Implicit Discourse Relation Detection via a Deep Architecture with Gated Relevance Network

Jifan Chen, Qi Zhang, Pengfei Liu, Xipeng Qiu, Xuanjing Huang
Shanghai Key Laboratory of Data Science
School of Computer Science, Fudan University
825 Zhangheng Road, Shanghai, China
jfchen14,qz,pfliu14,xpqiu,xjhuang@fudan.edu.cn

Abstract

Word pairs, which are one of the most easily accessible features between two text segments, have been proven to be very useful for detecting the discourse relations held between text segments. However, because of the data sparsity problem, the performance achieved by using word pair features is limited. In this paper, in order to overcome the data sparsity problem, we propose the use of word embeddings to replace the original words. Moreover, we adopt a gated relevance network to capture the semantic interaction between word pairs, and then aggregate those semantic interactions using a pooling layer to select the most informative interactions. Experimental results on Penn Discourse Tree Bank show that the proposed method without using manually designed features can achieve better performance on recognizing the discourse level relations in all of the relations.

1 Introduction

In a well-written document, no unit of the text is completely isolated, discourse relations describe how two units (e.g. clauses, sentences, and larger multi-clause groupings) of discourse are logically connected. Many downstream NLP applications such as opinion mining, summarization, and event detection, can benefit from those relations.

The task of automatically identify discourse relation is relatively simple when explicit connectives such as *however* and *because* are given (Pitler et al., 2009). However, the identification becomes much more challenging when such connectives are missing. In fact, such implicit discourse relations outnumber explicit relations in naturally occurring text, and identify those relations have been shown to be the performance bottleneck of an end-to-end discourse parser (Lin et al., 2014).

Most of the existing researches used rich linguistic features and supervised learning methods to achieve the task (Soricut and Marcu, 2003; Pitler et al., 2009; Rutherford and Xue, 2014). Among their works, word pairs are heavily used as an important feature, since word pairs like *(warm,cold)* might directly trigger a contrast relation. However, because of the data sparsity problem (McKeown and Biran, 2013) and the lack of metrics to measure the semantic relation between those pairs, which is so-called the *semantic gap* problem (Zhao and Grosky, 2002), the classifiers based on word pairs in the previous studies did not work well. Moreover, some text segment pairs are more complicated, it is hard to determine the relation held between them using only word pairs. Consider the following sentence pair with a *casual* relation as an example:

S1: Psyllium's not a good crop.

S2: You get a rain at the wrong time and the crop is ruined.

Intuitively, *(good, wrong)* and *(good, ruined)*, seem to be the most informative word pairs, and it is likely that they will trigger a contrast relation. Therefore, we can see that another main disadvantage of using word pairs is the lack of contextual information, and using n-gram pairs will again suffer from data sparsity problem.

Recently, the distributed word representations (Bengio et al., 2006; Mikolov et al., 2013) have shown an advantage when dealing with data sparsity problem (Braud and Denis, 2015), and many deep learning based models are generating substantial interests in text semantic matching and have achieved some significant progresses (Hu et al., 2014; Qiu and Huang, 2015; Wan et al.,

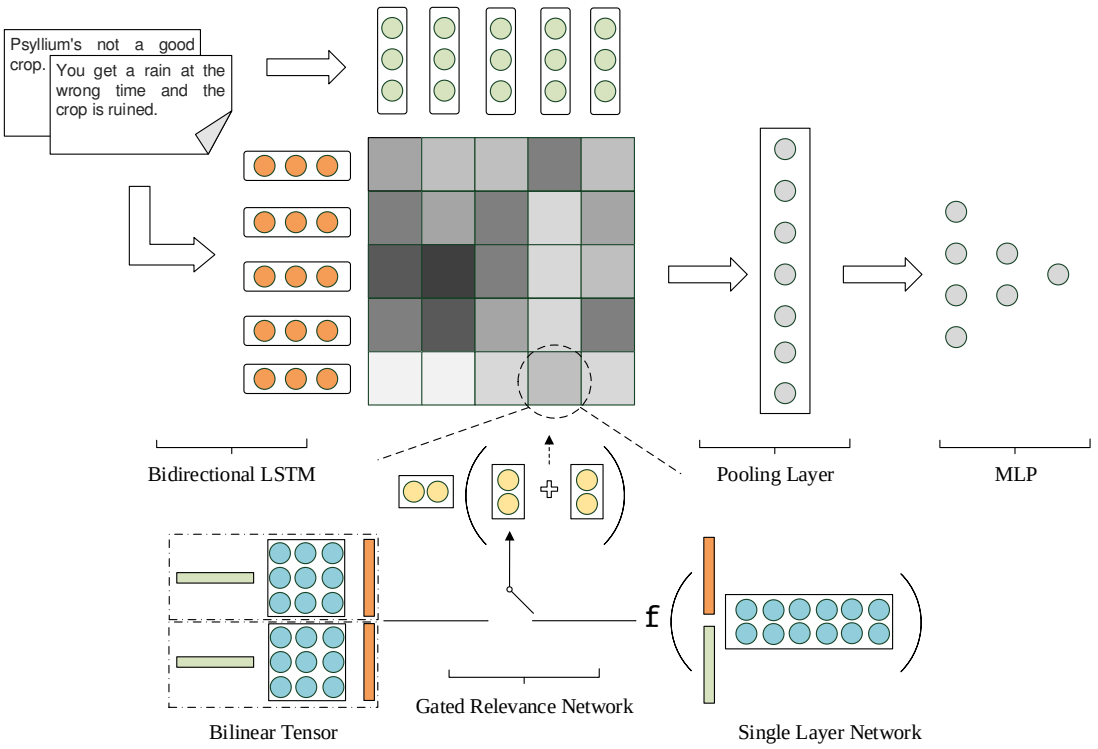


Figure 1: The processing framework of the proposed approach.

2015). Inspired by their work, we in this paper propose the use of word embeddings to replace the original words in the text segments to fight against the data sparsity problem. Further more, in order to preserve the contextual information around the word embeddings, we encode the text segment to its positional representation via a recurrent neural network, specifically, we use a bidirectional LSTM (Hochreiter and Schmidhuber, 1997). Then, to overcome the *semantic gap*, we propose the use of a gated relevance network to capture the semantic interaction between those positional representations. Finally, all the interactions generated by the relevance network are fed to a max pooling layer to get the strongest interactions. We then aggregate them to predict the discourse relation through a multi-layer perceptron (MLP). Our model is trained end to end by Back-Propagation and Adagrad.

The main contribution of this paper can be summarized as follows:

- We use word embeddings to replace the original words in the text segments to overcome data sparsity problem. In order to preserve the contextual information, we further encode the text segment to its positional representation through a recurrent neural network.
- To deal with the semantic gap problem, we adopt a gated relevance network to capture the semantic interaction between the intermediate representations of the text segments.
- Experimental results on PDTB (Prasad et al., 2008) show that the proposed method can achieve better performance in recognizing discourse level relations in all of the relations than the previous methods.

2 The Proposed Method

The architecture of our proposed method is shown in figure 1. In the following of this section, we will illustrate the details of the proposed framework.

2.1 Embedding Layer

To model the sentences with neural model, we firstly need to transform the one-hot representation of word into the distributed representation. All words of two text segments X and Y will be mapped into low dimensional vector representations, which are taken as input of the network.

Through this layer, we can filter the words appear in low frequency, and we then map these words to a special OOV (out of vocabulary) word embedding. In addition, all the text segments in our experiment are padded to have the same length.

2.2 Sentence Modeling with LSTM

Long Short-Term Memory network (LSTM) (Hochreiter and Schmidhuber, 1997) is a type of recurrent neural network (RNN), and specifically addresses the issue of learning long-term dependencies. Given a variable-length sentence $S = (x_0, x_1, ..., x_T)$, LSTM processes it by incrementally adding up new content into a single slot of maintained memory, with gates controlling the extent to which new content should be memorized, old content should be erased and current content should be exposed. At position t, the memory c_t and the hidden state h_t are updated with the following equations:

$$\begin{bmatrix} \tilde{\mathbf{c}}_t \\ \mathbf{o}_t \\ \mathbf{i}_t \\ \mathbf{f}_t \end{bmatrix} = \begin{bmatrix} \tanh \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} T_{\mathbf{A},\mathbf{b}} \begin{bmatrix} \mathbf{x}_t \\ \mathbf{h}_{t-1} \end{bmatrix}, \quad (1)$$

$$\mathbf{c}_t = \tilde{\mathbf{c}}_t \odot \mathbf{i}_t + \mathbf{c}_{t-1} \odot \mathbf{f}_t, \quad (2)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \quad (3)$$

where i_t, f_t, o_t, denote the input, forget and output gate at time step t respectively, and $T_{\mathbf{A},\mathbf{b}}$ is an affine transformation which depends on parameters of the network $\mathbf{A}$ and $\mathbf{b}$. σ denotes the logistic sigmoid function and $\odot$ denotes elementwise multiplication.

Notice that the LSTM defined above only get context information from the past. However, context information from the future could also be crucial. To capture the context from both past and the future, we propose to use the bidirectional LSTM (Schuster and Paliwal, 1997). Bidirectional LSTM preserves the previous and future context information by two separate LSTMs, one encodes the sentence from start to the end, and the other encodes the sentence from end to the start. Therefore, at each position t of the sentence, we can obtain two representations $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$. It is natural to concatenate them to get the intermediate representation at position t, i.e. $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$. A illustration for the bidirectional LSTM are shown in Figure 2.

Given a sentence $S = (x_0, x_1, ..., x_T)$, we can now encode it with a bidirectional LSTM, and replace the word w_t with h_t, we can interpret h_t as a representation summarizing the word at position t and its contextual information.

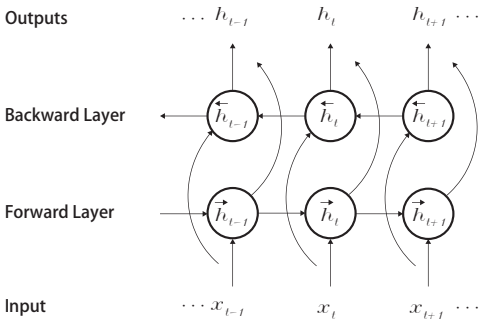


Figure 2: A illustration of bidirectional LSTM.

2.3 Gated Relevance Network

Given two text segments $X = x_1, x_2, ..., x_n$ and $Y = y_1, y_2, ..., y_m$, after the encoding procedure with a bidirectional LSTM, we can get their positional representation $X_h = x_{h_1}, x_{h_2}, ..., x_{h_n}$ and $Y_h = y_{h_1}, y_{h_2}, ..., y_{h_m}$. We then compute the relevance score between every intermediate representation pair x_{h_i} and y_{h_j} with dimension d_h. Traditional ways to measure their relevance includes cosine distance, bilinear model (Sutskever et al., 2009; Jenatton et al., 2012), single layer neural network (Collobert and Weston, 2008), etc. We then illustrate the bilinear model and the single layer neural network in details.

Bilinear Model is defined as follows:

$$s(h_{x_i}, h_{y_j}) = h_{x_i}^T M h_{y_j}, \quad (4)$$

where the only parameter $M \in \mathbf{R}^{d_h \times d_h}$. The bilinear model is a simple but efficient way to incorporate the strong linear interactions between two vectors, while the main weakness of it is the lack of ability to deal with nonlinear interaction.

Single Layer Network is defined as:

$$s(h_{x_i}, h_{y_j}) = u^T f(V \begin{bmatrix} h_{x_i} \\ h_{y_j} \end{bmatrix} + b), \quad (5)$$

where f is a standard nonlinearity applied element-wise, $V \in \mathbf{R}^{k \times 2d_h}$, $b \in \mathbf{R}^k$, and $u \in \mathbf{R}^k$. The single layer network could capture nonlinear interaction, while at the expense of a weak interaction between two vectors.

Each of the two models have its own advantages, and they can not take the place of each other.

In our work, we propose to incorporate the two models through the gate mechanism, so that the model is more powerful to capture more complex semantic interactions. The incorporated model, namely **gated relevance network** (GRN), is defined as:

$$s(h_{x_i}, h_{y_j}) = u^T(g \odot h_{x_i}^T M^{[1:r]} h_{y_j} + (1-g) \odot f(V \begin{bmatrix} h_{x_i} \\ h_{y_j} \end{bmatrix}) + b), \quad (6)$$

where f is a standard nonlinearity applied element-wise, $M^{[1:r]} \in \mathbf{R}^{r \times d_h \times d_h}$ is a bilinear tensor and the tensor product $h_{x_i}^T M^{[1:r]} h_{y_j}$ results in a vector $m \in \mathbf{R}^r$, where each entry is computed by one slice $k = 1, 2, ..., r$ of the tensor: $m_k = h_{x_i}^T M^k h_{y_j}$, $V \in \mathbf{R}^{r \times 2d_h}$, $b \in \mathbf{R}^r$, and $u \in \mathbf{R}^r$, g is a gate expressing how the output is produced by the linear and nonlinear semantic interactions between the input, defined as:

$$g = \sigma(W_g \begin{bmatrix} h_{x_i} \\ h_{y_j} \end{bmatrix} + b_g), \quad (7)$$

where $W_g \in \mathbf{R}^{r \times 2d_h}$, $b \in \mathbf{R}^r$ and σ denotes the logistic sigmoid function.

The gated relevance network is a little bit similar to the Neural Tensor Network (NTN) proposed by Socher et al. (2013):

$$s(h_{x_i}, h_{y_j}) = u^T f(h_{x_i}^T M^{[1:r]} h_{y_j} + V \begin{bmatrix} h_{x_i} \\ h_{y_j} \end{bmatrix} + b). \quad (8)$$

Compared with NTN, the main advantage of our model is we use a gate to tell how the linear and nonlinear interaction should be combined, while in NTN, the interaction generated by bilinear model and single layer network are treated equally. Also, NTN feeds the incorporated interaction to a nonlinearity, while we are not.

As we can see, for each pair of the intermediate representation, the gated relevance network will produce a semantic interaction score, thus, the entire output of two text segments is an interaction score matrix.

2.4 Max-Pooling Layer and MLP

The relation between two text segments is often determined by some strong semantic interactions, therefore, we adopt max-pooling strategy which partitions the score matrix as shown in Figure 1 into a set of non-overlapping sub-regions, and for each such sub-region, outputs the maximum value. The pooling scores are further reshaped to a vector and fed to a multi-layer perceptron (MLP). More specifically, the vector obtained by the pooling layer is fed into a full connection hidden layer to get a more abstractive representation first, and then connect to the output layer. For the task of classification, the outputs are probabilities of different classes, which is computed by a softmax function after the fully-connected layer. We name the full architecture of our model **Bi-LSTM+GRN**.

2.5 Model Training

Given a text segment pair (X, Y) and its label l, the training objective is to minimize the cross-entropy of the predicted and the true label distributions, defined as:

$$L(X, Y; \boldsymbol{l}, \hat{\boldsymbol{l}}) = -\sum_{j=1}^{C} \boldsymbol{l}_j \log(\hat{\boldsymbol{l}}_j), \quad (9)$$

where $\boldsymbol{l}$ is one-hot representation of the ground-truth label l; $\hat{\boldsymbol{l}}$ is the predicted probabilities of labels; C is the class number.

To minimize the objective, we use stochastic gradient descent with the diagonal variant of AdaGrad (Duchi et al., 2011) with minibatches. The parameter update for the i-th parameter $\theta_{t,i}$ at time step t is as follows:

$$\theta_{t,i} = \theta_{t-1,i} - \frac{\alpha}{\sqrt{\sum_{\tau=1}^{t} g_{\tau,i}^2}} g_{t,i}, \quad (10)$$

where α is the initial learning rate and $g_\tau \in \mathbf{R}^{|\theta_{\tau,i}|}$ is the gradient at time step τ for parameter $\theta_{\tau,i}$.

3 Experiment

3.1 Dataset

The dataset we used in this work is Penn Discourse Treebank 2.0 (Prasad et al., 2008), which is one of the largest available annotated corpora of discourse relations. It contains 40,600 relations, which are manually annotated from the same 2,312 Wall Street Journal (WSJ) articles as the Penn Treebank. We follow the recommended section partition of PDTB 2.0, which is to use sections 2-20 for training, sections 21-22 for testing and the other sections for validation (Prasad et al., 2008). For comparison with the previous work (Pitler et al., 2009; Zhou et al., 2010; Park

Table 1: The unbalanced sample distribution of PDTB.

Relation	Train	Dev	Test
Comparison	1894	401	146
Contingency	3281	628	276
Expansion	6792	1253	556
Temporal	665	93	68

Table 2: Hyperparameters for our model in the experiment.

Word Embedding size	$n_w = 50$
Initial learning rate	$\rho = 0.01$
Minibatch size	$m = 32$
Pooling Size	$(p, q) = (3, 3)$
Number of tensor slice	$r = 2$

and Cardie, 2012; Rutherford and Xue, 2014; Ji and Eisenstein, 2015), we train four binary classifiers to identify each of the top level relations, the *EntRel* relations are merged with *Expansion* relations. For each classifier, we use an equal number of positive and negative samples as training data, because each of the relations except *Expansion* is infrequent (Pitler et al., 2009) as what shows in Table 1. The negative samples were chosen randomly from training sections 2-20.

3.2 Experiment Protocols

In this part, we will mainly introduce the experiment settings, including baselines and parameter setting.

3.2.1 Baselines

The baselines for comparison with our proposed method are listed as follows:

- LSTM: We use two single LSTM to encode the two text segments, then concatenate them and feed to a MLP to do the relation detection.
- Bi-LSTM: We use two single bidirectional LSTM to encode the two text segments, then concatenate them and feed to a MLP to do the relation detection.
- Word+NTN: We use the neural tensor defined in (8) to capture the semantic interaction scores between every word embedding pair, the rest of the method is the same as our proposed method.
- LSTM+NTN: We use two single LSTM to generate the positional text segments representation. The rest of the method is the same as Word-NTN.
- BLSTM+NTN: We use two single bidirectional LSTM to generate the positional text segments representation. The rest of the method is the same as Word-NTN.
- Word+GRN: We use the gated relevance network proposed in this paper to capture the semantic interaction scores between every word embedding pair of the two text segments. The rest of the method is the same as our model.
- LSTM+GRN: We use the gated relevance network proposed in this paper to capture the semantic interaction scores between every intermediate representation pair of the two text segments generated by LSTM. The rest of the method is the same as our model.

3.2.2 Parameter Setting

For the initialization of the word embeddings used in our model, we use the 50-dimensional pre-trained embeddings provided by Turian et al. (2010), and the embeddings are fixed during training. We only preserve the top 10,000 words according to its frequency of occurrence in the training data, all the text segments are padded to have the same length of 50, the intermediate representations of LSTM are also set to 50. The other parameters are initialized by randomly sampling from uniform distribution in [-0.1,0.1].

For other hyperparameters of our proposed model, we take those hyperparameters that achieved best performance on the development set, and keep the same parameters for other competitors. The final hyper-parameters are show in Table 2.

3.3 Result

The results on PDTB are show in Table3, from the results, we have several experiment findings.

First of all, it is easy to notice that LSTM and Bi-LSTM achieve lower performance than all of the methods of using a tensor to capture the semantic interactions between word pairs and the intermediate representation pairs. Because the main

Table 3: The performances of different approaches on the PDTB.

	Comparison	Contingency	Expansion	Temporal
(Pitler et al., 2009)	21.96%	47.13%	76.42%	16.76%
(Zhou et al., 2010)	31.79%	47.16%	70.11%	20.30%
(Park and Cardie, 2012)	31.32%	49.82%	79.22%	26.57%
(Rutherford and Xue, 2014)	39.70%	54.42%	80.44%	28.69%
(Ji and Eisenstein, 2015)	35.93%	52.78%	80.02%	27.63%
LSTM	31.78%	45.39%	75.10%	19.65%
Bi-LSTM	31.97%	45.66%	75.13%	20.02%
Word+NTN	32.18%	46.45%	77.64%	21.60%
LSTM+NTN	36.82%	50.09%	79.88%	26.54%
Bi-LSTM+NTN	39.36%	53.74%	80.02%	28.41%
Word+GRN	32.67%	46.52%	77.68%	21.21%
LSTM+GRN	38.13%	52.25%	79.96%	27.15%
Bi-LSTM+GRN	**40.17**%	**54.76**%	**80.62**%	**31.32**%

disadvantage of using LSTM and Bi-LSTM to encode the text segment into a single representation is that some important local information such as key words can not be fully preserved when compressing a long sentence into a single representation.

Second, the performance improves a lot when using LSTM and Bi-LSTM to encode the text segments to positional representations instead of using word representations directly. We conclude it is mainly because the following two reasons: for one thing, some words are important only when they are associated with their context, for the other, the intermediate representations are the high-level representation of the sentence at each position, there is no doubt for they can obtain much more semantic information than the words along. In addition, Bi-LSTM also takes the future information of the text segments into consideration, resulting in a consistently better performance than LSTM.

Third, take a comparison to the methods using NTN and the methods using GRN, we can find that the GRN performs consistently better. Such results show that the gate we proposed to combine the information of two aspects is actually useful.

At last, our proposed model, namely, Bi-LSTM+GRN achieves best performance on all of the relations. It not only shows the interaction between word pairs is useful, but also shows the way we proposed to capture such information is useful too. Further more, compared with the previous methods (Pitler et al., 2009; Park and Cardie, 2012; Rutherford and Xue, 2014), which used either a lot of complex textual features and contextual information about the two text segments or a larger unannotated corpus to help the prediction, our model only uses the the information of the two text segments themselves, but yet achieves better performance. It demonstrates that our model is powerful in modeling the discourse relations.

3.4 Parameter Sensitivity

In this section, we evaluate our model through different settings of the proposed gated relevance network, the other hyperparameters are the same as mentioned above and we use a bidirectional LSTM to encode the text segments. The settings are shown as follows:

- **GRN-1** We set the parameters $r = 1$, $M^{[1]} = I$, $V = 0$, $b = 0$ and $g = 1$. The model can be regarded as cosine similarity.

	Cmp	Ctg	Exp	Tmp
GRN-1	34.01%	50.23%	78.66%	21.33%
GRN-2	35.74%	50.91%	79.40%	23.28%
GRN-3	34.15%	51.78%	79.33%	26.16%
GRN-4	37.18%	52.36%	79.86%	25.60%
GRN-5	**40.17**%	**54.76**%	**80.62**%	**31.32**%
GRN-6	38.26%	52.08%	80.02%	28.51%

Table 4: Comparison of our model with different parameter settings to the gated relevance network. *Cmp* denotes the comparison relation, *Ctg* denotes the contingency relation, *Exp* denotes the expansion relation and *Tmp* denotes the temporal relation.

- **GRN-2** We set the parameters $r = 1$, $V = 0$, $b = 0$, and $g = 1$. The model can be regarded as the bilinear model.
- **GRN-3** We set the parameters $r = 1$, $M^{[1]} = 0$, and $g = 0$. The model can be regarded as a single layer network.
- **GRN-4** We set the parameters $r = 1$. This model is the full GRN model.
- **GRN-5** We set the parameters $r = 2$. This model is the full GRN model.
- **GRN-6** We set the parameters $r = 3$. This model is the full GRN model.

The results for different parameter settings are shown in Table 4. It is obvious that **GRN-1** achieves a relatively lower performance, showing that the cosine similarity is not enough to capture the complex semantic interaction. Take a comparison on **GRN-2** and **GRN-3**, we can see that **GRN-2** outperforms **GRN-3** on *Comparison* and *Expansion* relation, while achieves a lower performance on the other two relations, moreover, the combination method **GRN-4** outperforms both of the methods, demonstrating that the semantic interactions captured by the bilinear model and the single layer network are different. Hence, they can not take the place of each other, and it is reasonable to use a gate to combine them.

Among the methods of using the full GRN model, **GRN-5** which has 2 bilinear tensor slices achieves the best performance. We explain this phenomenon on two aspects, on one hand, we can see each slice of the bilinear tensor as being responsible for one type of the relation, a bilinear tensor with 2 slices is more suitable for training a binary classifier than the original bilinear model. On the other hand, increasing the number of slices will increase the complexity of the model, thus making it harder to train.

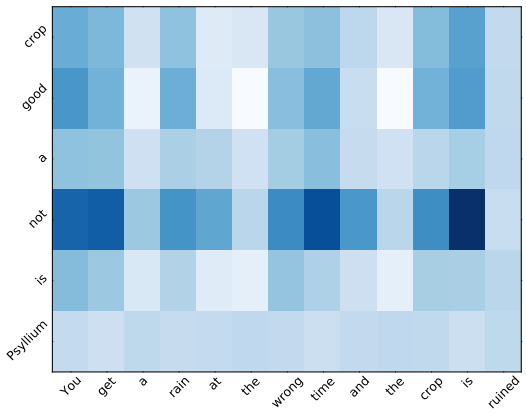
3.5 Case Study

In this section, we go back to the example mentioned above to show see what information between the text segment pairs is captured, and how the positional sentence representations affect the performance of our model.

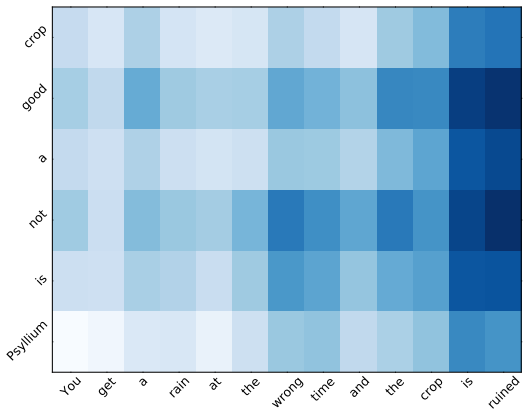
The examples is listed below:

S1: Psyllium's not a good crop.

S2: You get a rain at the wrong time and the crop is ruined.



(a) Word+GRN



(b) Bi-LSTM+GRN

Figure 3: A visualization of the interaction score matrix between two relatively complex sentences. The darker patches denote the corresponding scores generated by the gated relevance network are higher.

In this case, the relation between the sentence pair is *Contingency*, and the implicit connective annotated by human is "because". The pair is likely to be classified to a wrong contrast relation if we only focus on the informative word pairs *(good,wrong)* and *(good,ruined)*. It is mainly because their relation is highly depended on the semantic of the whole sentence, and the words should be considered with their context.

Figure 3 explains this phenomenon, in Figure 3a, we can see that the word pairs which associate with "not" get high scores, scores on the other pairs are relatively arbitrary. It demonstrates the word embedding model failed to learn which part of the sentence should be focused, although the useless word such as "Psyllium" and "a" are ignored, thus making it harder to identify the rela-

tion.

Take Figure 3b for a comparison, from the figure we can observe the pairs that associate with "not" and "good" which are import context to determine the semantic of the sentence get much higher scores. Moreover, the scores increase along with the sentence encoding procedure, especially when the last informative word "ruined" appears. Once again, some useless word are also ignored by this model. It demonstrates the bidirectional LSTM we used in our model could encode the contextual information to the intermediate representations, thus these information could help to determine which part of the two sentence should be focused when identifying their relation.

4 Related Work

Discourse relations, which link clauses in text, are used to represent the overall text structure. Many downstream NLP tasks such as text summarization, question answering, and textual entailment can benefit from the task. Along with the increasing requirement, many works have been constructed to automatically identify these relations from different aspects (Pitler et al., 2008; Pitler et al., 2009; Zhou et al., 2010; McKeown and Biran, 2013; Rutherford and Xue, 2014; Xue et al., 2015).

For training and comparing the performance of different methods, the Penn Discourse Treebank (PDTB) 2.0, which is large annotated discourse corpuses, were released in 2008 (Prasad et al., 2008). The annotation methodology of it follows the lexically grounded, predicate-argument approach. In PDTB, the discourse relations were predefined by Webber (2004). PDTB-styled discourse relations hold in only a local contextual window, and these relations are organized hierarchically. Also, every relation in PDTB has either an *explicit* or an *implicit* marker. Since explicit relations are easy to identify (Pitler et al., 2008), existing methods achieved good performance on the relations with explicit maker. In recent years, researchers mainly focused on implicit relations. For easily comparing with other methods, in this work, we also use PDTB as the training and testing corpus.

As we mentioned above, various approaches have been proposed to do the task. Pitler et al. (2009) proposed to train four binary classifiers using word pairs as well as other rich linguistic features to automatically identify the top-level PDTB relations. Park and Cardie (2012) achieved a higher performance by optimizing the feature set. McKeown and Biran (2013) aims at solving the data sparsity problem, and they extended the work of Pitler et al. (2009) by aggregating word pairs. Rutherford and Xue (2014) used Brown clusters and coreferential patterns as new features and improved the baseline a lot. Braud and Denis (2015) compared different word representations for implicit relation classification. The word pairs feature have been studied by all of the work above, showing its importance on discourse relation. We follow their work, and incorporate word embedding to deal with this problem.

There also exist some work performing this task from other perspectives. Zhou et al. (2010) studied the problem from predicting *implicit marker*. They used a language model to add implicit markers as an additional feature to improve performance. Their approach can be seen as a semi-supervised method. Ji and Eisenstein (2015) computes distributed meaning representations for each discourse argument by composition up the syntactic parse tree. Chen et al. (2016) used vector offsets to represent this relation between sentence pairs, and aggregate this offsets through the Fisher vector. Liu et al. (2016) used a a mutil-task deep learning framework to deal with this problem, they incorporate other similar corpus to deal with the data sparsity problem.

Most of the previous works mentioned above used rich linguistic features and supervised learning methods to achieve the task. In this paper, we propose a deep architecture, which does not need these manually selected features and additional linguistic knowledge base to do it.

5 Conclusion

In this work, we propose to use word embeddings to fight against the data sparsity problem of word pairs. In order to preserve contextual information, we encode a sentence to its positional representation via a recurrent neural network, specifically, a LSTM. To solve the semantic gap between the word pairs, we propose to use a gated relevance network which incorporates both the linear and nonlinear interactions between pairs. Experiment results on PDTB show the proposed model outperforms the existing methods using traditional features on all of the relations.

6 Acknowledgement

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (No. 61532011, 61473092, and 61472088), the National High Technology Research and Development Program of China (No. 2015AA015408).

References

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

Chloé Braud and Pascal Denis. 2015. Comparing word representations for implicit discourse relation classification. In *Empirical Methods in Natural Language Processing (EMNLP 2015)*.

Jifan Chen, Qi Zhang, Pengfei Liu, and Xuanjing Huang. 2016. Discourse relations detection via a mixed generative-discriminative framework. In *AAAI*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050.

Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. 2012. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*, pages 3167–3175.

Yangfeng Ji and Jacob Eisenstein. 2015. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *Transactions of the Association for Computational Linguistics*, 3:329–344.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A pdtb-styled end-to-end discourse parser. *Natural Language Engineering*, 20(02):151–184.

Yang Liu, Sujian Li, Xiaodong Zhang, and Zhifang Sui. 2016. Implicit discourse relation classification via multi-task neural networks. In *AAAI*.

Kathleen McKeown and Or Biran. 2013. Aggregated word pair features for implicit discourse relation disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 69–73. The Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Joonsuk Park and Claire Cardie. 2012. 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*, pages 108–112. Association for Computational Linguistics.

Emily Pitler, Mridhula Raghupathy, Hena Mehta, Ani Nenkova, Alan Lee, and Aravind K Joshi. 2008. Easily identifiable discourse relations. *Technical Reports (CIS)*, page 884.

Emily Pitler, Annie Louis, and Ani Nenkova. 2009. 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: Volume 2-Volume 2*, pages 683–691. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1305–1311.

Attapol Rutherford and Nianwen Xue. 2014. Discovering implicit discourse relations through brown cluster pair representation and coreference patterns. In *EACL*, volume 645, page 2014.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.

Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 149–156. Association for Computational Linguistics.

Ilya Sutskever, Joshua B Tenenbaum, and Ruslan R Salakhutdinov. 2009. Modelling relational data using bayesian clustered tensor factorization. In *Advances in neural information processing systems*, pages 1821–1828.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2015. A deep architecture for semantic matching with multiple positional sentence representations. *arXiv preprint arXiv:1511.08277*.

Bonnie Webber. 2004. D-ltag: Extending lexicalized tag to discourse. *Cognitive Science*, 28(5):751–779.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi PrasadO Christopher Bryant, and Attapol T Rutherford. 2015. The conll-2015 shared task on shallow discourse parsing. In *Proceedings of CoNLL*, page 2.

Rong Zhao and William I Grosky. 2002. Narrowing the semantic gap-improved text-based web document retrieval using visual features. *Multimedia, IEEE Transactions on*, 4(2):189–200.

Zhi-Min Zhou, Yu Xu, Zheng-Yu Niu, Man Lan, Jian Su, and Chew Lim Tan. 2010. Predicting discourse connectives for implicit discourse relation recognition. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1507–1514. Association for Computational Linguistics.