# Predicting Implicit Discourse Relations with Purely Distributed Representations

Haoran Li(✉), Jiajun Zhang, and Chengqing Zong

National Laboratory of Pattern Recognition, CASIA,
Beijing, People's Republic of China
{haoran.li,jjzhang,cqzong}@nlpr.ia.ac.cn

**Abstract.** Discourse relations between two consecutive segments play an important role in many natural language processing (NLP) tasks. However, a large portion of the discourse relations are implicit and difficult to detect due to the absence of connectives. Traditional detection approaches utilize discrete features, such as words, clusters and syntactic production rules, which not only depend strongly on the linguistic resources, but also lead to severe data sparseness. In this paper, we instead propose a novel method to predict the implicit discourse relations based on the purely distributed representations of words, sentences and syntactic features. Furthermore, we learn distributed representations for different kinds of features. The experiments show that our proposed method can achieve the best performance in most cases on the standard data sets.

**Keywords:** Implicit discourse relation · Distributed · Word level representation · Sentence level representation

## 1 Introduction

Automatic discourse relation recognition is an important task of discourse analysis, and it is beneficial to many NLP applications such as question answering, sentiment analysis and information retrieval. For instance, contingency relation detection can improve question answering systems and event relation extraction. Contrast relation recognition can eliminate intra-sentence polarity ambiguities.

Discourse relations can be grouped into explicit and implicit relations according to whether there are relation connectives. Connectives play a crucial role in inference of explicit discourse relations. For example, "but" is a strong indicative for COMPARISON relation while "and" for EXPANSION. In fact, previous studies [1] show that just using discourse connectives can achieve 94 % accuracy for explicit relation classification. In contrast, implicit discourse relations are much more difficult to be determined due to the absence of evident relation cues. In the natural texts, sentences without connectives account for a large proportion. For instance, about 39.54 % of the total discourse relations are annotated as implicit in Penn Discourse Treebank (PDTB). Consequently, researchers mainly focus on implicit relation recognition.


M. Sun et al. (Eds.): CCL and NLP-NABD 2015, LNAI 9427, pp. 293–305, 2015.
DOI: 10.1007/978-3-319-25816-4_24

To solve this issue, prior work resorted to explore diverse linguistically informed features, which started with lexical features introduced by [2], and then syntactic features were proved more effective [3,4]. Subsequent work focused on introducing more features [5,6] or selecting an optimal feature set [7]. These features are usually acquired by the help of external linguistic resources. For instance, word's polarity features are assigned according to its attribution in the Multi-perspective Question Answering Opinion Corpus, and inquirer tags are extracted from General Inquirer lexicon. WordNet features, verb class, affect features are leveraged as well.

The traditional methods introduced above for implicit relation classification face two major challenges. Firstly, they strongly depend on the external linguistic lexicons, which lack the generalization of the approaches for different languages. Taking the Inquirer tags feature as an example, the number of words which can't find the corresponding semantic categories in General Inquirer lexicon is more than 10,000, accounting for around 40 % of the whole words in PDTB.

Secondly, all of the traditional methods adopt discrete features which lead to severe data sparsity and cannot explore the similarities between discrete features. More than 10,000 production rules and 100,000 word pairs can be extracted from the PDTB even though a frequency threshold 5 has been used. [6,8,9] attempted to make the features less sparse. For example, [9] employed the Brown word clusters. These approaches can alleviate the data sparse problem to some extent, but they are still based on discrete features and cannot take full advantage of the similarities between discrete features.

In this paper, we propose a novel method based on the purely distributed representations. Our goal is to learn the low-dimensional dense vector representations for the discrete features from words, sub-sentence to syntactic production rules. Furthermore, we explore different algorithms for representation learning, such as deep neural networks (DNN) and principle component analysis (PCA).

With the learnt distributed representations, we then design an ensemble model to investigate different kinds of incorporation of the features in different levels.

We make the following contributions in this paper.

- Instead of discrete features, we propose a novel method of implicit discourse relation classification using only the purely distributed representations.
- We have explored the representation learning for different level features, from words, part-of-speech (POS), sentences to syntactic production rules. In addition, we apply different algorithms to learn different presentations for different kinds of discrete features.
- We have designed an ensemble model to explore various combinations of the distributed representations in different levels. The experiments show that we can obtain the best performance in most cases on the standard test sets. Figure 1 shows the framework of our method.

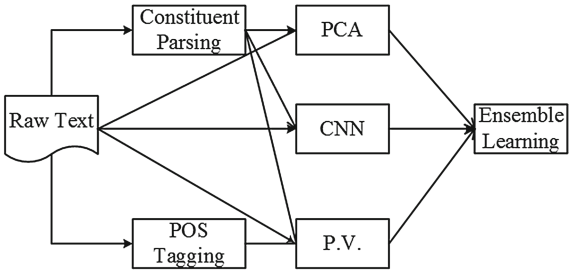


**Fig. 1.** Framework of our method, P.V. denotes paragraph vector model (can perform word2vec and sent2vec)

## 2 Related Work

### 2.1 Implicit Discourse Relation Recognition

Marcu and Echihabi [10] presented an unsupervised approach to identify discourse relations trained on a huge artificial corpus with the word pair features. Their training data was generated manually from raw corpora by detecting certain patterns based on cue phrases. For example, if there is a word "but" between two successive sentences, the sentence pairs are extracted as CONTRAST instance after removing the discourse connectives. They showed lexical features are effective for identifying discourse relations.

This work was followed by [11]. They performed an experiment to recognize discourse relations in Japanese by combining word pairs and phrasal patterns. A phrasal pattern explores the information existing in longer context beyond two sentence pairs. For instance, the pattern "... should have done ..." immediately following "... did ..." can intensely imply the discourse relation is CONTRAST.

Pitler et al. [2] is the first study to infer implicit discourse relations on natural data. They compared the performance of various types of word pair features, and explored the utility of several linguistically informed features including polarity tags, inquirer tags, verb classes and modality, showing lexical features are useful for recognizing implicit discourse relations. Lin et al. [3] is the first to introduce syntactic features, specifically, constituent parse features and dependency parse features. They presented an implicit discourse relation classifier on Level 2 types in the PDTB using syntactic and lexical features. [12] extended this work by applying tree kernel method to the syntactic features. [4] proposed a method that uses the language models to automatically predict implicit connectives. [13] extended this work by integrating various linguistically informed features. Park et al. [7] optimized the combination of lexical and syntactic features and achieved a solid performance.

Subsequence works focused on settling the data sparsity problem. [6] aggregated lexical feature by re-weighting word pairs. [8] introduced the simplification of the parse tree to relieve the sparsity of syntactic features. [9] employed Brown cluster to generate more compact word pair features, reducing the feature size

from quadratic in the vocabulary to $3200^2$. To date, [14] manifested the best results by training the model with elaborately collected explicit discourse relation data.

Compared to the previous work, we resort to the purely distributed representations of these discrete features. In this way, we can not only alleviate the data sparsity problem to a large extent, but also fully explore the similarities between the discrete features.

### 2.2 Distributed Representations

Word embeddings as a kind of distributed representations for words have shown successful applications in many NLP tasks including statistical language modeling, machine translation, named entity recognition, semantic role labeling, syntactical parsing and word sense disambiguation By projecting words from one-hot representation onto a much lower dimensional vector space, we can obtain the denser representations of words which have been proved contains more semantic information: vectors that are close to each other are shown to be semantically related.

Phrase and sentence distributed representations attracted much attention in recent years [15,16] applied convolutional neural networks with shallow architectures to learn sentence distributed representations for classification. This model consists of a local convolution layer and a global max-pooling layer over sentence.

Inspired by the above methods, we propose to learn appropriate distributed representations of the discrete features in different levels. Besides the surface level words and sentences, we also try to present the POS and syntactic production rules in the dense continuous vector space. In addition to the neural network based approaches, we also apply other algorithms, such as PCA, to learn the distributed representations.

## 3 Overview of the Penn Discourse Treebank

The Penn Discourse Treebank (PDTB) is the largest available discourse corpus annotated on 2,312 Wall Street Journal (WSJ) articles, where there are 40,600 instances with five main types of discourse relation labels: Explicit, Implicit, AltLex, EntRel and NoRel. The senses of discourse relations are organized into a hierarchical structure in which the top level contains four major classes: "COMPARISON", "CONTINGENCY", "EXPANSION" and "TEMPORAL". The next two levels consist of more fine-grained relation types. PDTB provides an available resource for analyzing discourse structure and detecting discourse relations.

A discourse relation instance has two components: the substring bounded by the connective is called Arg2, the remainder is Arg1. There are natural connectives such as "but", "because", "and" in explicit discourse relations, while for implicit relations, the connectives are manually inserted. Here we give an example of implicit discourse relation in PDTB style: Implicit connectives are in small caps. For the arguments, Arg1 is presented in italics, and Arg2 in bold.

Comparison: *A figure above 50 indicates the economy is likely to expand*, Implicit = WHILE **one below 50 indicates a contraction may be ahead.**

## 4 Distributed Representation Learning

The focus of our method is to learn the better representation of the semantic and syntactic features of the argument pairs for implicit discourse relation recognition without any external lexicons. The previous methods exploring discrete features (words, clusters and syntactic production rules) suffer from data sparsity problem and cannot fully use the similarity between words. We attempt to learn distributed representations for each kind of discrete features from words, sentence to deep syntactic features. We will explain our feature learning methods in detail below.

### 4.1 Representation Learning for Words/Word-Pairs

Lexical words are usually the most important features for NLP classification tasks. In implicit discourse relation recognition, almost all of the previous approaches have employed the lexical features which were proved to be effective since the work of [1]. All of the lexical feature templates are combinations of surface words, such as bigrams and skip-bigrams. Following the previous work, we focus on the most popular two types of lexical feature templates: word-pair and first-last-first3, which are explained below.

**Word-Pair**: any skip-bigram in which the first word is located in Arg1 and the second word appears in Arg2. For the sentence of Comparison relation given in Sect. 3, we can extract "A" and "one" from Arg1 and Arg2 respectively, therefore, (A, one) is a word pair feature of this sentence.

**First-Last-First3**: the first, the last and the first trigram in Arg1 and Arg2 along with the pair of the first words in Arg1 and Arg2, the pair of the last words in Arg1 and Arg2. We use an example to clarify this kind of feature. Regarding the sentence in Comparison relation given in Sect. 3, the first-last-first3 feature can be expressed as: (A, one, expand, ahead, A_figure_above, one_below_50, A_one, expand_ahead).

In spite of the validity of word pairs and first-last-first3, this kind of discrete lexical features heavily suffered from sparsity problem. For example, when we set a cutoff number of 5 to remove the infrequent word pairs, we can still extracted 123,472 word pairs in total. We aim to learn dense continuous representations to alleviate the data sparsity and explore the similarities between words, instead of regarding them as independent tokens. We investigate two directions: one is matrix factorization based methods such as Singular Value Decomposition (SVD) and Principal Component Analysis (PCA), and the other is neural network based approaches such as word2vec.

Word2vec first randomly initialize each word with a fixed low-dimensional real-valued vector. Then, in the continuous vector space, it uses the context vectors to predict the central word or utilizes the central word vector to predict the context words. The word embeddings are optimized to maximize the likelihood of the large-scale monolingual data. As the training data PDTB is very limited,

we employ other unlabeled monolingual data to enrich the training set so that we can obtain the accurate word embeddings.

With the learnt word embeddings, we concatenate them to form the representation of trigrams and other combination of lexical words. However, it is difficult to represent the word pair feature. Since sentences in different length contain different number of word pairs, concatenation of word embeddings cannot lead to a fixed-length vector representation. We thus resort to PCA.

PCA is a simple unsupervised linear algebra algorithm which can reduce a sophisticated feature matrix to a low-dimensional representation. PCA is widely used for optimal feature extraction and data compression because of the ability to uncover simplified structures that often underlie it. Therefore, we employ PCA to transform traditional one-hot vectors used in the prior work to a low-dimensional real-valued vector representation. Through a linear transformation matrix P, PCA can project an original data set X into a new set Y:

$$PX = Y \tag{1}$$

Here, X denotes discrete surface features such as word pairs (as well as first-last-first3) in binary vector whose dimension is the vocabulary size of word pairs, and Y denotes dense distributed representations whose dimension can be adjusted.

### 4.2 Representation Learning for Sentences

To determine the implicit relation between Arg1 and Arg2, the ideal method is to acquire the semantic meaning of these two sub-sentences respectively. However, it is impossible to represent the meaning of (sub-) sentences using discrete variables due to the severe data sparsity. Previous methods did not consider the sub-sentence or sentence level features. In this work, we plan to explore the effectiveness of distributed representations of (sub-) sentences.

There are several neural network based approaches to perform sentence embedding. In this work, we try two simple but effective methods: Paragraph Vector (sent2vec) [17] and convolutional neural network (CNN) [18].

Paragraph Vector is the augmentation of word2vec. Based on word2vec, the sentence embedding is also employed to predict the target word. After the optimization of the same objective as word2vec, we can get the final sentence embeddings. In order to obtain the distributed representation of Arg1 and Arg2, we also regard them as single sentences during Paragraph Vector training.

Consisting of the convolution and pooling layers, CNN provides a standard architecture which maps variable-length sentences into fixed-size distributed vectors. The CNN model takes as input the sequence of word embeddings, summarizes the sentence meaning by convolving the sliding window and pooling the saliency through the sentence, and yields the fixed-length distributed vector with other layers, such as dropout layer and fully-connected layers.

Given a sentence $w_1$, $w_2$,...,$w_t$,...,$w_T$, each word $w_t$ is first projected into a vector $x_t$. Then, we concatenate all the vectors to form the input $\mathrm{X} = [x_1, x_2, \ldots, x_t, \ldots, x_T]$.

Convolution Layer involves a number of filters $W \in R^{h \times k}$ which summarize the information of h-word window and produce a new feature. For the window of h words $X_{t:t+h-1}$ a filter $F_l$ $(1 \leq l \leq L)$ generates the feature $y_t^l$ as follows:

$$y_t^l = f(WX_{t:t+h-1} + b) \tag{2}$$

When a filter traverses each window from $X_{1:h-1}$ to $X_{T-h+1:T}$, we get the output of the feature map: $y^l = [y_1^l, y_1^2, ..., y_1^{T-t+1}]$, $(y^l \in R^{T-h+1})$. Note that the sentences differ from each other in length T, and then $y^l$ has different dimensions for different sentences. It becomes a key question how to transform the variable-length vector $y^l$ into a fixed-size vector.

Pooling Layer is designed to perform this task. In most cases, we apply a standard max-over-time pooling operation over $y^l$ and choose the maximum value $\hat{y}^l$=max$\{y^l\}$. With L filters, the dimension of the pooling layer output will be L. Using two layers of fully-connected linear layers, we obtain a fixed-length output representation.

The final fixed-length sentence representation is used to classify the implicit discourse relations in PDTB. Since the labeled training data in PDTB is not adequate, the leant sentence embedding with CNN may be not good enough. We will combine this sentence representation with the distributed representations of other kinds of features, such as word-level and syntactic-level features.

### 4.3 Representation Learning for Syntactic Features

To infer the implicit relation between Arg1 and Arg2, the structure difference between them may provide some clues. We thus consider two types of syntactic features and learn their distributed representations respectively. The first type is POS tags and the second one is syntactic production rules extracted from the constituent parse tree such as S → NP VP, VB → "have".

For POS tags, we learn their distributed representations in the same way as we have done in word embedding. After POS tagging of the training data, we just retain the POS sequence for each sentence and adopt word2vec/ Paragraph Vector to obtain their low-dimensional real-valued vectors.

For syntactic production rules, we first simplify the rules by splitting a rule with more than one child into unary rules following [19]. For example, S → NP VP becomes S → NP and S → VP. [19] showed that this heuristic can get an average of 1.04 % F-score improvement. For these production rules, we apply both PCA and word2vec to learn their distributed representations. When applying PCA, we collect all the production rules appearing in Arg1, Arg2 and the whole sentence, and then form a binary vector whose dimension is the vocabulary size of production rules. PCA will transform this high-dimensional binary vector into a low-dimensional real-valued vector. When applying word2vec, we should first transform the set of production rules in a sentence into a unique sequence. To achieve this goal, we use breadth-first traversal to transform a constituent parse tree into a unique sequence of production rules (e.g. S → NP). Each production rule is considered as a single token and the sequence is fed to word2vec.

## 5 Ensemble Learning for Implicit Discourse Relation Classification

In the previous section, we investigate the distributed representation learning for words, sentences and syntactic features with different methods (e.g. PCA, word2vec and CNN). Each kind of distributed representation can be regarded as a different view of the sentence (Arg1 and Arg2) considered. We attempt to integrate all the distributed features by ensemble learning. The ensemble learning methods can be categorized into two types based on fixed rules and trained methods. Fixed rules, such as voting rule, combine the individual outputs in a fixed manner. On the other hand, trained methods, such as weighted combination, combine outputs via training on a development dataset.

In weighted combination, the final score for class j can be expressed as follow:

$$O_j = \sum_{k=1}^{K} w_k o_{kj} \tag{3}$$

where $O_j$ denote the final score for class j, $w_k$ denote the weight of feature k, $o_{kj}$ denote the score of feature k for class j. We intuitively assign the F1 score of feature k on development dataset to $w_k$.

## 6 Experiment

### 6.1 Experimental Settings

Following the previous work [2,4,14] on implicit relation inference, we use sections 2–20 of PDTB as training data, sections 0–1 as development set and sections 21–22 as test set.

It should be noted that the data preparation for EXPANSION relations follows the work of [4,14]. It is different from the work of [2] in which they regarded EntRel relation as a part of EXPANSION.

The distribution of implicit discourse relations in PDTB is highly skewed, especially for COMPARISON and TEMPORAL, which only occupy 15.06 % and 5.72 % of the total. [2] randomly chose the negative instances to balance the number of positive, but this down-sampling approach was proved throwing away useful information. [19] presented several strategies to address this problem by including down-sampling, up-sampling and weight-cost. The experiments showed that weight-cost is better than re-sampling. [9] employed a similar method as weight-cost by re-weighting the instance in each class so that the positive and negative classes acquire the same weights in total. We adopt this method in our experiments to achieve balance between positive and negative instances.

For evaluating effect of syntactic feature in real-world data, we didn't use the gold standard parse results provided by the Penn Treebank. Our constituent parse results are obtained by Stanford Parser, and POS tags are generated by nltk package. We also employ lowercasing, stemming and tokenization. Additionally, when we are training with discrete features, we follow the previous

methods and remove the discrete features whose occurrence number lower than a cut-off, which is 3, 5, and 5 for first-last-first3, words pairs and production rules respectively.

When we learn distributed representations with word2vec and sent2vec models, we remove the words and production rules which only appear once in the dataset. To enlarge the data scale for word/sent2vec training, we employ a large-scale unlabeled monolingual data from Reuters. From the raw Reuters data, we choose only the sentences in which all the words should appear in PDTB so as to avoid noise. The chosen Reuters corpus contains 1.7 billion tokens and 67.2 million sentences.

We have studied the effects of parameter set in word/sent2vec including the dimension of vectors, the size of context windows, the number of negative sample tokens and the learning rate. We found that the dimension of vector has a significant influence on the final performance, and 25 is a satisfactory size. To make a fair comparison, the dimension of the vectors acquired via PCA is set the same as word/sent2vec.

For CNN, we use one convolution layer, one max-pooling layer, one dropout layer and two fully-connected linear layers. We adopt sigmoid function for non-linear projection. In the convolution layer, we set the window h=3 and employ F=100 filters. We set the dropout ratio 0.5 in the dropout layer. The dimension of two fully-connected layers is set 50 and the output length is set 25.

Given the distributed representations, we compare the performance between SVM from libsvm and MaxEnt from MALLET. We found that these two classifiers perform similarly. Our experimental results that we reported in this paper are obtained by linear model in SVM with the parameters set to default values.

### 6.2 Experimental Results

In this section, we try to answer four questions: (1) whether distributed representation is better than discrete features in implicit relation recognition; (2) which kinds of features are more effective; (3) which method for distributed representation learning performs better; and (4) whether can our method surpass the best reported results. The detailed experimental results listed in Table 1 can answer the first three questions. We can see from Table 1 that distributed representations can substantially outperform the discrete features in most cases. The largest improvement can be up to 16.43 F1-score for CONTINGENCY relation inference using syntactic production rules.

When we focus on the rows in Table 1, we can find that different kinds of features con-tribute very differently to the final performance. Overall, the syntactic features perform stably over the four relations and achieve the best performance in most cases. The sentence-level features can obtain good performance as well, indicating that representations of Arg1, Arg2 and the whole sentence are helpful in implicit relation recognition.

When we look at the columns in Table 1, we will see that word/sent2vec wins 3 out of 4 relation recognition tasks. The exception is the EXPANSION relation task, in which CNN performs best.

**Table 1.** The performance (F1-score) of discrete and distributed feature on recognition four classes of implicit discourse relations. Disc. denotes discrete features, P.V. denotes the distributed featured acquired by Paragraph vector model (can perform word2vec and sent2vec). Scores marked by "∗" are significantly better ($p < 0.05$; t-test) than counterpart discrete features.

| COMPARISON | | | | | CONTINGENCY | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Features | Disc | PCA | P.V | CNN | Features | Disc | PCA | P.V | CNN |
| Word Pair | 26.71 | 29.24 | - | - | Word Pair | 34.48 | 47.92 | - | - |
| First-Last-First3 | 25.55 | 30.37 | - | - | First-Last-First3 | 34.67 | 45.64 | - | - |
| Word-sentence | 27.75 | 30.79* | 37.81* | 23.81 | Word-sentence | 36.36 | 47.96* | 48.58* | 33.91 |
| Production Rules | 27.81 | 34.68* | 37.57* | 18.83 | Production Rules | 35.09 | 49.32* | 51.52* | 32.67 |
| POS | - | - | 35.37 | - | POS | - | - | 51.52 | - |
| EXPANSION | | | | | TEMPORAL | | | | |
| Features | Disc | PCA | P.V | CNN | Features | Disc | PCA | P.V | CNN |
| Word Pair | 60.18 | 64.22 | - | - | Word Pair | 20.93 | 15.75 | - | - |
| First-Last-First3 | 56.22 | 61.71 | - | - | First-Last-First3 | 12.50 | 13.90 | - | - |
| Word-sentence | 59.75 | 64.7* | 65.44* | 69.79* | Word-sentence | 24.32 | 17.18 | 26.19* | 9.71 |
| Production Rules | 60.82 | 66.00* | 65.21* | 69.28* | Production Rules | 17.98 | 28.97* | 29.72* | 10.77 |
| POS | - | - | 67.13 | - | POS | - | - | 25.48 | - |

The final ensemble results shown in Table 2 answers the last question. In ensemble learning, we apply the weighted combination approach. We investigate three-level ensemble models: we first ensemble features learnt by word/sent2vec models because they perform best in Table 1. Then we add PCA method. Finally, we combine all the distributed features learnt by word/sent2vec, PCA and CNN. The experimental results in Table 2 tell us that our method can achieve the best performance in three out of three relation recognition tasks when compared to the state-of-the-art approaches using discrete features. It gets the improvement over the state-of-the-art by 1.34, 1.31 and 1.7 F-score respectively. In the COMPARISON task, we also obtain a competitive result. These results demonstrate that our ensemble method with distributed representations is very promising for implicit discourse relation inference.

### 6.3 Discussion

Distributed representations learned by word/sent2vec and CNN model outperform the discrete features by a larger margin, and achieve the best results against the state-of-the-art except the relation of Expansion. We analyze the gap of F1 score in Expansion relation and find that the majority of performance comes from the polarity tag features used in the prior work. It is verified by [2,6] that polarity tags have a remarkable effect for inferring Expansion relations, while our model seems insensitive to polarity. For example, by calculating the cosine distance between the word vectors trained by word2vec, "good" is most similar to "bad". Similarly, others include "better" to "worse", "positive" to "negative"

**Table 2.** Experimental results of our system.

| | | Com. vs Not | Con. vs Not | Exp. vs Not | Tem. vs Not |
|---|---|---|---|---|---|
| Pitler et al. (2009) | | 21.96 | 47.13 | - | 16.76 |
| Zhou et al. (2010) | | 31.79 | 47.16 | 65.95 | 20.30 |
| Rutherford and Xue (2014) | | 39.70 | 54.42 | 70.23 | 28.69 |
| Rutherford and Xue (2015) | | **41.00** | 53.80 | 69.41 | 33.30 |
| prospose | P.V. | 40.55 | 53.52 | 67.67 | **35.00** |
| | P.V.+PCA | 39.10 | **55.14** | 69.46 | 28.91 |
| | P.V.+PCA+CNN | 38.52 | 54.15 | **70.71** | 31.51 |

and "can" to "cannot". The reason is that we use context-based models and these words share the similar contexts.

Table 1 shows that principal component analysis (PCA), acquire substantially improvement over discrete counterpart. And the distributed representations learned by PCA achieve a similar performance to word/sent2vec model except for TEMPORAL relation, verifying the finding of [20] that word2vec is proved actually implicitly factorization of word-context matrix. F1 score decline for TEMPORAL relation may be due to the smallest proportion of TEMORAL instances in the whole dataset.

## 7 Conclusion and Future Work

In this paper, we have proposed a novel method for implicit discourse relation recognition using purely distributed representations without relying on any external linguistic resources, such as WordNet. We presented different approaches (e.g. word2vec, CNN and PCA) to learn distributed vectors for discrete features in different levels from words, sentences to syntactic features. Finally, we adopted the ensemble learning algorithm to integrate all of the distributed representations in various linguistic levels.

The experimental results show that dense low-dimensional representations have advantage over sparse discrete counterparts. Our final system obtain the best performance for CONTINGENCY, EXPANSION and TEMPORAL relations, while keeps competitive in COMPARISON relation compared to the state-of-the-art methods. In the future, we will devote ourselves to encode more linguistically features into distributed representation and explore more effective method to learn long-distance dependency of the syntax and semantics.

**Acknowledgments.** The research work has been partially funded by the Natural Science Foundation of China under Grant No. 61333018 and No. 61402478.

## References


1. Pitler, E., Raghupathy, M., Mehta, H., Nenkova, A., Lee, A., Joshi, A.K.: Easily identifiable discourse relations. In: COLING (2008)
2. Pitler, E., Louis, A., Nenkova, A.: Automatic sense prediction for implicit discourse relations in text. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, vol. 2, pp. 683–691. Association for Computational Linguistics (2009)
3. Lin, Z., Kan, M.-Y., Ng, H.T.: Recognizing implicit discourse relations in the penn discourse treebank. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, vol. 1, pp. 343–351. Association for Computational Linguistics (2009)
4. Zhou, Z.-M., Xu, Y., Niu, Z.-Y., Lan, M., Su, J., Tan, C.L.: Predicting discourse connectives for implicit discourse relation recognition. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters. Association for Computational Linguistics, pp. 1507–1514 (2010)
5. Louis, A., Joshi, A., Prasad, R., Nenkova, A.: Using entity features to classify implicit discourse relations. In: Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 59–62. Association for Computational Linguistics (2010)
6. Biran, O., McKeown, K.: Aggregated word pair features for implicit discourse relation disambiguation. In: Proceedings of the Conference ACL, p. 69 (2013)
7. Park, J., Cardie, C.: Improving implicit discourse relation recognition through feature set optimization. In: Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 108–112. Association for Computational Linguistics (2012)
8. Li, J.J., Nenkova, A.: Reducing sparsity improves the recognition of implicit discourse relations. In: 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, p. 199 (2014)
9. Rutherford, A.T., Xue, N.: Discovering implicit discourse relations through brown cluster pair representation and coreference patterns. EACL **2014**, 645 (2014)
10. Marcu, D., Echihabi, A.: An unsupervised approach to recognizing discourse relations. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 368–375. Association for Computational Linguistics (2002)
11. Saito, M., Yamamoto, K., Sekine, S.: Using phrasal patterns to identify discourse relations. In: Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers, pp. 133–136. Association for Computational Linguistics (2006)
12. Wang, W., Su, J., Tan, C.L.: Kernel based discourse relation recognition with temporal ordering information. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 710–719. Association for Computational Linguistics (2010)
13. Xu, Y., Lan, M., Lu, Y., Niu, Z.Y., Tan, C.L.: Connective prediction using machine learning for implicit discourse relation classification. In: The 2012 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2012)

14. Rutherford, A., Xue, N.: Improving the inference of implicit discourse relations via classifying explicit discourse connectives. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Denver, Colorado: Association for Computational Linguistics, pp. 799–808, May-June 2015. http://www.aclweb.org/anthology/N15-1081
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
16. Kim, Y.: Convolutional neural networks for sentence classification (2014). arXiv preprint arXiv:1408.5882
17. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents (2014) arXiv preprint. arXiv:1405.4053
18. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences (2014) arXiv preprint. arXiv:1404.2188
19. Li, J.J., Nenkova, A.: Addressing class imbalance for improved recognition of implicit discourse relations. In: 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, p. 142 (2014)
20. Levy, O., Goldberg, Y., Dagan, I.: Improving distributional similarity with lessons learned from word embeddings. Trans. Assoc. Comput. Linguist. **3**, 211–225 (2015)