Methods 83 (2015) 44–50

Contents lists available at ScienceDirect

Methods

journal homepage: www.elsevier.com/locate/ymeth

An approach to improve kernel-based Protein–Protein Interaction extraction by learning from large-scale network data

Lishuang Li *, Rui Guo, Zhenchao Jiang, Degen Huang

School of Computer Science and Technology, Dalian University of Technology, Dalian 116023, China

ARTICLE INFO

Article history:
Received 29 January 2015
Received in revised form 26 March 2015
Accepted 28 March 2015
Available online 9 April 2015

Keywords:
Protein–Protein Interaction
Word representation
Distributed representation
Brown clusters

ABSTRACT

Protein–Protein Interaction extraction (PPIe) from biomedical literatures is an important task in biomedical text mining and has achieved desirable results on the annotated datasets. However, the traditional machine learning methods on PPIe suffer badly from vocabulary gap and data sparseness, which weakens classification performance. In this work, an approach capturing external information from the web-based data is introduced to address these problems and boost the existing methods. The approach involves three kinds of word representation techniques: distributed representation, vector clustering and Brown clusters. Experimental results show that our method outperforms the state-of-the-art methods on five publicly available corpora. Our code and data are available at: http://chaoslog.com/improving-kernel-based-protein-protein-interaction-extraction-by-unsupervised-word-representation-codes-and-data.html.

1. Introduction

As one of the major branches in the biomedical text-mining field, the technology for Protein–Protein Interaction extraction (PPIe) is of great application value and research significance, particularly for the construction of knowledge networks and the ontology for proteins. However, with the fast development of information technology, the amount of biomedical literature in the network grows rapidly, which makes it more difficult for researchers to grab the required information.

Protein–Protein Interaction extraction aims to judge whether the sentence, which contains more than one protein, actually implies interactions between a pair of proteins or not. The current methods for the relation extraction fall into three main categories: word co-occurrence [1], pattern matching [2,3] and statistical machine learning [4,5]. Statistical machine learning methods which have achieved great success on other Nature Language Processing (NLP) tasks are more widely used in the PPIe task. The words surrounding the pair of protein names are usually collected into a bag which is referred as "Bag of Word (BOW) Feature" in the machine learning methods. However, this traditional mode for feature representation encounters several obstacles. One problem is "vocabulary gap", meaning that synonymous words are completely isolated from each other, and the model cannot handle the words that are not in the annotated training data at the test phrase. As an example shown in Fig. 1, the two words "induced" in Sent.1 and "caused" in Sent.2, play the same role in the detection of PPI. However, the classifiers cannot learn the similar meaning from the BOW features, as well as a word and its derived word, for instance, "binding" and "binds" are completely different in BOW features.

> *Sent.1*: Shedding of the extracellular domain of TNF-R2 was induced by 4 beta-phorbol 12-myristate 13-acetate but not by TNF-alpha or TNF-beta.
> *Sent.2*: Microinjection of anti-cdc25A antibodies into HeLa cells causes their arrest in mitosis.

Another problem is "data sparseness". Parameters in the trained model will be poorly estimated for rare words in the annotated training data. To tackle the problems mentioned above, one approach is to introduce extra features into the supervised task, which is generally called "Word Representation" [6–9].

One classical approach to realize word representation is word clustering, especially hierarchical clustering. For example, Ratinov [6] applied Brown clustering [7] results as features in Name Entity Recognition (NER) tasks. Sun [8] augmented the

* Corresponding author at: School of Computer Science and Technology, Dalian University of Technology, Chuang Xin Park A930, No. 2 Ling Gong Road, Gan jing zi District, Dalian 116023, China. Fax: +86 041184708140.
E-mail address: lilishuang314@163.com (L. Li).

http://dx.doi.org/10.1016/j.ymeth.2015.03.026
1046-2023/

Sent.1: Shedding of the extracellular domain of TNF-R2 was induced by 4 beta-phorbol 12-myristate 13-acetate but not by TNF-alpha or TNF-beta.
Sent.2: Microinjection of anti-cdc25A antibodies into HeLa cells causes their arrest in mitosis.

Fig. 1. Two sentences from AIMed corpus to illustrate the problem of BOW Feature.

relation extraction task by representing different lexical features with appropriate clustering methods. In recent years, another unsupervised method, called "distributed representations", has been studied further to deal with NLP problems. Collobert and Weston [9] proposed an approach called SENNA for distributed representations and applied it in the Part-Of-Speech Tagging, Chunking, NER, and Semantic Role Labeling tasks. In brief, the word representation approach has been widely applied in some sequence labeling tasks and has achieved improvements in accuracy. However, to our knowledge, the distributed representation method has never been applied into relation extraction, including Protein–Protein Interaction extraction.

In this work, we use the ensemble kernel method that is composited by feature-based kernel and tree kernel to extract Protein–Protein Interaction, and the word representation approach is introduced to avoid negative impacts caused by vocabulary gap and data sparseness. We adopt three types of word representation methods for the comparison, which includes distributed representation, vector clustering representation and Brown clusters representation. The results show that the distributed representation method performs better than the other two clustering representation methods as well as other state-of-the-art PPIe methods on five public corpora.

2. Workflow and tools

A workflow to describe our method is shown in Fig. 2.

The input of our method contains five annotated corpora: AIMed, BioInfer, HPRD50, IEPA and LLL for testing PPIe performance.

Pre-processing is operated using the OpenNLP tools. Word representation based on the large-scale text from the PubMed database and syntactic parsing based on the five annotated corpora are performed to extract rich feature information using Word2Vec and Stanford Parser respectively. Then the feature information is imported into SVM-LIGHT-TK to learn a model that can judge if a pair of proteins interacts with each other in given context automatically.

The output of our method is the classification results of protein pairs in the sentences of the five annotated corpora, treating the positive instances as the interacting pairs while the negative instances as the non-interacting pairs.

The following tools and databases are used in the preceding steps:

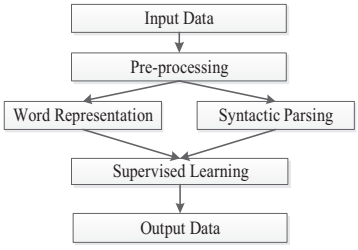

Fig. 2. Workflow of our PPIe method.

(1) Word2Vec: a tool that computes continuous distributed representations of words.
(2) OpenNLP: a toolkit for the processing of natural language text, such as tokenization, sentence segmentation, part-of-speech tagging.
(3) Stanford Parser: a tool that works out the grammatical structure of sentences.
(4) SVM-LIGHT-TK: a tool that can calculate the dot product between two feature vectors and the similarity between two syntactic trees simultaneously.
(5) PubMed: a database that has indexed millions of citations for biomedical literature from MEDLINE (a bibliographic database of life sciences and biomedical information), life science journals, and online books.

3. Methods

As shown in Fig. 3, our method mainly covers three steps: document collection, feature extraction, and supervised learning. Each component is described in the following subsection.

3.1. Document collection

We construct two datasets in our method: one for training word representation which consists of 5.33 million biomedical abstracts (5.6 GB unannotated text) downloaded from PubMed using "protein" as a keyword; the other for testing PPIe performance which only consists of the five annotated corpora. The pre-processing on the two datasets is performed as follows:

(1) *Sentence detection*: we divide the unannotated large-scale text into sentences.
(2) *Disjunction*: we cut a sentence into parts, for example, "Raf-1 was activated by <JAK2>." → "Raf-1", "was", "activated", "by", "<", "JAK2", ">", ".".
(3) *Unified Greek*: we convert the Greek letters in the two datasets into the unified format, such as α → alpha, β → beta.
(4) *Lowercased*: each word in the two datasets is lowercased.

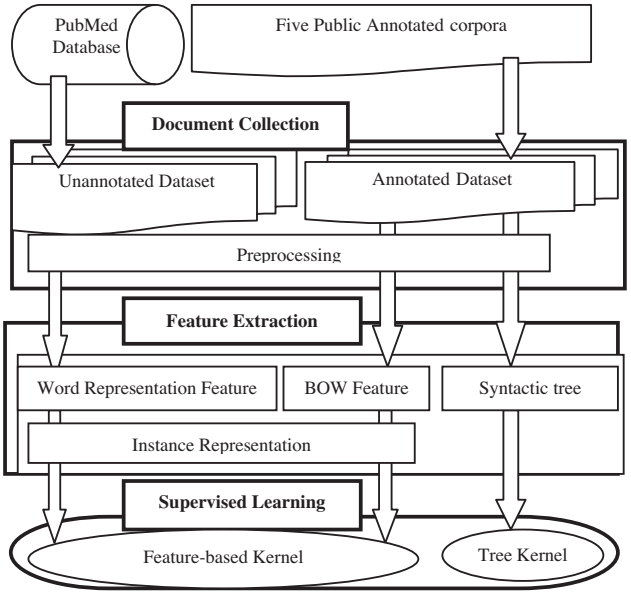

Fig. 3. Overview of our PPIe framework.

3.2. Feature extraction

Different types of features reveal different information aspects of the sentence which contains the current PPI instance. In order to draw out more useful information for a given PPI instance, we adopt the following features: linear feature information and syntactic feature information. The linear feature information aims to catch the context information, whereas the syntactic feature information focuses on the syntactic structure of the sentence.

3.2.1. Linear feature information

3.2.1.1. Bag of words(BOW) feature. The BOW features are presented as follows:

(i) *Words from protein names*: all the words in two protein names are included in the features.
(ii) *Words between two protein names*: these features include all words that are located between two protein names. If no word appears between two protein names, the feature will be "NULL".
(iii) *Words surrounding two protein names*: these features include the left n words of the first protein name and the right n words of the second protein name. n is set to be five in our experiments. If there are no words surrounding two protein names, "NULL" will be used.
(iv) *Interaction terms*: A sentence is considered to contain PPI information only if it includes at least one interaction word or keyword (such as "regulate", "interact", "modulate", etc.). If there is one keyword between or among the surrounding words of two protein names, the keyword is added into the keyword feature; if there is no keyword, the keyword feature will be set to "NULL".

3.2.1.2. Word representation feature.

3.2.1.2.1. Distributed representation feature. Distributed representation, also known as word embeddings, has recently been proposed to address several NLP problems and has achieved great success [9–12]. A word expressed by the distributed representation is a dense, low-dimensional and real-valued vector, hopefully capturing the syntactic and semantic information in each dimension. The basic idea of distributed representation is that vectors are trained by surrounding words. Aiming at a much lower computational cost, Mikolov [10–12] recently developed the Word2Vec tool for computing continuous distributed representations of words. The tool implemented the Skip-gram model [10] expanded on the n-gram model architectures, which aimed to employ the current word to train other words which are in the fixed window of the current word in the same sentence. The word vectors can capture many linguistic regularities, which is superior to one-hot representation e.g., the biomedical terms "*gene*", "*proteins*", "*kinase*" are close to each other, and far from the nouns "*months*", "*weeks*", while for one-hot encoding, all these words are equally distant.

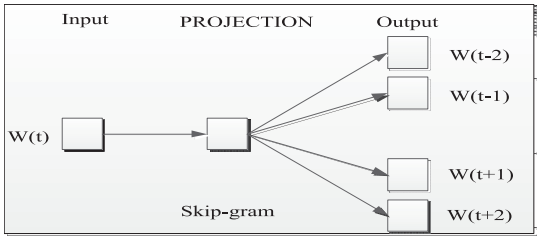

Fig. 4. The Skip-gram model architecture in Word2Vec.

We use the skip-gram model (described in Fig. 4) to obtain the distributed representation from the large-scale unannotated text. Trained on the large-scale text, each word in the corpora is represented as a low-dimensional real-valued vector. In our experiment, the vector dimension is set to 400.

3.2.1.2.2. Vector clustering representation feature. As one of the simplest and most well-known clustering algorithms, K-means clustering algorithm has also been introduced to deal with the semi-supervised learning problem. Lin and Wu [13] presented a K-means-like non-hierarchical clustering algorithm for phrases, and used the clustering results as features in NER and query classification applications. K-means algorithm will divide the n vectors $\{x_1, x_2 \ldots x_n\}$ into k sets (k clusters), where $k \leqslant n$. Different from the distributed representation, the vector clustering representation puts the words that are close in the vector space into a cluster based on the K-means algorithm.

In this work, we employ K-means algorithm to perform clustering on the word vectors (the vectors in distributed representation).The max iteration C is set to 1000, and the number of clusters $K = \sqrt{|V|/2} = 1732$, where $|V| = 1500000$ is the vocabulary size. The clustering output is added as additional features. Since the clustering number in our method is 1732, each word that appears in the BOW feature gets a number (1-1732) indicating its category. Then, all the numbers corresponding to the words in the BOW feature are adopted as additional vector clustering features. The same strategy is also applied on the Brown clustering output.

3.2.1.2.3. Brown clusters feature. Different from the non-hierarchical K-means algorithm, Brown algorithm [7] is a classic hierarchical clustering algorithm based on n-gram language models. One shortage of Brown clusters is that it is based solely on bigram statistics, and does not consider word usage in a wider context. In this algorithm, each word gets its own cluster initially. Then, the algorithm repeatedly picks two clusters with the maximum mutual information and merges them into a single cluster until the number of the clusters is equal to the given parameter. The result of Brown clustering algorithm is a binary tree, where each word occupies a single leaf node, and where each leaf node contains a single word. A particular word can be assigned a binary string by following the traversal path from the root to its leaf, assigning 0 for each left branch, and 1 for each right branch. Examples of Brown clustering results are shown in Table 1.

3.2.1.3. Instance representation. Different strategies for instance representation are applied on different features.

3.2.1.3.1. One-hot encoding. Each PPI instance is represented as a binary vector. Each feature of the current PPI pair is mapped to a dimension in the binary vector, and the dimension value is set to '1', otherwise, '0'. This strategy is applied on the BOW feature, vector clustering representation feature, Brown clusters feature.

3.2.1.3.2. Distributed representation. Different from one-hot encoding, one PPI instance is represented as a real-valued vector. In this method, every word in an instance is represented as a vector obtained from distributed representation method. Thus an instance will be shaped by concatenating several vectors corresponding to words in the instance.

Table 1
Examples of Brown clustering results.

Path	Words
0100001100	Diminution, elevations, reductions, alteration
01000011011	Fall, delay, rise, decrease
0100011100	Suggestion, assumption, notion, idea, concept
010011011111	Edge, face, ends, end, terminus
01001110101	Trigger, causes, cause

3.2.2. Syntactic feature information

In this work, we use the Dynamic Extended Tree (DET) kernel implemented in our previous work [14] to evaluate the syntactic information. The tree kernel adopts the convolution tree kernel proposed by Collins et al. [15]. A convolution tree kernel aims to capture the structured information in a sentence. It calculates the syntactic structure similarity between two parse trees by counting the number of common sub-trees [15,16]. In order to focus on the most relevant information to relations, a standard tree kernel is defined on the Minimum Complete Tree (MCT). It is the sub-tree rooted by the nearest common parent node of the two proteins under consideration. In our tree kernel, the protein pairs in the sentence are replaced by PROTEIN_1 and PROTEIN_2, and the other protein names in the same sentence are replaced by PROTEIN. Stanford Parser [17] is used to parse the sentence. However, MCT includes too much left and right context information, which may elicit many noisy features. Zhang et al. [18] proposed five pruning strategies and found that the Shortest Path enclosed Tree (SPT) performed best. SPT is the smallest common sub-tree including the two proteins. In other words, the sub-tree is enclosed by the shortest path linking the two proteins in the parse tree. Fig. 5 illustrates the different representation of MCT and SPT of a relation instance. The candidate interaction pair is marked as PROTEIN_1 and PROTEIN_2, the other proteins are marked as PROTEIN. The SPT between the focused proteins is shown in the dotted line and MCT is the whole figure.

Despite the better performance of SPT, in some cases, the information contained in SPT is not sufficient to determine the relation between two proteins. For example, the word "interact" is critical to determine the relation between PROTEIN_1 and PROTEIN_2 in the sentence "PROTEIN_1 and PROTEIN_2 interact with each other". However, it is not contained in the SPT (dotted circle in Fig. 6). By analyzing the corpus, the quantity of nodes in these SPTs are found often less than seven, so they include less information except the two protein names. Here, we propose a novel dynamic extended strategy. By default, the tree kernel adopts the original SPT; When the number of the nodes in a SPT is smaller than 7:

(1) If SPT is different with MCT, SPT will be replaced with MCT;
(2) otherwise, SPT will be replaced with the MCT rooted by the parent node of the root of the original SPT.

Thus the extended SPT (solid circle in Fig. 6) will include richer context information than the original SPT. In the above example, SPT and MCT are the same, then the SPT will be extended from

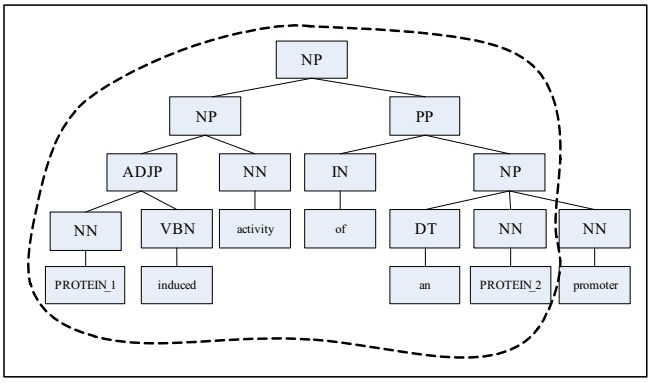

Fig. 5. The MCT and the SPT of the sentence, "PROTEIN coexpression largely abolished PROTEIN_1 induced activity of an PROTEIN_2 promoter". The SPT between the focused proteins is shown in the dotted line and MCT is the whole figure.

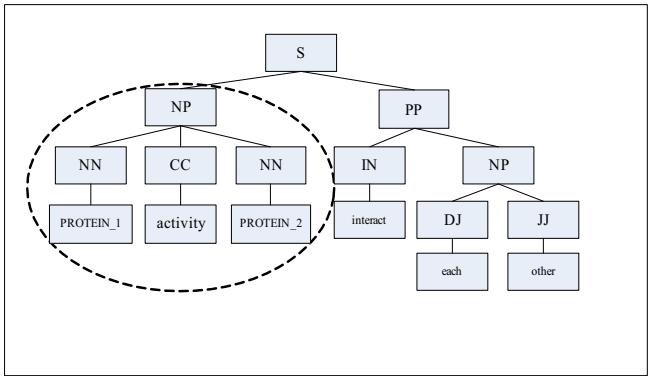

Fig. 6. An example of the extension of SPT. The sentence is "PRITEIN_1 and PROTEIN_2 interact with each other". The original SPT is in dotted circle and the extended SPT is in solid circle.

"(NP (NNP PROTEIN_1) (CC and) (NNP PROTEIN_2))" to "(S (NP (NNP PROTEIN_1) (CC and) (NNP PROTEIN_2)) (VP (VBP interact) (PP (IN with) (NP (DT each) (JJ other)))))", which includes the keyword "interact" and richer context information. We call it Dynamic Extended Tree (DET).

3.3. Supervised learning

We employ Support Vector Machine (SVM) as the basic learning algorithm and the kernel is composited by the feature-based kernel and tree kernel.

The composite kernel can be obtained by (1). R_1 and R_2 are two instances (represented by feature vectors). T_1 and T_2 are the dynamic extended trees of R_1 and R_2 respectively. The parameter λ is the weight of the K_T ($\lambda = 1$ in our experiment).

$$K(R_1, R_2) = K_L(R_1, R_2) + \lambda K_T(R_1, R_2) \quad (1)$$

where $K_L(R_1, R_2)$ is the feature-based kernel that is used to calculate the similarity between two feature vectors, and $K_T(T_1, T_2)$ is the DET-based kernel used to calculate the similarity between two dynamic extended trees [14] containing the corresponding protein pairs, the equation is represented as (2). N_1 and N_2 are the node sets for tree T_1 and T_2 respectively. $\Delta(ro_1, ro_2)$ is the convolution kernel to calculate the similarity between two sub-trees (ro_1 and ro_2 are the root nodes of sub-trees).

$$K_T(T_1, T_2) = \sum_{ro_1 \in N_1,\ ro_2 \in N_2} \Delta(ro_1, ro_2) \quad (2)$$

4. Results

4.1. Experimental settings

Our method is evaluated on five public corpora: AIMed [19], BioInfer [20], HPRD50 [21], IEPA [22], and LLL [23]. All the corpora, with slightly different annotating policies, are organized in a common format [24] and are generally used in the assessment of PPIe methods.

The ensemble kernel in our method is implemented on the SVMLIGHT-TK toolkit. We evaluate our method by *F*-score which is defined as:

$$P = \frac{TP}{TP + FP} R = \frac{TP}{TP + FN} F - score = \frac{2 \times P \times R}{P + R} \quad (3)$$

where P is precision rate, R is recall rate, TP is true positives, FP is false positives and FN is false negatives. In this paper, the positive result is TP if it actually contains PPI; the positive result is FP if it

Table 2
Results of the kernel-based methods.

Method	Dataset				
	AIMed (%)	BioInfer (%)	HPRD50 (%)	IEPA (%)	LLL (%)
Feature-based kernel	43.3	64.2	54.4	62.5	82.2
Tree kernel	55.3	58.9	74.1	71.1	81.5
Ensemble Kernel	**61.9**	**69.2**	**75.2**	**72.6**	**84.9**

doesn't contains PPI; the negative result is *FN* if it contains PPI actually. All the performance is measured by 10-fold cross-validation.

4.2. Baseline results

Table 2 compares the ensemble kernel method with its two individual kernels, achieving better *F*-scores of 61.9% on AIMed, 69.2% on BioInfer, 75.2% on HPRD50, 72.6% on IEPA and 84.9% on LLL respectively. In this work, we employ the ensemble kernel method as our baseline, and the feature-based kernel of the baseline only contains the BOW feature.

4.3. Word representation results

We boost the baseline by replacing the BOW feature in the baseline with three different word representation methods respectively, and the results are shown in Table 3. For simplicity, the Brown clusters is referred as W_1 feature, the vector clustering representation feature is referred as W_2 feature, and the distributed representation feature as W_3 feature. When the W_1 feature is adopted, the F-score is improved by 2.7% on AIMed, while decreasing on other corpora slightly. When the W_2 feature is applied, the F-scores increase by 2.8% on AIMed, 1% on BioInfer, and 1.3% on IEPA corpus, however, the F-score is decreased from 75.2% to 74.8% on BioInfer and from 84.9% to 83.8% on HPRD50. The performance on the five corpora all increases significantly with the W_3 feature, precisely, the F-scores are improved by 7.8% on AIMed, 4.8% on BioInfer, 2.8% on HPRD50, 3.9% on IEPA and 2.4% on LLL respectively. We can see that the W_3 feature can achieve great improvement on PPIe task, and it outperforms the two clustering-based representation methods.

In order to further compare the effects of the three word representation methods, we add them as extra features into baseline respectively and the results are shown in Table 4. When the W_1 feature is added, the F-scores are improved on the AIMed, BioInfer, and HPRD50 corpus, while the F-scores are decreased on other corpora (72.6% vs. 72.0% on IEPA, and 84.9% vs. 84.3% on LLL). When the W_2 feature is added, the *F*-scores on the five corpora are all increased, that is, by 1.8% on AIMed, 1.5% on BioInfer, 0.7% on HPRD50, 1.0% on IEPA and 0.3% on LLL. When the W_3 feature is added, the performances on the five corpora are all improved, by 7.1% on AIMed, 4.9% on BioInfer, 2.8% on HPRD50, 3.7% on IEPA, and 2.9% on LLL.

From the above analysis we can see that:

(1) The W_1 feature can make improvements on some corpora;
(2) Adding the W_2 feature improves the performance on five corpora, while the performance degrades on HPRD50 and LLL corpus replacing the W_2 feature with the BOW feature;
(3) The W_3 improves the performance on five corpora.

4.4. Comparisons between our method and the previous works

Table 5 shows the comparisons between our method and other state-of-the-art works on the five corpora. We can see that our method achieves the best performance on all five corpora. The methods in [25–27] also adopted multiple kernel combination. Miwa [25] combined three individual kernels to extract PPI: the feature-based kernel, the tree kernel and the graph kernel. Yang [26] also improved the PPI extraction by combining the following kernels: feature-based, tree, graph and POS path kernel. Li [27] combined feature-based kernel and tree kernel, reaching an F-score of 63.23% on AIMed corpus. Choi [28] made the in-depth analyses of the convolution parse tree kernel and achieved remarkable performance on the five corpora. By introducing the word representation features, our method reaches an F-score of 69.7% on AIMed corpus, exceeding 5.34% than Li [27], and performing much better than Miwa [25] on five corpora. In addition, the F-scores of our method outperform Yang [26] on AIMed, BioInfer, HPRD50, IEPA and LLL by 5.3%, 8.2%, 3.7%, 0.8% and 4.3% respectively. Generally, our method achieves the best performance. It can be concluded that the introduction of word representation, especially distributed representation, achieves great improvement on PPIe task.

Table 3
Results of replacing the BOW features with word representation features. W_1 is Brown clusters feature, W_2 is vector clustering representation feature, W_3 is distributed representation feature.

Method	Dataset				
	AIMed (%)	BioInfer (%)	HPRD50 (%)	IEPA (%)	LLL (%)
Baseline	61.9	69.2	75.2	72.6	84.9
W_1	**64.6**	67.8	74.2	72.2	84.1
W_2	**64.7**	**70.2**	74.8	**73.9**	**83.8**
W_3	**69.7**	**74.0**	**78.0**	**76.5**	**87.3**

Table 4
Results of integrating word representation features into Base line. W_1 is Brown clusters feature, W_2 is vector clustering re presentation feature, W_3 is distributed representation feature.

Method	Dataset				
	AIMed (%)	BioInfer (%)	HPRD50 (%)	IEPA (%)	LLL (%)
Baseline	61.9	69.2	75.2	72.6	84.9
Baseline + W_1	**64.2**	**69.7**	**76.1**	72.0	84.3
Baseline + W_2	**63.7**	**70.7**	**75.9**	**73.6**	**85.2**
Baseline + W_3	**69.0**	**74.1**	**78.0**	**76.3**	**87.8**

5. Discussion

In this work, three word representations methods are adopted as extra features to improve PPIe and achieve better performance than the existing methods. Experimental result shows that all the word representation methods can improve the performance of PPIe task. Furthermore, the distributed representation method performs much better than the two clustering representation methods. The reasons are analyzed as follows:

Table 5
Performance comparisons on five corpora with previous works.

Method	Dataset				
	AIMed (%)	BioInfer (%)	HPRD50 (%)	IEPA (%)	LLL (%)
Ours	**69.7**	**74.0**	**78.0**	**76.5**	**87.3**
Miwa [25]	60.8	68.1	70.9	71.7	80.1
Choi [28]	67.0	72.6	73.1	73.1	82.1
Yang [26]	64. 41	65.84	74.38	75.73	83.01
Li [27]	63.23	–	–	–	–

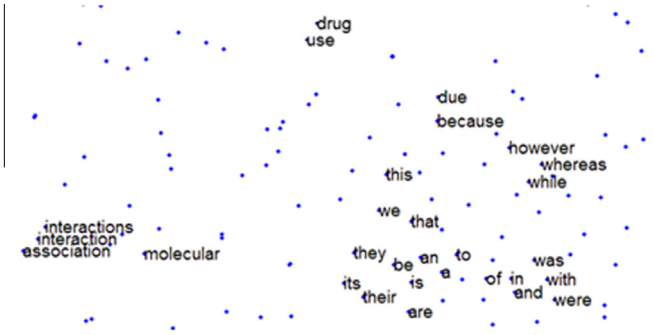

Fig. 7. Visualizing data using t-SNE [30] (Part of words are hidden to make the figure simple). Each vector in the distributed representation method is given a location in a two-dimensional map.

(1) The main problems of the BOW feature are: any two words are isolated and similar meaning cannot be learnt from other words or context; for the words that are not or rare in the annotated training data, the latent information about the current word cannot be obtained. However, the word representation methods can learn much deeper syntactic and semantic information from the large set of out-of-domain data. For example, Fig. 7 shows the words with the same semantic or syntactic information in distributed representation method is closer to each other. We can see that more and deeper information can be learnt compared with the BOW feature.

(2) The distributed representation method can utilize more context information than the Brown clustering method. Results in previous work [13] showed that different word representations are preferable for different tasks. Turian [29] proved that the Brown clusters performed better than the distributed representation method in NER task. Different from NER task, the PPIe task involves a wider context. The Brown algorithm is solely based on bigram statistics, and does not consider a wider context. Comparatively, the Skip-gram model in our distributed representation method considers a larger-sized window, which captures more context information.

(3) The distributed representation method performs better than vector clustering representation method. The vector clustering representation selects the words with similar distributed representation vectors into a cluster. Therefore, the information granularity for the distributed representation is much more fine-grained than the vector clustering representation. As a result, the rich information in the distributed representation method cannot be covered by the vector clustering representation method.

6. Conclusion

In this paper, we present an unsupervised approach to draw extra information from the unannotated data crawled from the network to boost the existing PPIe methods. Different word representation methods are systematically compared in this work. Experimental results show that the distributed representation method performs much better than the baseline and the clustering representation methods, and achieves the stated-of-art performance with the F-scores of 69.7% on AIMed, 74.0% on BioInfer, 78.0% on HPRD50, 76.5% on IEPA and 87.3% on LLL corpus. We come to the conclusion that the distributed word representation method can make great improvement on PPIe task.

The previous works show that the accuracy for PPIe is severely affected by the features designed by the researchers in the field. Since the words have been expressed into the real-valued vectors in the distributed representation methods, which can be an advance step, we would like to exploit the deep learning methods to avoid artificially making rules to extract the features. Furthermore, related researches show that more semantic information could be captured by phrase representation [31–33], which could enable the sharing of related phrases, and we will explore the methods to use word representations in compound features.

Acknowledgments

The authors gratefully acknowledge the financial support provided by the National Natural Science Foundation of China under Nos. 61173101, 61173100.

References

[1] R. Bunescu, R. Mooney A. Ramani, W. Marcotte, Integrating co-occurrence statistics with information extraction for robust retrieval of protein interactions from medline. In: Proc of BioNLP. New York city, pp. 49–56, 2006.
[2] M.L. Huang, X.Y. Zhu, Y. Hao, Donald G. Payan, Bioinformatics 20 (18) (2004) 3604–3612.
[3] K. Fundel, R. Kuffner, R. Zimmer, Bioinformatics 23 (3) (2007) 365–371.
[4] C.J. Sun, L. Lin, X.L. Wang, Y. Guang, Using maximum entropy model to extract protein–protein interaction information from biomedical literature, Lecture Notes in Computer Science, pp. 730–737, 2007.
[5] B.J. Cui, H.F. Lin, Z.H. Yang, SVM-based protein–protein interaction extraction from Medline abstracts. International Conference on Bio-inspired Computing: Theories and Application, pp. 546–551, 2007.
[6] L. Ratinov, R. Dan, Design challenges and misconceptions in named entity recognition. In the 13th Conference on Computational Natural Language Learning. Association for Computational Linguistics, Colorado, pp. 147–155, 2009.
[7] P.F. Brown, V.J.D. Pietra, Comput. Ling. 18 (1992) 467–479.
[8] A. Sun, G. Ralph, S. Satoshi, Semi-supervised relation extraction with large-scale word clustering, In the 49th Annual Meeting of the Association for Computational Linguistics, Human Language Technologies. Association for Computational Linguistics, USA, 2011. pp. 521–529.
[9] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. Kuksa, J. Machine Learn. Res. 12 (2011) 2493–2537.
[10] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv, 1301.3781, 2013.
[11] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119. 2013.
[12] T. Mikolov, W. Yih, G. Zweig, Linguistic regularities in continuous space word representations. In the 2013 Conference of the North American Chapter of the Association for Computational Linguistics, Human Language Technologies, Atlanta, pp. 746–751, 2013.
[13] D.K. Lin, X.Y. Wu, Phrase clustering for discriminative learning. In the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP. Association for Computational Linguistics, pp. 1030–1038, 2009.
[14] L.S. Li, P.P. Zhang, T.F. Zheng, H.Y. Zhang, Z.C. Jiang, D.G. Huang, PlOS ONE 9 (3) (2014) e91898.
[15] M. Collins, N. Duffy, Convolution kernels for natural language. In Advances in Neural Information Processing Systems, pp 625–632, 2001.
[16] A. Moschitti, A study on convolution kernels for shallow semantic parsing, In the 42nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, Barcelona, 2004.
[17] MC Marneffe, B. MacCartney, CD. Manning, Generating typed dependence parses from phrase structure parses. In Proceedings of LREC 6: 449–454, 2006.
[18] M. Zhang, J. Zhang, J. Su, G. D. Zhou, A composite kernel to extract relation between entities with both flat and structured feature, In the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics, Sydney, Australia, pp. 825–832, 2006.
[19] B. Razvan, G. Ruifang, J.K. Rohit, M.M. Edward, J.M. Raymond, K.R. Arun, W.W. Yuk, Artif. Intell. Med., 139–155, 2005.
[20] S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, T. Salakoski, BMC Bioinfor., 50, 2007.
[21] F. Katrin, R. Küffner, R. Zimmer, RelEx,Bioinformatics, 365–371, 2007.
[22] J. Ding, D. Berleant, D. Nettleton, E. Wurtele, Mining medline: abstracts, sentences, or phrases?. In the Proceedings of the pacific symposium on Biocomputing, pp. 326–327, 2002.
[23] C. Nédellec, Learning language in logic-genic interaction extraction challenge, In the 4th Learning Language in Logic Workshop, vol. 7, 2005.
[24] S. Pyysalo, A. Airola, J. Heimonen, J. Björne, BMC Bioinform. 9 (2008) S6.
[25] M. Miwa, R. Sætre, Y. Miyao, J. Tsujii, Int. J. Med. Inform. 78 (12) (2009) 39–46.
[26] Z.H. Yang, N. Tang, X. Zhang, H.F. Lin, Y.P. Li, Z.W. Yang, Artif. Intell. Med. 51 (3) (2011) 163–173.

[27] L.S. Li, Y. Liu, D.G. Huang, J. Chin. Inform. Process. 7 (2013) 86–92.
[28] S.P. Choi, S.H. Myaen, Simplicity is better: revisiting single kernel PPI extraction, In the 23rd International Conference on Computational Linguistics, Beijing, pp. 206–214, 2010.
[29] J. Turian, L. Ratinov, Y. Bengio, Word Representations: a Simple and General Method for Semi-supervised Learning, In the 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Uppsala, Sweden, 2010. pp. 384–394.
[30] V.D.M. Laurens, G. Hinton, J. Machine Learn. Res. 9 (2008) 2579–2605.
[31] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Adv. Neural Inform. Process. Sys. 26 (2013).
[32] R. Socher, C.D. Manning, Andrew Y. Ng Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks, Deep Learning and Unsupervised Feature Learning Workshop – NIPS, 2010.
[33] J. F. Gao, X. D He, W. T. Yih, and L. Deng. Learning Continuous Phrase Representations for Translation Modeling. Proceedings of ACL. 2014.