs of entities as input. The argument embeddings we obtained for $\langle T_1, T_2\rangle$ are $\langle \text{arg}_{\text{action}}(T_1) \oplus \text{arg}_{\text{target}}(T_1), \text{arg}_{\text{target}}(T_2) \oplus \text{arg}_{\text{action}}(T_2)\rangle$. Since we are not aware of the argument type, the entities belong to, we concatenated both argument embeddings for each entity and let VeComNet to determine. The argument embeddings are obtained for other candidate entity pairs with respect to different event types in a similar way. We used argument embeddings as the input of VeComNet models. The predicted labels for the aforementioned candidate entity pairs are $\langle 1,1\rangle, \langle 1,0\rangle, \langle 0,0\rangle \ldots \langle 0,0\rangle$ with respect to ActionTarget event and $\langle 0,0\rangle, \langle 0,0\rangle, \langle 1,0\rangle, \langle 1,1\rangle, \langle 1,0\rangle, \langle 1,1\rangle, \langle 0,0\rangle, \langle 0,0\rangle$ with respect to Interaction event, in which the first label indicates the existence of the corresponding event and the second label indicates whether the event is pointed from the first entity to the second one. The binary labels were further post-processed to generate the predicted biomedical events. For instance, the candidate pairs $\langle T_1, T_2\rangle$ and $\langle T_2, T_3\rangle$ are predicted as $\langle 1,1\rangle$ and $\langle 1,0\rangle$ for ActionTarget and Interaction events, respectively. It means that it exists an ActionTarget event $T_1 \rightarrow T_2$ (expression->rsfA) and an Interaction event $T_3 \rightarrow T_2$ [rsfA->sigma(F)] in this sentence.

6 Discussion

For many years, scientific literature has served as the major outlet for novel discovery and result dissemination. To extract useful knowledge from the literature for downstream management and query tasks, IE is proposed to automate this process. Biomedical event extraction is fundamentally important because it is able to systematically organize the knowledge as controlled representations such as directed knowledge graphs. However, the existing event detection methods are not satisfactory in performance because most of them are constrained in the trigger-based approach which relies on the lexical and syntactic features extracted from dependency parsing. The quality of manual trigger annotation and the error propagation from trigger detection to the event detection have limited our progress for years.

In this study, we proposed a bottom-up event detection framework using deep learning techniques. We built an LSTM-based model VecEntNet to construct argument embeddings for each recognized entity. We further utilized the compositional attributes of the argument vectors to train a directed event classifier VeComNet.

LSTM and context embedding have shown its applicability in several NLP tasks. Our main contribution is the proposed framework for argument embedding using BiLSTM and the downstream directed event detection using multioutput neural network. This strategy for event detection is proposed for the first time. It overcomes the error propagation as well as the extra annotations of trigger-based approaches. Besides, the continuous space of argument embedding significantly lessen the sensitivity of event detection. In addition, we developed our own loss functions for training the argument embedding with unbalanced data and training the multioutput neural network for directed event detection. These are the key reasons why our method can achieve outstanding performance. Broadly speaking, the proposed method is suitable for general event extraction by using the pretrained word embedding in the specific area. Assumed the entities are correctly recognized, all the possible pairs of entities within a predefined scope (i.e. sentence or abstract) will be considered for the events. Besides the ones that could be easily filtered by the constraints (i.e. possible entity types that can be marked as a specific argument type within each kind of event) defined in the tasks, the remaining candidate entity pairs still contain numerous negative samples. Balancing the training samples and improving the performance of event prediction are the inherent difficulties for biomedical event extraction. The experimental results show that our method works well on the two datasets BioNLPST-BGI and BioNLPST-BB which are given in sentence level and abstract level, respectively. However, we have not evaluated it on the full-text level, which may be the main limitation.

Our method is not sensitive to the hyperparameters and it works well for a wide range of instances. The results indicate that the proposed method is competent in the biomedical event extraction. In the future, we envision that it can fundamentally benefit the related downstream tasks in biomedical text mining with broad impacts.

Acknowledgements

The authors are grateful to the organizers of BioNLP Shared Task, who provided the public annotated dataset. We thank the reviewers for their time. In particular, we thank the first reviewer for his/her careful and thoughtful comments which have improved the manuscript reader friendliness in a significant manner. The authors also thank Prashant Sridhar for his English proofreading.

Funding

The work described in this article was substantially supported by three grants from the Research Grants Council of the Hong Kong Special Administrative Region: [CityU 21200816], [CityU 11203217] and [CityU 11200218]. We acknowledge the donation support of a Titan Xp GPU from the NVIDIA Corporation.

Conflict of Interest: none declared.

References

Ananiadou,S. *et al*. (2010) Event extraction for systems biology by text mining the literature. *Trends Biotechnol*., **28**, 381–390.

Bengio,Y. *et al*. (2003) A neural probabilistic language model. *J. Mach. Learn. Res*., **3**, 1137–1155.

Björne,J. and Salakoski,T. (2011) Generalizing biomedical event extraction. In: *Proceedings of the BioNLP Shared Task 2011 Workshop, Portland, Oregon, USA*, pp.183–191.

Björne,J. and Salakoski,T. (2015) TEES 2.2: biomedical event extraction for diverse corpora. *BMC Bioinformatics*, **16 (Suppl. 16)**, S4.

Bjorne,J. *et al*. (2010) Complex event extraction at PubMed scale. *Bioinformatics*, **26**, i382–i390.

Björne,J. *et al*. (2012) University of Turku in the BioNLP'11 Shared Task. *BMC Bioinformatics*, **13 (Suppl. 11)**, S4.

Bui,Q.-C. and Sloot,P.M.A. (2012) A robust approach to extract biomedical events from literature. *Bioinformatics*, **28**, 2654–2661.

Cañada,A. *et al*. (2017) LimTox: a web tool for applied text mining of adverse event and toxicity associations of compounds, drugs and genes. *Nucleic Acids Res*., **45**, W484–W489.

Chen,D. and Manning,C. (2014) A fast and accurate dependency parser using neural networks. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar*, pp.740–750.

Delèger,L. *et al*. (2016) Overview of the bacteria biotope task at BioNLP Shared Task 2016. In: *Proceedings of the 4th BioNLP Shared Task Workshop, Berlin, Germany*, pp.12–22.

Gers,F. (1999) Learning to forget: continual prediction with LSTM. In: *9th International Conference on Artificial Neural Networks: ICANN'99, Edinburgh, UK*, 1999, pp. 850–855. IEE.

Jagannatha,A.N. and Yu,H. (2016) Bidirectional RNN for medical event detection in electronic health records. In: *Proceedings of the Conference, Association for Computational Linguistics, San Diego, California, USA. North American Chapter, Meeting*, 2016, pp. 473–482.

Kim,J.-D. *et al*. (2009) Overview of BioNLP'09 Shared Task on event extraction. In: *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task, Boulder, Colorado, USA*, pp. 1–9. Association for Computational Linguistics.

Kim,J.-D. *et al*. (2011) Overview of BioNLP Shared Task 2011. In: *Proceedings of the BioNLP Shared Task 2011 Workshop, Portland, Oregon, USA*, pp.1–6. Association for Computational Linguistics.

Kosmopoulos,A. *et al*. (2015) Biomedical semantic indexing using dense word vectors in BioASQ. *J. BioMed. Semant. Suppl. BioMedl. Inf. Retr*., **3410**, 959136040–1510456246.

Lever,J. and Jones,S.J.M. (2016) VERSE: Event and relation extraction in the BioNLP 2016 Shared Task. In: *Proceedings of the 4th BioNLP Shared Task Workshop, Berlin, Germany*, pp. 42–49.

Li,H. *et al*. (2016) DUTIR in BioNLP-ST 2016: utilizing convolutional network and distributed representation to extract complicate relations. In: *Proceedings of the 4th BioNLP Shared Task Workshop, Berlin, Germany*, pp. 93–100.

Lim,K.M.K. *et al*. (2016) @MInter: automated text-mining of microbial interactions. *Bioinformatics*, **32**, 2981–2987.

Luo,Y. *et al*. (2017) Bridging semantics and syntax with graph algorithms – state-of-the-art of extracting biomedical relations. *Brief. Bioinform*., **18**, 160–178.

Mallory,E.K. *et al*. (2016) Large-scale extraction of gene interactions from full text literature using DeepDive. *Bioinformatics*, **32**, 106–13.

McClosky,D. and Charniak,E. (2008) Self-training for biomedical parsing. In: *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers, Columbus, Ohio, USA*, pp. 101–104. Association for Computational Linguistics.

Mehryary,F. *et al*. (2016) Deep learning with minimal training data: TurkuNLP entry in the BioNLP Shared Task 2016. In: *Proceedings of the 4th BioNLP Shared Task Workshop, Berlin, Germany*, pp. 73–81.

Melamud,O. *et al*. (2016) context2vec: learning generic context embedding with bidirectional lstm. In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, Berlin, Germany*, pp. 51–61.

Mikolov,T. *et al*. (2013) Distributed representations of words and phrases and their compositionality. In: Burges,C.J.C., Bottou,L., Welling,M., Ghahramani,Z. and Weinberger,K.Q. (eds.) *Advances in Neural Information Processing Systems*. Vol. 26. Curran Associates, Inc., pp. 3111–3119.

Nédellec,C. *et al*. (2013) Overview of BioNLP shared task 2013. In: *Proceedings of the BioNLP Shared Task 2013 Workshop, Sofia, Bulgaria*, pp. 1–7.

Nédellec,C. *et al*. (2016) In: *Proceedings of the 4th BioNLP Shared Task Workshop, Berlin, Germany*. Association for Computational Linguistics. Available at: https://www.aclweb.org/anthology/papers/W/W16/W16-3000/.

Nguyen,T.H. *et al*. (2016) Joint event extraction via recurrent neural networks. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, California, USA*, pp. 300–309.

Nivre,J. *et al*. (2016) Universal dependencies v1: a multilingual treebank collection. In: *LREC, Portorož, Slovenia*.

Perfetto,L. *et al*. (2016) SIGNOR: a database of causal relationships between biological entities. *Nucleic Acids Res*., **44**, D548–D554.

Pyysalo,S. *et al*. (2012) Event extraction across multiple levels of biological organization. *Bioinformatics*, **28**, i575–i581.

Rebholz-Schuhmann,D. *et al*. (2012) Text-mining solutions for biomedical research: enabling integrative biology. *Nat. Rev. Genet*., **13**, 829–839.

Srivastava,N. *et al*. (2014) Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res*., **15**, 1929–1958.

Zhao,Z. *et al*. (2016) Drug drug interaction extraction from biomedical literature using syntax convolutional neural network. *Bioinformatics*, **32**, 3444–3453.

Zhou,D. *et al*. (2014) Event trigger identification for biomedical events extraction using domain knowledge. *Bioinformatics*, **30**, 1587–1594.