Two End-to-end Shallow Discourse Parsers for English and Chinese in CoNLL-2016 Shared Task

Jianxiang Wang, Man Lan*
Shanghai Key Laboratory of Multidimensional Information Processing
Department of Computer Science and Technology,
East China Normal University, Shanghai 200241, P. R. China
51141201062@ecnu.cn, mlan@cs.ecnu.edu.cn*

Abstract

This paper describes our two discourse parsers (i.e., English discourse parser and Chinese discourse parser) for submission to CoNLL-2016 shared task on Shallow Discourse Parsing. For English discourse parser, we build two separate argument extractors for single sentence (SS) case, and adopt a convolutional neural network for Non-Explicit sense classification based on (Wang and Lan, 2015b)'s work. As for Chinese discourse parser, we build a pipeline system following the annotation procedure of Chinese Discourse Treebank in (Zhou and Xue, 2015). Our English discourse parser achieves better performance than the best system of CoNLL-2015 and the Chinese discourse parser achieves encouraging results. Our two parsers both rank second on the blind datasets.

1 Introduction

A discourse relation between two segments of textual units expresses how they are logically connected to one another (*cause* or *contrast*), which is considered a crucial step for the ability to properly interpret or produce discourse. It can be of great benefit to many downstream natural language processing (NLP) applications and has attracted lots of research (Pitler and Nenkova, 2009; Lin et al., 2009; Pitler et al., 2009; Lan et al., 2013; Rutherford and Xue, 2014; Lin et al., 2014; Kong et al., 2014; Ji and Eisenstein, 2015; Fisher and Simmons, 2015; Braud and Denis, 2015; Zhang et al., 2015).

Following the first edition in CoNLL-2015 (Xue et al., 2015), CoNLL-2016 (Xue et al., 2016) is the 2nd edition of the CoNLL Shared Task on Shallow Discourse Parsing, which contains following tasks: discourse parsing task and supplementary task (sense classification using gold standard argument pairs) in English and Chinese.

To build an English parser, we follow (Wang and Lan, 2015b)'s work except for several modifications described later in section 2. In consideration of distinct linguistic and syntactic difference between English and Chinese, for Chinese parser, we design a new pipeline system which simulates the annotation procedure of Chinese Discourse Treebank in (Zhou and Xue, 2015). And for both English and Chinese sense classification (i.e., supplementary task), we just regard them as parts of the whole parser.

2 English Discourse Parser

2.1 System Overview

The English discourse parser shown in Figure 1 is a pipeline system, which is quite similar with that in (Wang and Lan, 2015b) except for several differences in: (1) build two separate argument extractors for single sentence (SS) case; (2) adopt a convolutional neural network for Non-Explicit Sense Classification; (3) add or remove several features for each component based on hill-climbing strategy.

Therefore, we only describe the differences above-mentioned in details in this paper.

2.2 Separate SS Arguments Extractor

Unlike the work in (Kong et al., 2014; Wang and Lan, 2015b) which built global argument extractor for the SS case in Explicit parser, we build two different argument extractors for

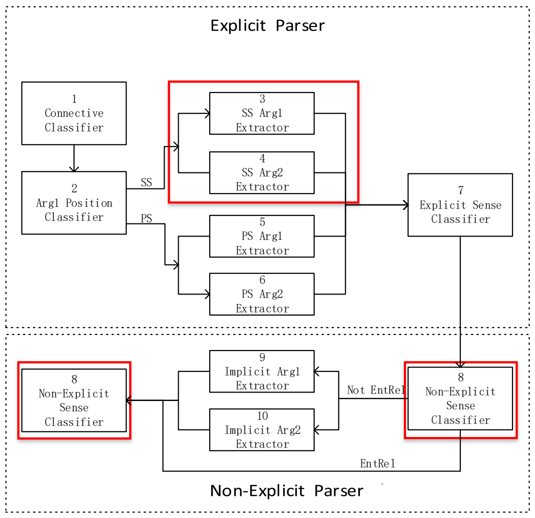
Explicit Parser
1 Connective Classifier
2 Arg1 Position Classifier
SS
PS
3 SS Arg1 Extractor
4 SS Arg2 Extractor
5 PS Arg1 Extractor
6 PS Arg2 Extractor
7 Explicit Sense Classifier
8 Non-Explicit Sense Classifier
9 Implicit Arg1 Extractor
10 Implicit Arg2 Extractor
Not EntRel
8 Non-Explicit Sense Classifier
EntRel
Non-Explicit Parser

Figure 1: System pipeline for the English discourse parser

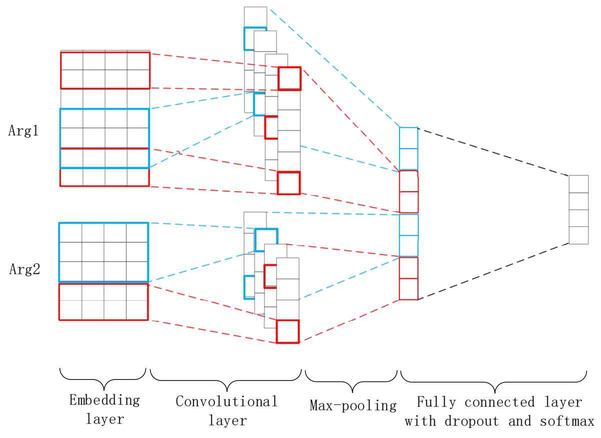

Figure 2: Convolutional neural network for Non-Explicit sense classification

Arg1 and Arg2 separately. Our consideration is that these two arguments have different syntactic and discourse properties and a unified model with the same feature set used for both cases may not have enough discriminating power (Wang and Lan, 2015a).

Specifically, we follow the constituent-based approach in (Kong et al., 2014) which consists of three steps: (1) collecting argument candidates (i.e., constituents) from the parse tree of the sentence containing the connective C; (2) deciding each constituent whether it belongs to Arg1, Arg2 or NULL; (3) merging all the constituents for Arg1 and Arg2 to obtain the Arg1 and Arg2 text spans respectively.

In the second step, however, different from (Kong et al., 2014), we view it as a binary classification. That is, we use two argument extractors to determine each constituent whether it belongs to the argument (Arg1 or Arg2) or not. And in the third step, we merge the constituents for Arg1 and Arg2 from **SS Arg1 Extractor** and **SS Arg2 Extractor** to obtain the Arg1 and Arg2 text spans, respectively.

2.3 Convolutional Neural Network for Non-Explicit Sense Classification

Instead of using lots of handcrafted features, we adopt a convolutional neural network (CNN) to perform Non-Explicit sense classification as shown in Figure 2.

The inputs are two tokenized sentences (i.e., Arg1 and Arg2). For each token, the 300-dimensional word vector representation is obtained from pre-trained word2vec model which was trained on 100 billion words from Google News using the skip-gram architecture (Mikolov et al., 2013) and then we convert each of them into a sentence matrix. We denote the sentence matrix by $\mathbf{A} \in \mathbb{R}^{s \times d}$ (s is the length of sentence and d is the dimensionality of the word vector), and use $\mathbf{A}[i:j]$ to represent the sub-matrix of $\mathbf{A}$ from row i to row j. A convolution operation involves a *filter* $\mathbf{w} \in \mathbb{R}^{h \times d}$ (h is the height of filter, window size of the filter). The output sequence $\mathbf{o} \in \mathbb{R}^{s-h+1}$ of the convolution operator is obtained by repeatedly applying the filter on sub-matrices of $\mathbf{A}$:

$$o_i = \mathbf{w} \cdot \mathbf{A}[i : i + h - 1] \qquad (1)$$

where $i = 1 \ldots s - h + 1$, $\cdot$ is the dot product between the sub-matrix and the filter (a sum over element-wise multiplications). A bias term $b \in \mathbb{R}$ and an activation function f are added to each o_i to compute the *feature map* $\mathbf{c} \in \mathbb{R}^{s-h+1}$ for this filter:

$$c_i = f(o_i + b) \qquad (2)$$

The *max pooling* operation is applied over each feature map to take the maximum value $\hat{c} = max\{\mathbf{c}\}$, and then the outputs generated from each feature map can be concatenated into a fixed-length feature vector (i.e., the representation of the input sentence). We apply convolution and *max pooling* over Arg1 and Arg2 sentence matrix to obtain the Arg1 and

Arg2 representation respectively, then concatenate them to obtain the representation of discourse relation (penultimate layer in CNN) which will be passed to a fully connected softmax layer. Moreover, we apply dropout (Hinton et al., 2012) to penultimate layer. We choose cross-entropy loss as our training objective and use AdaGrad (Duchi et al., 2011) with a learning rate of 0.01 and a minibatch size of 50 to train the model. Table 1 shows the configuration of our model.

Description	Values
input word vectors	Google word2vec
filter window size	(4, 6, 13)
feature maps	100
activation function	Tanh
pooling	max pooling
dropout rate	0.5

Table 1: Configuration of our CNN model

2.4 Feature Engineering

Following (Wang and Lan, 2015b) and (Wang and Lan, 2015a)' work, we tune the features for each component according to hill-climbing strategy. Table 2 lists the specific features for each component in English discourse parser.

Note that for a node in the parse tree, we use the POS combinations of the node, its parent, its right sibling and left sibling to represent the *node context*, and use the POS combinations of the node, its parent, its children to represent the *linked context*. And we use *level distance* to represent the distance between the heights of two nodes in the parse tree. For the Connective Classifier, $prev_1$ and $next_1$ indicate the first previous word and the first next word of connective C. For SS Arg1&2 extractors, we use NT to indicate the constituent. For PS Arg1&2 Extractor and Implicit Arg1&2 Extractor, *prev*, *curr*, *next* refer to previous, current, next clause in the sentence respectively.

For the detail explanation of the features, please refer to (Wang and Lan, 2015b) and (Wang and Lan, 2015a).

3 Chinese Discourse Parser

Due to the distinct differences between English and Chinese language, the discourse annotation procedure of Chinese is quite different from that of English. There are three main differences between English and Chinese parsers.

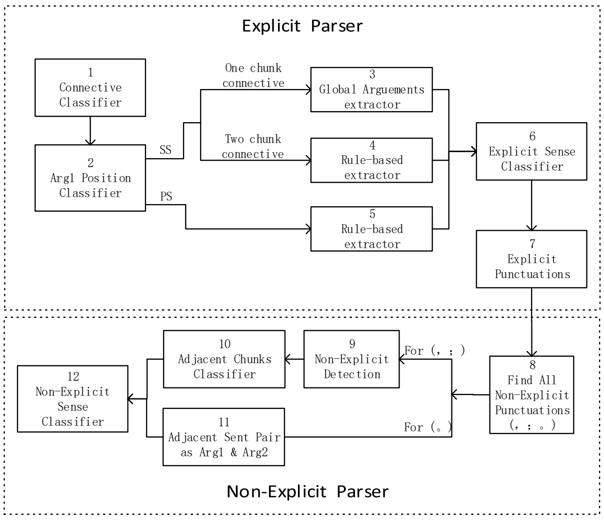
Explicit Parser
1 Connective Classifier
2 Arg1 Position Classifier
SS
PS
One chunk connective
Two chunk connective
3 Global Arguements extractor
4 Rule-based extractor
5 Rule-based extractor
6 Explicit Sense Classifier
7 Explicit Punctuations
8 Find All Non-Explicit Punctuations (，；。)
For (，；)
For (。)
9 Non-Explicit Detection
10 Adjacent Chunks Classifier
11 Adjacent Sent Pair as Arg1 & Arg2
12 Non-Explicit Sense Classifier
Non-Explicit Parser

Figure 3: System pipeline for the Chinese discourse parser

Firstly, there are many punctuations in one Chinese sentence, which indicate discourse relations between text spans, therefor it is important to use them as potential indicators to indicate discourse relations. Secondly, in the case of Explicit connectives, Arg2 in English parser is always the argument to which the connective is syntactically bound, while in Chinese parser the relative positions of Arg1 and Arg2 in Explicit relation are dependent on the relation sense rather than the position of explicit connective. Thirdly, most English discourse connectives are single words while most Chinese discourse connectives come in pairs (e.g., in train set, 62.46% of the connectives are in pairs).

Following the annotation procedure of the Chinese discourse, we design the pipeline system for Chinese discourse parser as shown in Figure 4. Although the Chinese discourse parser is divided into Explicit parser and Non-Explicit parser, which is similar with English discourse parser, most components in Chinese parser perform quite differently as described in following section.

3.1 Explicit Parser

3.1.1 Connective Classifier

First, we use three punctuations (i.e., comma, semi-colon and colon) to split all sentences into *chunks*. Then, we only identify the connectives which span no more than two *chunk*s. That is, we do not consider the connectives ranging

Component	Features
Connective Classifier	lowercased C string, C category, C POS, C + $next_1$, $prev_1$ + C , $prev_1$ POS + C POS, path of C's parent → root, the POS tags of nodes from C's parent → root, *self category*, *right sibling category*, *left sibling category*, *self category* + *right category*, C + *node context* of *right sibling category*, C + *linked context* of *right sibling category*, C + *node context* of *parent category*, C + *linked context* of *parent category*
SS Arg1 Extractor	C *category*, C iLSib, *self category*, *left sibling category*, path of C's parent → root, the *node context* of the *parent category*, *node context* of NT, path of NT → root, NT iRSib, *node context* of NT's parent, path from current NT → next NT, *level distance* between current and previous NT, path of NT → C, C NT position , whether C and NT are in the same clause, whether previous and current NT are in the same clause, whether current and next NT are in the same clause
SS Arg2 Extractor	C *category*, C iLSib, *self category*, *right sibling category*, the POS tags of nodes from C's parent → root, *node context* of NT, *node context* of NT's parent, path of NT → root, path from previous NT → current NT, path from current NT → next NT, path of NT → C, C NT position, *level distance* between C and NT, whether previous and current NT are in the same clause
PS Arg1 Extractor	*curr* first + *next* first, *curr* last + *next* first, *curr* last + *curr* last, path from *curr* first to *prev* last in parse tree
PS Arg2 Extractor	C string, lowercased C string, C category, path of C's parent → root, compressed path of C's parent → root, *next* first, *prev* first, *prev* last + *curr* last, production rule of *curr*, *curr* + the first lemma verb of *curr*.
Explicit Sense Classifier	C + *prev*, C POS, *self category*, *parent category*, *left sibling category*, *right sibling category*, Syn-Syn, C + *left sibling category*, C + *right sibling category*, C + the *node context* of *left sibling category*, *node context* of C, *linked context* of C, previous connective and its POS of as and previous connective and its POS of when.
Implicit Arg1 Extractor	immediately preceding punctuations of *curr*, first lowercased verb in *curr*, *curr* first + first lemma verb in *curr*, *curr* first + *curr* last, path from *curr* first to *prev* last in the parse tree, *prev* first, *prev* first + *curr* first, *prev* last + *curr* last, *prev* last + *curr* first.
Implicit Arg2 Extractor	*curr* first punctuation, *curr* first + first lemma verb in *curr*, *curr* first punctuation + *curr* last punctuation, *prev* first, *prev* last + *curr* first, *prev* first + *curr* first, *prev* last + *curr* last,

Table 2: Features for the components in English discourse parser

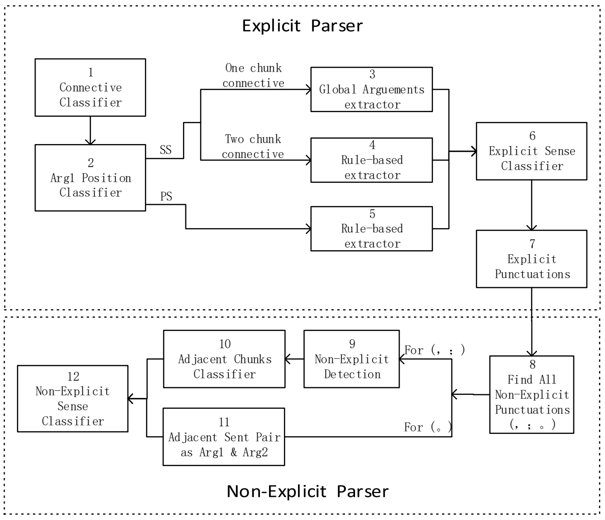
Explicit Parser
1 Connective Classifier
2 Arg1 Position Classifier
SS
PS
One chunk connective
Two chunk connective
3 Global Arguements extractor
4 Rule-based extractor
5 Rule-based extractor
6 Explicit Sense Classifier
7 Explicit Punctuations
8 Find All Non-Explicit Punctuations (, ; 。)
For (, ;)
For (。)
9 Non-Explicit Detection
10 Adjacent Chunks Classifier
11 Adjacent Sent Pair as Arg1 & Arg2
12 Non-Explicit Sense Classifier
Non-Explicit Parser

Figure 4: System pipeline for the Chinese discourse parser

across three or more chunks since their frequency is quite low in our preliminary statistics. Here we refer *one chunk* connective to the connectives ranging across only one chunk, and *two chunk* connective to the connectives cross two chunks. For example, "并" and "在 ··· 情况下" are *one chunk* connectives in Example 1 and 2, whereas "除 ··· 外 ··· 还" is *two chunk* connective in Example 3.

(1) 德波尔大名鼎鼎, [今年入选荷兰队]$_{Arg1}$, 并 [在法国世界杯赛上打入半决赛]$_{Arg2}$ 。

(2) 分析表明, 在 [机遇良多、国际形势十分有利]$_{Arg1}$ 的情况下 , [中国今年经济发展仍面临严竣挑战]$_{Arg2}$ 。

(3) 他们指出, 除 [比索汇率过高]$_{Arg1}$ 外 , [墨出口今年 还 将面临一些新的不利因素]$_{Arg2}$ 。

For each identified connective, we build a connective classifier to decide whether they function as discourse connective or not. The features we used in connective classifier are as follows: C + *node context* of C, $prev_1$ + C, $prev_1$ POS, $prev_1$ POS + C POS, *left sibling category* + *right sibling category*, path of C's parent → root, compressed path of C's parent → root, the POS tags of nodes from C's parent → root. Note that $prev_1$ indicate the first previous word of C. The *self category*, *parent* *category*, *left sibling category*, *right sibling category* features are borrowed from (Pitler and Nenkova, 2009).

3.1.2 Arg1 Position Classifier

The Arg1 Position Classifier is to identify the relative position of Arg1 as whether it is located within the same sentence as the connective (SS) or in some previous sentences of the connective (PS).

The features consist of the following: C string, path of $C \rightarrow$ root, $prev_1$, $prev_1 + C$, $prev_1$ POS + C POS, $prev_2$, $prev_2 + C$, $next_1$, $next_1 + C$, $next_1$ POS, $next_1$ POS + C POS, $next_2 + C$, $next_2$ POS, $next_2$ POS + C POS. Note that $prev_2$ and $next_2$ indicate the second previous word and the second next word of connective C, respectively.

3.1.3 SS Arguments Extractor

Unlike English discourse, even in SS case, most Chinese discourse connectives come in pairs, for example, "在 ··· 的情况下", "虽然 ··· 但是" etc, which makes it hard to label arguments for these connectives.

In the case that a connective is *one chunk* connective, we build a global extractor to label Arg1 and Arg2. In the case of *two chunk* connective, we then adopt a rule-based method to extract Arg1 and Arg2.

***One chunk* connective**: For the *chunk*s in the sentence containing *one chunk* connective, we build a classifier to decide each *chunk* whether it belongs to Arg1, Arg2 or NULL, then merge all the *chunk*s for Arg1 and Arg2 to obtain the Arg1 and Arg2 text spans respectively. Note that if the chunk contains the connective, we remove the connective from this chunk.

The features of this extractor consist of the following: *curr last*, *curr first*, the verbs in *curr*, *next first*, *next last*, the punctuations in the tail of the *curr*, *curr last* + *next last*, *curr last* + *next first*, *prev last* + *curr last*, C ,C + whether C in the *curr*, the relative position of C to *curr* (C before, after or in *curr*), the number of chunks from *curr* to C. Note that *curr* and *next* indicate the current and next following *chunk* respectively, and *first* and *last* mean the first and last word in the *chunk*.

***Two chunk* connective:** If the connectives are *two chunk* connectives, we then use a simple rule-based extractor to extract Arg1 and Arg2. We view the text spans from the first *chunk* to second chunk as Arg1, and the text span in the second *chunk* as Arg2.

3.1.4 PS Arguments Extraction

For the PS cases, we also use a rule-based extractor to extract the arguments. We label the previous sentence of the connective as Arg1, and the text span between the connective and the beginning of the next sentence as Arg2.

3.1.5 Explicit Sense Classifier

To build the Explicit sense classifier, we extract features from the connective C, its context and the parse tree of its sentence, which are listed in the following: C string, C + previous word of C, C + *self category*, C + *left sibling category*, C + *right sibling category*, C + the *node context* of *parent category.*

3.1.6 Explicit Punctuations

According to (Zhou and Xue, 2012), the general annotation procedure for Chinese parser is to scan the text for finding punctuations, and to judge whether there is a discourse relation when a punctuation is encountered. If yes, annotators then characterize the relation and if not, they keep on scanning, whereas they identify Explicit connective firstly. Inspired by this annotation, we present this component to obtain the Explicit punctuation for each Explicit relation according to the connective and its two arguments using the following strategy:

(1) If the two arguments are not embedded into each other, we use the punctuation between two arguments as Explicit punctuation, and if more than one punctuation between two arguments, choose the closest one to Arg1.

(2) If the two arguments are embedded into each other, choose the closest punctuation following Arg1.

3.2 Non-Explicit Parser

3.2.1 Find All Non-Explicit Punctuations

After the above-mentioned Explicit punctuations procedure, we denote all remaining (i.e., not identified as Explicit punctuations) three punctuations (i.e., 。，；) from the texts as Non-Explicit candidate punctuations.

3.2.2 For ， ； punctuations

However, not each of the Non-Explicit candidate punctuations contains a Non-Explicit relation, for example, the first punctuation (，) in Example 2. Therefore, we use Non-Explicit Detection component to judge each of ， ； punctuations whether or not, and if there is, an Adjacent Chunk Classifier is adopted to obtain its Arg1 and Arg2.

Non-Explicit Detection: The features for this component contain the following: *prev last*, first verb of *prev*, length of *prev*, *prev last* + *prev first*, unigram of *prev*, the punctuation token. *prev* denotes the previous *chunk* of the punctuation.

Adjacent Chunk Classifier: If there is a Non-Explicit relation in ， or ； punctuation, we use an Adjacent Chunk Classifier to judge whether the previous one chunk or two chunks of the punctuation is labelled as Arg1 and whether the next one chunk or two chunks of the punctuation is labelled as Arg2.

We use the following features to build the classifier: unigram of *first*, length of *first*, first word of *first*, last word of *first*, first word of *first* + last word of *first*, unigram of *second*, length of *second*, first word of *second*, last word of *second*, first word of *second* + last word of *second*, the connectives in two chunks. Note that *first* and *second* refer to the first and second *chunk*, respectively.

3.2.3 For 。 punctuation

For all 。punctuations, we assume each of them contains a Non-Explicit relation, and then extract Arg1 and Arg2 by labeling the previous sentence of the connective as Arg1, and the text span between the connective and the beginning of the next sentence as Arg2.

3.2.4 Non-Explicit Sense Classifier

From the previous components, we have obtained the two arguments of the Non-Explicit relations. To perform the Non-Explicit sense classification, we extract features from the arguments pair: *production rules*, word pairs in the first *chunk* of each argument, verb pairs in the argument pair, first verb pair in the argument pair, Arg1 *last*, Arg1 *first3*, Arg1 *first*, Arg1 *first* + Arg2 *last*, Arg1 *last* + Arg2 *last*, Arg2 *first3*.

4 Experiments

All classifiers in the two parsers are trained using logistic regression with the default parameters (i.e., c=1) implemented in LIBLINEAR toolkit [1]. We adopt the same Explicit Sense Classifier and Non-Explicit Sense Classifier used in the discourse parser for both English and Chinese supplementary tasks which are sense classification using gold standard argument pairs.

From Table 3, compared with the best system in CoNLL-2015 (Wang and Lan, 2015b) on blind dataset, our system achieves better performances on Explicit arguments extraction and Non-Explicit arguments extraction and beat them on the overall performance. From table 4, we see that the performance of Explicit sense classification is better on dev and blind test set, which is slight lower on the test set than the performance of (Wang and Lan, 2015b). As for the Non-Explicit sense classification in supplementary task, we achieve much better performance than (Wang and Lan, 2015b) on dev and test set when using CNN instead of handcrafted features. However, our CNN model achieve a worse performance on blind test set, the possible reason might be that the blind test set has a different sense distribution compared with dev and test sets. Note that the dev and test set are both from PDTB dataset, whereas the blind test set is annotated from English Wikinews [2].

For Chinese discourse parser, from table 3, we see the performance of the Explicit connective identification on Chinese is much lower than that in English and reduced a lot from dev to test and blind test, the possible reason might be that there are lots of connectives come in pairs and much more unseen connectives in the Chinese test than in English which makes it hard to detect and classify them from the texts. From Table 4, the performance of Non-Explicit sense classification in Chinese is much higher than in English, due to the high performance of the baseline system (labelling the sense of all the Non-Explicit relations as "Conjunction" can achieve the 64.61% accuracy on train set). Due to the variety distri-

[1]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/multicore-liblinear/

[2]https://en.wikinews.org/

	English				Chinese		
	dev	test	blind	(Wang and Lan, 2015b)'s blind	dev	test	blind
Explicit connective	95.22	93.96	91.34	91.86	86.27	72.41	63.07
Explicit Arg1 extraction	62.01	51.39	**51.05**	48.31	67.97	59.77	41.13
Explicit Arg2 extraction	81.26	76.43	74.20	74.29	70.59	62.07	47.53
Explicit Both extraction	55.11	44.31	**42.84**	41.35	56.21	47.13	31.81
Non-Explicit Arg1 extraction	68.84	64.66	**61.05**	60.87	59.86	59.55	54.21
Non-Explicit Arg2 extraction	73.81	66.86	**75.83**	74.58	65.25	65.26	54.99
Non-Explicit Both extraction	58.39	50.83	**51.15**	50.41	50.50	50.12	42.10
All Arg1 extraction	66.39	59.18	**57.22**	55.84	63.17	61.63	56.19
All Arg2 extraction	77.32	71.38	**75.10**	74.45	67.60	67.35	57.20
All Both extraction	56.85	47.79	**47.43**	46.37	52.45	50.82	41.99
Overall parser	40.43	30.70	**25.99**	24.00	42.42	40.25	26.60

Table 3: Results of our English and Chinese discourse parsers on dev, test and blind test datasets

	English			Chinese		
	dev	test	blind	dev	test	blind
Explicit	92.56 (90.00)	90.13 (90.79)	77.41 (76.44)	96.10	94.24	76.69
Non-Explicit	**46.51** (42.72)	**40.91** (34.45)	34.20 (36.29)	73.53	72.42	60.52
ALL	**67.97** (65.11)	**64.34** (61.27)	54.06 (54.76)	78.07	77.01	64.73

Table 4: Results of the supplementary tasks on both English and Chinese discourses, which are sense classification using gold standard argument pairs. The corresponding performance of (Wang and Lan, 2015b)'s system is shown within parentheses.

bution of the arguments, the arguments extraction is more challenging than other components, and achieve low performance on test set.

5 Conclusion

In this work, we improve the English discourse parser on previous best system (Wang and Lan, 2015b) in three aspects: (1) build two separate argument extractors for the SS case; (2) adopt convolutional neural network to do Non-Explicit Sense Classification; (3) add or remove some features for each component based on the hill-climbing strategy. And we build a Chinese discourse parser following the annotation procedure of Chinese Discourse Treebank. Our English discourse parser achieves a better performance than the best system in CoNLL-2015, and we have obtained encouraging results of the Chinese discourse parser.

Acknowledgements

This research is supported by grants from Science and Technology Commission of Shanghai Municipality under research grant no. (14DZ2260800 and 15ZR1410700) and Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (ZF1213).

References

Chloé Braud and Pascal Denis. 2015. Comparing word representations for implicit discourse relation classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2201–2211, Lisbon, Portugal, September. Association for Computational Linguistics.

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.

Robert Fisher and Reid Simmons. 2015. Spectral semi-supervised discourse relation classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 89–93, Beijing, China, July. Association for Computational Linguistics.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

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.

Fang Kong, Hwee Tou Ng, and Guodong Zhou. 2014. A constituent-based approach to argument labeling with joint inference in discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 68–77, Doha, Qatar, October. Association for Computational Linguistics.

Man Lan, Yu Xu, Zheng-Yu Niu, et al. 2013. Leveraging synthetic discourse data via multi-task learning for implicit discourse relation recognition. In *ACL (1)*, pages 476–485. Citeseer.

Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the Penn Discourse Treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 343–351. Association for Computational Linguistics.

Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2014. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering*, pages 1–34.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Emily Pitler and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 13–16. Association for Computational Linguistics.

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.

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

Jianxiang Wang and Man Lan. 2015a. Building a high performance end-to-end explicit discourse parser for practical application. In *Knowledge Science, Engineering and Management*, pages 324–335. Springer.

Jianxiang Wang and Man Lan. 2015b. A refined end-to-end discourse parser. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning - Shared Task*, pages 17–24, Beijing, China, July. Association for Computational Linguistics.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi Prasad, Christopher Bryant, and Attapol Rutherford. 2015. The CoNLL-2015 Shared Task on Shallow Discourse Parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning: Shared Task*, Beijing, China.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Bonnie Webber, Attapol Rutherford, Chuan Wang, and Hongmin Wang. 2016. The conll-2016 shared task on multilingual shallow discourse parsing. In *Proceedings of the Twentieth Conference on Computational Natural Language Learning - Shared Task*, Berlin, Germany, August. Association for Computational Linguistics.

Biao Zhang, Jinsong Su, Deyi Xiong, Yaojie Lu, Hong Duan, and Junfeng Yao. 2015. Shallow convolutional neural network for implicit discourse relation recognition. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2230–2235, Lisbon, Portugal, September. Association for Computational Linguistics.

Yuping Zhou and Nianwen Xue. 2012. Pdtb-style discourse annotation of chinese text. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 69–77. Association for Computational Linguistics.

Yuping Zhou and Nianwen Xue. 2015. The chinese discourse treebank: A chinese corpus annotated with discourse relations. *Language Resources and Evaluation*, 49(2):397–431.