BIOINFORMATICS ORIGINAL PAPER

Vol. 30 no. 11 2014, pages 1587–1594
doi:10.1093/bioinformatics/btu061

Data and text mining

Advance Access publication January 30, 2014

Event trigger identification for biomedical events extraction using domain knowledge

Deyu Zhou[1,*], Dayou Zhong[1] and Yulan He[2]

[1]School of Computer Science and Engineering, Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, Nanjing 210096, China, and [2]School of Engineering and Applied Science, Aston University, Birmingham B4 7ET, UK

Associate Editor: Jonathan Wren

ABSTRACT

Motivation: In molecular biology, molecular events describe observable alterations of biomolecules, such as binding of proteins or RNA production. These events might be responsible for drug reactions or development of certain diseases. As such, biomedical event extraction, the process of automatically detecting description of molecular interactions in research articles, attracted substantial research interest recently. Event trigger identification, detecting the words describing the event types, is a crucial and prerequisite step in the pipeline process of biomedical event extraction. Taking the event types as classes, event trigger identification can be viewed as a classification task. For each word in a sentence, a trained classifier predicts whether the word corresponds to an event type and which event type based on the context features. Therefore, a well-designed feature set with a good level of discrimination and generalization is crucial for the performance of event trigger identification.

Results: In this article, we propose a novel framework for event trigger identification. In particular, we learn biomedical domain knowledge from a large text corpus built from Medline and embed it into word features using neural language modeling. The embedded features are then combined with the syntactic and semantic context features using the multiple kernel learning method. The combined feature set is used for training the event trigger classifier. Experimental results on the golden standard corpus show that >2.5% improvement on F-score is achieved by the proposed framework when compared with the state-of-the-art approach, demonstrating the effectiveness of the proposed framework.

Availability and implementation: The source code for the proposed framework is freely available and can be downloaded at http://cse.seu.edu.cn/people/zhoudeyu/ETI_Sourcecode.zip.

Contact: d.zhou@seu.edu.cn

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

Received on October 1, 2013; revised on January 23, 2014; accepted on January 24, 2014

1 INTRODUCTION

In molecular biology, molecular events describe observable alterations of biomolecules, such as binding of proteins or RNA production. These molecular events influence the formation of a phenotype, which may be responsible for drug reactions or development of certain diseases. However, knowledge about these events is scattered in the scientific literature with continuing fast growth. Tremendous systematic and automated efforts are required to use the underlying information. As such, biomedical event extraction attracted much research interest recently. Several evaluation tasks, such as BioNLP'09 (Kim *et al.*, 2009), BioNLP'11 (Kim *et al.*, 2012) and BioNLP'13 (Nédellec *et al.*, 2013) shared tasks, have been held in recent years to allow researchers to develop and compare their methods for biomedical events extraction.

In general, each biomedical event consists of a trigger and one or more arguments. For example, '...inhibiting tyrosine phosphorylation of STAT6...' describes two events, one is the phosphorylation event and the other is the negative regulation event, which is signaled by the word 'inhibiting' and takes the first phosphorylation event as its argument. In a typical biomedical event annotation, these two events are represented as follows:

E1 (Event Type:Phosphorylation, Theme:STAT6, ToLoc: tyrosine)

E2 (Event Type: Negative_regulation:inhibiting Theme:E1)

Biomedical event extraction aims to extract such event information from biomedical literature and reformats this extracted information in structures as represented by the two annotations presented above. By extracting detailed behaviors of biomolecules, biomedical event extraction can be used to support the development of biomedical-related databases.

To extract events from texts, most systems rely on a pipeline procedure, which usually consists of three cascaded modules including biomedical term identification, event trigger identification and event argument detection (Zhou and He, 2011). In such pipeline-based approaches, it is crucial to identify event triggers reliably, as errors in an early stage will be propagated and hurt the performance of the subsequent module. Analysis on the event extraction results show that >60% of extraction errors are attributed to the errors of event trigger identification (Pyysalo *et al.*, 2012). To achieve a better performance, existing approaches to event trigger identification are mostly based on learning classifiers from annotated data instead of using manually constructed dictionaries containing a list of trigger words or manually defined linguistic rules. In such approaches, event types are treated as classes, and the aim is to classify words in sentences as indicating a particular event type or not by taking the context features including the syntactic and semantic features into account.

*To whom correspondence should be addressed.

Table 1. The sentences in the MLEE Corpus in which 'hydrolysis' was annotated as an event trigger

Sentences
1 Angiostatin inhibits both ATP synthesis and ATP hydrolysis (Moser *et al*., 2001) and interferes with intracellular pH regulation (Wahl and Grant, 2002; Wahl *et al*., 2002).
2 Our data suggest that VEGFR2-mediated regulation of endothelial function is dependent on different, but specific, Rab-mediated GTP hydrolysis activity required for endosomal trafficking.

Table 2. Examples of the similar contexts where the two words 'proteolysis' and 'hydrolysis' occur in Medline

Hydrolysis	Proteolysis
An increase of the products of *casein* hydrolysis, the *proteose–peptone* (p–p) fraction and *minor (m) caseins*	Use of indices of proteolysis of *caseins* such as the *proteose–peptone, m-casein* and PI
AApeptides are *resistant* to enzymatic hydrolysis	With the ease of *resistance* to proteolysis, the development of sequence-specific *AApeptides*...

However, such approaches rely on abundant annotated training data and may not work well when certain event instances are rare in the training data. For example, the word 'proteolysis' does not occur as an event trigger for catabolism type in the training data of the multi-level event extraction (MLEE) corpus (Pyysalo *et al*., 2012). Therefore, it is difficult to recognize it as an event trigger in the sentence 'The effects of IGF-1 are mediated principally through the IGF-1R but are modulated by complex interactions with multiple IGF binding proteins that themselves are regulated by phosphorylation, **proteolysis**, polymerization, and cell or matrix association', which appears in the test set. Nevertheless, we notice that another word 'hydrolysis' was annotated as an event trigger in the training data as shown in Table 1. If we search through Medline (http://www.ncbi.nlm.nih.gov/entrez/query/static/overview.html), we can find that the two words 'proteolysis' and 'hydrolysis' occur in similar context thus tend to have similar meanings following the distributional hypothesis (Harris, 1970). Examples of the similar context where 'proteolysis' and 'hydrolysis' occur in Medline are presented in Table 2. If we can learn such domain knowledge and incorporate it into trigger word identification, then 'proteolysis' might be correctly identified as an event trigger even if it did not appear in the training data at all.

In this article, we argue that biomedical domain knowledge, such as words, tends to occur in similar context, is highly related and this can be incorporated into the learning process of the event trigger classifier to improve the performance of trigger word identification. In specific, we propose a novel framework to learn biomedical knowledge from a large text corpus built from Medline and embed it into word features using neural language modeling. The embedded features are further combined with the well-designed syntactic and semantic context features using the multiple kernel learning (MKL) method for classifier training. We conducted extensive experiments on the MLEE corpus (Pyysalo *et al*., 2012), and the results show that >2.5% improvement on F-score is achieved using the proposed framework when compared with the state-of-the-art approach, demonstrating the effectiveness of the proposed framework.

The rest of the article is organized as follows. Section 2 presents the proposed framework, which consists of three steps, domain knowledge embedding, local context features extraction and MKL. Experimental setup and results are discussed in Section 3. Finally, Section 4 concludes the article.

2 OUR APPROACH

Our proposed framework for event trigger identification works as follows, which is illustrated in Figure 1. First, scientific publications from Medline are crawled to form a corpus where domain knowledge can be obtained. Then a neural language model is built from such a corpus using unsupervised learning. The distributional representation for each word is induced as the feature of the word (word embedding). Then, for sentences in the training and testing datasets, protein name identification, syntactic parsing and dependency parsing are performed and local context features are extracted from the parsing results. After that, features induced by neural language model and features extracted from syntactic and dependency parsing results are combined through MKL. Finally, training and testing are conducted on the combined feature set.

In what follows, we first describe how to formulate the task of event trigger identification as a classification problem, in which two sets of features, domain knowledge embedding and local context features, are used. Then, we present how to learn the parameters of our proposed unified classification framework using MKL. Finally, we discuss how the two feature sets can be constructed.

2.1 Problem definition

Event trigger identification in the biomedical domain can be seen as the task of assigning labels to words. Existing approaches for event trigger identification typically rely on annotated training data where those event trigger words are labeled with their corresponding event types. A rich set of manually designed features are then extracted from annotated sentences and fed into a classification algorithm such as support vector machines (SVMs) for training. In our approach here, we adopt a similar procedure of training a classifier from annotated data for trigger word identification. However, apart from the annotated training data, we additionally crawled articles from Medline to form a corpus where domain knowledge can be extracted.

Given sentences $\mathcal{S} = \{s_i : w_{i1}w_{i2}...w_{in_i}, i = 1...L\}$, their corresponding trigger annotations $\mathcal{T} = \{t_i : a_{i1}a_{i2}...a_{in_i}, i = 1...L\}$ and an additional unannotated corpus where domain knowledge can be extracted, $\mathcal{S}^u = \{s_i : w_{i1}w_{i2}...w_{in_i}, i = (L+1), (L+2)...(L+UL)\}$ where L and UL are the numbers of sentences in the training data and the domain corpus, respectively, the objective is to estimate a hypothesis $f : \mathcal{S} \mapsto \mathcal{T}$ minimizing the prediction error on unseen data. For traditional machine learning

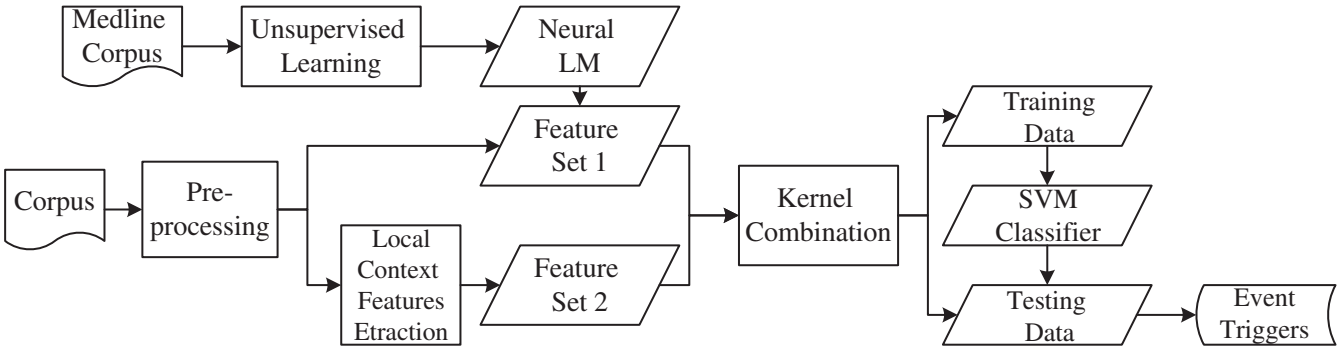

Fig. 1. The system architecture of our proposed framework for event trigger identification

approaches, f is determined by minimizing the loss between the prediction $f(\Phi(w))$ for the training instance w and its actual label a_w based on some loss function *Loss*. Here $\Phi(w)$ is the feature set related to w. As will be shown in Section 2.4, local context around w is used for constructing $\Phi(w)$. Moreover, to make sure that words occurring in similar contexts share the same class label, the loss between the prediction $f_2(\Psi(w'))$ of w' and a_w is also minimized, where w' is the word that is found to have highest contextual similarity with w from in the domain corpus and $\Psi(w')$ is another type of feature set related to w'. As will be shown in Section 2.3, contextually similar words can be modeled using neural language modeling. Because w' is the word with the highest contextual similarity with w, $\Psi(w')$ can be approximated as $\Psi(w)$. Our final objective function is

$$\hat{f} = \underset{f=(f_1,f_2)\in\Theta}{\operatorname{argmin}} \sum_{w} (Loss(f_1(\Phi(w)), a_w) + rLoss(f_2(\Psi(w)), a_w)) \quad (1)$$

where r is a parameter controlling the trade-off between two losses. When $r=0$, Equation (1) reduces to the object function for classification based on local context features only.

2.2 Parameter learning

In our framework, we use the local context features derived from the annotated training data for Φ and use word embeddings induced from the domain corpus using neural language modeling for Ψ. We use SVM for both f_1 and f_2 and $f_1 = \langle \mathbf{w_1}, \Phi(w)\rangle + b_1$ and $f_2 = \langle \mathbf{w_2}, \Psi(w)\rangle + b_2$. Therefore, the above problem can then be solved by optimizing the parameters of $\mathbf{w_1}, \mathbf{w_2}, b_1, b_2, r$. However, these parameters cannot be optimized directly using the general learning approach for SVM. By considering parameters optimization as learning the optimal weights of different types of features from the data automatically, the problem is converted into to feature combination. Under kernel learning, feature combination is translated into kernel combination by defining two kernels K_1, K_2 based on $\Phi(w), \Psi(w)$. There are many possible ways for kernel combination. A simplest one is to average several kernels by setting $r=1$.

In our work here, we use MKL (Bach *et al.*, 2004), which has been shown to produce good results in object classification in computer vision (Gehler and Nowozin, 2009). The aim of MKL is to learn a kernel combination during the training phase of the algorithm by optimizing jointly over a linear combination of kernels $\sum_{i=1}^{m} \beta_i K_i(w, w')$ and the parameters of an SVM, where m is the number of kernels to be combined. Under MKL, the object function described in Equation (1) is changed to

$$\min_{\alpha,\beta,b} \frac{1}{2} \sum_{i=1}^{2} \beta_i \alpha^T K_i \alpha + C \sum_{j=1}^{N} L(a_{w_j}, b + \sum_{i=1}^{2} \beta_i K_i(w)^T \alpha) \quad (2)$$

where N is the number of training instances, C is a predefined positive trade-off parameter between model simplicity and classification error, typically used in SVMs, $\alpha = (\alpha_1, ..., \alpha_N)^T$ is the vector of dual variables corresponding to each separation constraint, $K_1(w) = (\langle \Phi(w_1), \Phi(w)\rangle, ..., \langle \Phi(w_N), \Phi(w)\rangle)^T$, $K_2(w) = (\langle \Psi(w_1), \Psi(w)\rangle, ..., \langle \Psi(w_N), \Psi(w)\rangle)^T$, $K_1 = (\langle \Phi(w_i), \Phi(w_j)\rangle)_{N\times N}$, $K_2 = (\langle \Psi(w_i), \Psi(w_j)\rangle)_{N\times N}$ and $L(a_{w_j}, t) = \max(0, 1 - a_{w_j} t)$ is the hinge loss. For efficiency and interpretability, the objection function subjects to

$$\beta_1 + \beta_2 = 1, \beta_1 \geq 0, \beta_2 \geq 0 \quad (3)$$

Here, β_1 and β_2 are the weighting of two features set. The problem can be solved using the SimpleMKL (Rakotomamonjy *et al.*, 2008) Toolbox (http://asi.insa-rouen.fr/enseignants/~arakotom/code/mklindex.html). The decision function is of the following form,

$$\text{sign}(\sum_{i=1}^{2} \beta_i (K_i(x)^T \alpha + b)) \quad (4)$$

2.3 Word embeddings learned by neural language modeling

We use neural language modeling (Huang *et al.*, 2012) to learn word representations by discriminating the next word given its local context and global context. Given a word sequence $s_i = (w_{i1}, w_{i2}, ..., w_{in})$ and a document $d_j = (w_{j1}, w_{j2}, ..., w_{jm})$, which contain s_i, the goal of the model is to discriminate the w_{in} (the correct one) from a random word w. Thus, the object function of the model is to minimize the ranking loss for each (s_i, d_j):

$$\sum_i \sum_j \sum_{w\in V\setminus w_{in}} \max(0, 1 - f(s_i, d_j) + f(s_i^w, d_j)), \quad (5)$$

where $s_i^w = w_{i1}, w_{i2}, ..., w_{i,n-1}, w$ is the sequence by changing the last word w_{in} into w. The dataset for learning the language model can be constructed by considering all the word sequences in the Medline corpus. Positive examples are the word sequences from

Medline, whereas negative examples are the same word sequence with the last word replaced by a random one.

Instead of using only local context for language model learning, document context (or global context) is also considered. Thus, the score function $f(s_i, d_j)$ is replaced by two functions, $\text{score}^l(s_i, d_j)$ and $\text{score}^g(s_i, d_j)$, which are defined to capture local context and global context, respectively.

The score function of local context $\text{score}^l(s_i, d_j)$ is calculated by a neural network with one hidden layer:

$$a_1^l = g(W_1^l X^l + b_1^l) \tag{6}$$

$$\text{score}^l = W_2^l a_1^l + b_2^l \tag{7}$$

where $X^l = [x_1, x_2, ..., x_n]$ is the concatenation of the n word embeddings representing sequence s_i, g is an element-wise activation function such as *tanh*, a_1^l is the activation of the hidden layer with h^l hidden nodes, W_1^l and W_2^l are the first and second layer weights of the neural network, respectively, and b_1^l, b_2^l are the biases of each layer.

The score function of global context $score^g(s_i, d_j)$ is calculated by a two-layer neural network:

$$a_1^g = h(W_1^g X^g + b_1^g) \tag{8}$$

$$\text{score}^g = W_2^g a_1^g + b_2^g \tag{9}$$

where

$$X^g = [\frac{\sum_{t=1}^{m} \phi(w_{jt}) X_t}{\sum_{t=1}^{m} \phi(w_{jt})}, x_n]$$

which is the weighted average of all word vectors in the document d_j, ϕ is a weighting function describing the importance of word w_{jt} in the document d_j, h is an element-wise activation function such as *tanh*, $a_1^g \in \mathbb{R}^{h^g \times 1}$ is the activation of the hidden layer with h^g hidden nodes, W_1^g and W_2^g are the first and second layer weights of the neural network, respectively, and b_1^g, b_2^g are the biases of each layer. The local score preserves word order and syntactic information, whereas the global score uses a weighted average that is similar to bag-of-words features, capturing more of the semantics and topics of the document.

The gradient of the objective is sampled by randomly choosing a word from the vocabulary as a corrupted example for each sequence–document pair (s_i, d_j). The derivative of the ranking loss is taken with respect to the parameters and these weights are updated via backpropagation.

2.4 Local contexts features

The syntactic and semantic features used in the framework are generated from the outputs of GDep (a dependency parser) (Sagae and Tsujii, 2007) and Enju parser (a syntactic parser) (Miyao and Tsujii, 2008).

All the features used in the framework are extracted based on (Pyysalo *et al.*, 2012), described as follows:

- *Lexical and syntactic features of the word itself*. The features such as whether the word has a capital letter, whether it is at the beginning of the sentences, whether it has a number, whether it has a symbol, whether it is in a trigger word dictionary, whether it is in a protein base form, its POS tag and n-grams of characters ($n = 2, 3, 4$) are extracted. For features like whether it has certain property, boolean value is used for the feature value. In addition, to check whether a word is in the trigger word dictionary, we constructed a dictionary by collecting all the trigger words from the training set. Triggers that contain more than one word are filtered. Also, hyphenated compound words are added into the dictionary if one of its words already appears in the trigger word dictionary.
- *Local context features*. For the sequence of three words before or after the candidate word, n-grams ($n = 1, 2, 3, 4$) are used. For example, for the word 'retarget' in the sentence 'The binding of I kappa B/MAD-3 to NF-kappa B p65 is sufficient to retarget NF-kappa B p65 from the nucleus to the cytoplasm', the word sequence 'is sufficient to retarget protein from the' is used to generate the relevant n-grams. Also, each word is represented by its base form, the POS tag and the relative position (before or after) to the target word.
- *Local dependency features*. The two-depth path started from the candidate word in the dependency tree generated from the GDep parser is identified first. Features are then extracted from the path such as n-grams ($n = 2$) of dependencies, n-grams ($n = 2, 3$) of words represented by their base forms and the POS tags and n-grams ($n = 2, 3, 4$) of dependencies and words. For word tokens not having two-depth paths, such as the root node or the direct children of the root node, these types of features are ignored. N-grams ($n = 2$) of dependencies are represented as dependency1–dependency2. Similarly, n-grams ($n = 2, 3$) of words or n-grams ($n = 2, 3, 4$) of dependencies and words are represented as word1–word2–word3 or word1–dependency1–word2 and so on. For example, for the word 'retarget' in the sentence 'the binding of I kappa B/MAD-3 to NF-kappa B p65 is sufficient to retarget NF-kappa B p65 from the nucleus to the cytoplasm.', its two-depth path 'retarget → AMOD → sufficient → PRD → is' can be retrieved from the GDep parsing results. Its n-grams ($n = 2$) of dependencies are given as 'AMOD PRD'.
- *Shortest path features*. The shortest path, a directed path between the candidate and the closest protein, is also retrieved from the dependency parse generated from GDep parser. The vertex walks, edge walks, n-grams ($n = 2, 3, 4$) of dependencies, n-grams ($n = 2, 3, 4$) of words represented as base forms plus POS tags and the length of path are extracted as the path features. For example, for the word 'retarget' in the sentence 'The binding of I kappa B/MAD-3 to NF-kappa B p65 is sufficient to retarget NF-kappa B p65 from the nucleus to the cytoplasm.', its shortest path is 'retarget← OBJ ←protein'. The length of path that is 1, edge walks as retarget← OBJ ←protein, vertex walks as OBJ can be extracted. The reason of using shortest path is that a candidate and its closest proteins are much more likely to be involved in a biomedical event. Thus, features extracted from the shortest path should be useful for detecting triggers in biomedical event extraction.

3 EXPERIMENTS

In this section, we present our experiments to evaluate the effectiveness of the proposed framework. We will first discuss results obtained on event trigger identification in comparison with the best performance obtained so far. We will then present performance comparison results with or without using the MKL method for comparison, followed by the results of using neural language models trained under different corpora. Finally, we compare our results with those obtained using the latent Dirichlet allocation (LDA) model as another distributional semantics approach instead of neural language modeling.

3.1 Experimental setup

We used MLEE corpus for our experiments on trigger words identification. Instead of focusing exclusively on molecular-level entities and process, MLEE corpus is extended to encompass all levels of biological organization from the molecular to the whole organism. The corpus is generated from 262 PubMed abstracts on angiogenesis, which involves a tissue/organ-level process closely associated with cancer and other organism-level pathologies. Texts in that domain represent a good test case for event extraction across multiple levels of biological organization. The annotation follows the guideline formalized in the BioNLP 2009 Shared Task on event extraction. In this guideline, events are *n*-ary associations of participants (entities or other events) with specific role such as theme and cause. Each event is assigned a type from a fixed set defined for the task (e.g. binding and phosphorylation) and is associated with a specific span of text stating the event, termed the event trigger. The events are categorized as four groups such as 'ANATOMICAL', 'MOLECULAR', 'GENERAL' and 'PLANNED', which are further classified into 19 classes. These 19 classes are the target classes of our trigger word classifier. It is worth noting that we used a combination of training and development datasets of the MLEE corpus for training, and the test set for testing.

To train a neural language model, we additionally built a corpus from Medline because of its wide coverage of topics in the biomedical domain. Abstracts of biomedical literature published in 2011 and 2012 were retrieved to build the corpus.

All the sentences in the Medline corpus were preprocessed such as lowercasing and stemming. We chose the most frequent words in the corpus to construct vocabularies with different size $D = \{15,000, 30,000, 60,000, 90,000\}$. Words starting with a digital number are mapped to the 'NUMBER' token. Words starting with a special character are mapped to the 'UNUSUAL' token. Other rare words not in the dictionary are replaced with the 'UNKNOWN' token. For neural language model training, we used 50 dimensional embeddings and set the number of hidden units to 100.

3.2 Experimental results

This section presents the evaluation results in details. In our framework, the one-versus-rest SVMs are used for trigger word classification. To alleviate the unbalanced classification problem, we boosted the positive examples by placing more weights on them during training.

Table 3. Comparison of the performance of event trigger identification

Method	Recall (%)	Precision (%)	F-score (%)
Baseline	81.69	70.79	75.84
Proposed	81.29	75.56	78.32

3.2.1 Event trigger identification results We implemented a baseline following the approach proposed in (Pyysalo *et al.*, 2012), which achieved the state-of-the-art performance on trigger word identification using the features extracted from the syntactic and semantic parsing results as described in Section 2.4. We conducted experiments on the MLEE corpus and compared our framework with the baseline approach. Table 3 lists the recall, precision and F-score obtained on the test set of the MLEE corpus. In the results reported here, we trained a neural language model on the Medline corpus with the vocabulary size of 30 000. Using the features induced from the learned neural language model, the performance of event trigger identification is improved significantly with ~5% on precision. The overall improvement on F-score is ~2.5%. To further investigate how the improvement is achieved, we analyzed the experimental results of the baseline approach and the proposed framework. We found that positive instances identified correctly by the baseline approach are still identified correctly by the proposed framework in 97.8% of cases. Out of the false-negative instances identified by the baseline, 7.8% were correctly identified as positive instances by our framework.

To further study the difference of our proposed framework against the existing state-of-the-art approach in different event categories, we list the detailed results in each event category in Table 4. It can be observed from the table that of 19 event types, our proposed framework outperforms the baseline approach on 13 event types and gives almost identical results on another 5 event types. To investigate the performance improvement under different event types, we analyze the relationship between performance improvement and the size of the training data in each event category. The results are illustrated in Figure 2. It can be observed that the performance improvement decreases when the size of the training data increases. The largest improvement (100%) is achieved in the 'dephosphorylation' event type when there are only five training instances. Our approach successfully identified the 'dephosphorylation' event triggers in all three instances in the test set, while the baseline approach failed to identify any of them. When the training data are relatively abundant, our approach appears to have less improvement compared with the baseline.

From the above observations, we can speculate that our proposed framework with domain knowledge incorporated is particularly effective when facing with scarce training data. The only exception is the 'transcription' event type with 30 training instances for which the baseline identified one event trigger correctly from the test set, while our approach failed to recognize any. It is shown as negative performance improvement in Figure 2. One possible reason is that words contextually similar to 'transcription' are not annotated in the training set either.

Table 4. Performance comparison of event trigger identification in different event types

Event category	Event type	Method	Recall (%)	Precision (%)	F-score (%)
Anatomical	Cell proliferation	B	69.77	63.83	66.67
		P	67.44	78.38	**72.5**
	Development	B	83.51	68.07	**75**
		P	81.44	69.30	74.88
	Blood vessel develop	B	96.33	95.70	96.01
		P	97.33	98.65	**97.99**
	Growth	B	83.93	69.12	75.81
		P	83.92	77.05	**80.34**
	Death	B	94.29	56.90	70.97
		P	88.57	72.09	**79.49**
	Breakdown	B	34.78	80	48.48
		P	34.78	80	48.48
	Remodeling	B	60	85.71	70.59
		P	60	85.71	70.59
Molecular	Synthesis	B	50	33.33	40
		P	50	40	**44.44**
	Gene expression	B	93.94	83.78	**88.57**
		P	92.42	84.72	88.41
	Transcription	B	14.28	25	**18.18**
		P	0	0	0
	Catabolism	B	0	0	0
		P	33.33	16.67	**22.22**
	Phosphorylation	B	100	50	66.66
		P	100	75	**85.71**
	Dephosphorylation	B	0	0	0
		P	100	100	**100**
General	Localization	B	83.46	79.86	81.62
		P	85.71	80.85	**83.21**
	Binding	B	76.36	84	**80**
		P	78.18	81.13	79.63
	Regulation	B	60.37	46.48	52.52
		P	53.05	56.49	**54.72**
	Positive regulation	B	86.73	67.85	76.14
		P	86.41	71.58	**78.30**
	Negative regulation	B	77.03	74.35	75.66
		P	78.83	77.09	**77.95**
Planned	Planned process	B	75	53.92	62.73
		P	75.64	56.46	**64.66**

Note: 'B' denotes the baseline approach, 'P' denotes our proposed method in the 'Method' column and the better performance is shown in boldface.

3.2.2 Comparison of feature combination methods To investigate the effectiveness of the feature combination method based on MKL, experiments were conducted where MKL is replaced with a simple averaging method. In the average method, features induced by neural language model and features extracted from syntactic and semantic parsing results are combined with equal weights. Table 5 shows the comparison result of the two methods. It can be observed that the precision of event trigger identification is improved by >3% when using MKL for feature combination. Nevertheless, its recall value slightly dropped. The overall improvement on F-score is ~1.2%. Although the improvement appears to be marginal, the MKL method should still be favored for feature combination when precision value could well be regarded as much more important than recall in the open biomedical domain.

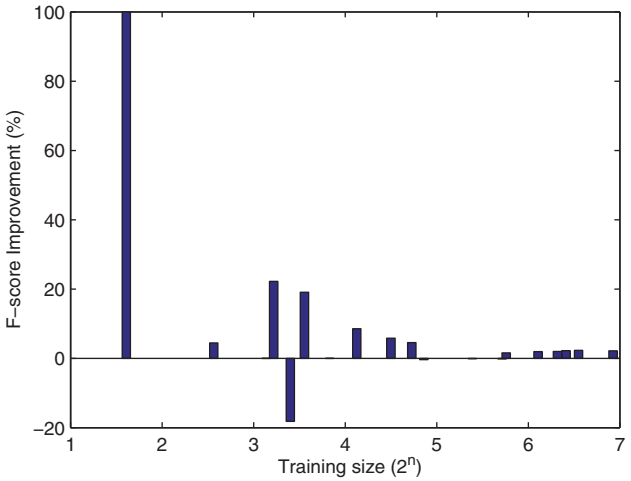

Fig. 2. Performance improvement versus size of training data in each event category

Table 5. Event trigger identification results with or without MKL

Method	Recall (%)	Precision (%)	F-score (%)
Averaging	82.89	72.14	77.14
MKL	81.29	75.56	78.32

Table 6. The coverage of all the distinct words and the coverage of all the words in our crawled Medline corpus for each vocabulary

Size of vocabulary	Coverage of all the distinct words (%)	Coverage of all the words (%)
15 000	0.84	95.07
30 000	1.68	96.57
60 000	3.36	97.56
90 000	5.04	97.98

3.2.3 Impact of dictionary size on neural language modeling The vocabulary size D in the neural language model is set in advance. If D is too small, some semantically important words might be omitted. On the contrary, if D is too big, some noisy words might be included and it becomes expensive to train the neural language model. To explore whether and how vocabulary size in the neural language model impacts the trigger word identification performance of the proposed framework, four different vocabularies were used in neural language model learning. We first list in Table 6 the coverage of all the distinct words and the coverage of all the words in our crawled Medline corpus for each vocabulary. It shows that the top most frequent words occur most of the time. For all the vocabularies we experimented

Table 7. Event trigger identification performance with neural language model with different vocabularies

Size of vocabulary	Recall (%)	Precision (%)	F-score (%)
15 000	82.03	73.90	77.75
30 000	81.29	75.56	78.32
60 000	81.29	75	78.02
90 000	80.60	74.68	77.53

Table 8. Event trigger identification performance with neural language model trained from difference sources

Method	Recall (%)	Precision (%)	F-score (%)
Wikipedia	82.60	70.50	76.07
Medline	81.29	75.56	78.32

here, they cover at least 95% of word occurrences in the whole corpus.

Table 7 lists the results obtained on the test set of the MLEE corpus with different vocabulary size. It can be observed that the final performance of the proposed framework outperforms the baseline approach regardless which vocabulary was used. The relative improvement on F-score ranges between 1.7 and 2.5%. We also observe that increasing the vocabulary size improves the performance with the peak reached at 30 000. Based on the above observation, we can conclude that the choice of the vocabulary can be made by considering its coverage of the words in the corpus.

3.2.4 Learning neural language model from difference source To explore the effectiveness of embedding domain knowledge into language model, we compare the event trigger identification results with neural language model trained on Wikipedia (Collobert *et al*., 2011). The Wikipedia corpus contains a wide range of topics in general domains. The results are shown in Table 8. Compared with the baseline approach, using the Wikipedia corpus did not appear to improve the performance of event trigger identification. Nevertheless, learning the neural language model from the Medline corpus gives superior performance on event trigger identification than the baseline. Only domain-specific knowledge can be used to improve the performance of event trigger identification.

3.2.5 Neural language model versus topic model To further investigate the effectiveness of neural language model, we compare the event trigger identification results with word classes induced by the LDA model, which is a generative graphical model originally proposed for topic discovery (Blei *et al*., 2003). Assuming that each document is represented as an unordered collection of words and characterized by a particular set of topics, disregarding grammar and word order, the LDA model can be used for grouping the words in similar topics in an unsupervised way. Each word in the LDA model is represented as probability distribution over topics, and then combined with the features described in Section 3.2 for training SVM classifiers for event trigger identification. In our experiments, the LDA model by varying the number of topics {50, 100, 150, 200, 250} using the Stanford topic modeling toolbox (http://nlp.stanford.edu/downloads/tmt/tmt-0.4/). The optimal topic number is chosen using the perplexity measure on the 10% held-out set from our Medline corpus. The final event trigger identification results using LDA are reported in Table 9 by setting the topic number to 200. It can be observed that LDA only gives an almost negligible improvement of 0.24% in F-score compared with the baseline and it performs worse than our proposed framework using neural language modeling. Two possible reasons are (i) LDA ignores word ordering in documents, which is important when comparing words occurring in similar semantic context and (ii) it is difficult to choose the proper number of topics (or word classes) that group words into well-separated semantic clusters. On the contrary, our proposed framework is based on neural language modeling, which learns the distributional representation of words without the need of specifying the number of induced word classes.

Table 9. Event trigger identification performance using neural language modeling versus LDA

Method	Recall (%)	Precision (%)	F-score (%)
Baseline	81.69	70.79	75.84
LDA	81.12	71.64	76.08
NLM	81.29	75.56	78.32

4 CONCLUSIONS AND FUTURE WORK

In this article, we have proposed a novel framework to construct a feature set for learning classifiers for event trigger identification. In particular, biomedical domain knowledge is learned from a large text corpus built from Medline and embedded into word features using neural language modeling. The embedded features are combined with the well-designed syntactic and semantic context features, which is further used for event trigger classifier learning. Experimental results on the MLEE corpus show that >2.5% improvement on F-score is achieved by the proposed framework when compared with the state-of-the-art feature-based approach, demonstrating the effectiveness of our proposed framework. In future work, we will further investigate the feasibility of our proposed framework on other corpora. Another possible future direction is to incorporate domain-specific prior knowledge into neural language model learning using semi-supervised learning to further improve the performance of event trigger identification.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their insightful comments. They also thank Dr Makoto Miwa for his suggestions on constructing the baseline system.

Funding: This work was funded by the National Natural Science Foundation of China (61103077), Ph.D. Programs Foundation of Ministry of Education of China for Young Faculties (20100092120031), the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, the Cultivation Program for Young Faculties of Southeast University and the Shenzhen International Cooperation Research Funding (GJHZ20120613110641217).

Conflict of Interest: none declared.

REFERENCES

Bach,F.R. *et al.* (2004) Multiple kernel learning, conic duality, and the SMO algorithm. In: *Proceedings of the 21st International Conference on Machine Learning*. New York.

Blei,D. *et al.* (2003) Latent Dirichlet allocation. *J. Mach. Learn. Res.*, **3**, 993–1022.

Collobert,R. *et al.* (2011) Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, **12**, 2493–2537.

Gehler,P. and Nowozin,S. (2009) On feature combination for multiclass object classification. In: *IEEE 12th International Conference on Computer Vision* 2009, Vol.1. pp. 221–228.

Harris,Z. (1970) Distributional structure. In: *Papers in Structural and Transformational Linguistics*. D. Reidel Publishing Company, Dordrecht, Holland, pp. 775–794.

Huang,E.H. *et al.* (2012) Improving word representations via global context and multiple word prototypes. In: *Annual Meeting of the Association for Computational Linguistics 2012*, Vol. 1. pp. 873–882.

Kim,J.-D. *et al.* (2009) Overview of bionlp'09 shared task on event extraction. In: *Proceedings of the Workshop on BioNLP*. NJ, pp. 1–9.

Kim,J.D. *et al.* (2012) The genia event and protein coreference tasks of the bionlp shared task 2011. *BMC Bioinformatics*, **13**, S1.

Miyao,Y. and Tsujii,J. (2008) Feature forest models for probabilistic hpsg parsing. *Comput. Linguist.*, **34**, 35–80.

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

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

Rakotomamonjy,A. *et al.* (2008) SimpleMLK. *J. Mach. Learn. Res.*, **9**, 2491–2521.

Sagae,K. and Tsujii,J. (2007) Dependency parsing and domain adaptation with LR models and parser ensembles. In: *EMNLP-CoNLL'2007*, Vol. 1, pp. 1044–1050.

Zhou,D. and He,Y. (2011) Biomedical events extraction using the hidden vector state model. In: *Artificial Intelligence in Medicine*, Vol. 53, pp. 205–213.