Argument Labeling of Explicit Discourse Relations using LSTM Neural Networks

Sohail Hooda **Leila Kosseim**
Depart. of Computer Science and Software Engineering
Concordia University
1515 Ste-Catherine Street West
Montréal, Québec, Canada H3G 2W1
{s_hooda, kosseim}@encs.concordia.ca

Abstract

Argument labeling of explicit discourse relations is a challenging task. The state of the art systems achieve slightly above 55% F-measure but require hand-crafted features. In this paper, we propose a Long Short Term Memory (LSTM) based model for argument labeling. We experimented with multiple configurations of our model. Using the PDTB dataset, our best model achieved an F1 measure of 23.05% without any feature engineering. This is significantly higher than the 20.52% achieved by the state of the art RNN approach, but significantly lower than the feature based state of the art systems. On the other hand, because our approach learns only from the raw dataset, it is more widely applicable to multiple textual genres and languages.

1 Introduction

In well written texts, discourse relations are used to provide additional meaning to the underlying content by connecting two textual segments logically. This in turn facilitates the reader's understanding of the text. For example, in:

(1) *We would stop index arbitrage* <u>when</u> **the market is under stress.** [1]

two discourse segments, or arguments (`Arg1` in italics and `Arg2` in bold), are explicitly connected via the <u>connective</u> underlined and related by the discourse relation of CONDITION. Discourse relations can be made explicit through the use of discourse connectives such as *although*, *but*, *since*, *because*, etc. or can be left implicit, when no explicit cue phrase is used to signal the relation.

[1] This example is taken from the Penn Discourse Treebank (Prasad et al., 2007).

Discourse parsing involves two main tasks: (1) argument labeling, or identifying the boundaries and labeling `Arg1` and `Arg2`, and (2) relation labeling, or identifying the discourse relation that holds between the arguments. Because discourse parsing allows a deeper understanding of the communicative goal of text segments, it has been used in a variety of downstream NLP applications such as text summarization (Barzilay and Lee, 2004; Yoshida et al., 2014) and question-answering (Chai and Jin, 2004; Verberne et al., 2007).

As witnessed in the recent CoNLL shared tasks (Xue et al., 2015, 2016), full end-to-end discourse parsing is still a challenge. In particular, argument labeling is difficult as the exact boundaries of both `Arg1` and `Arg2` must be identified. The state-of-the-art system (Wang and Lan, 2016) achieves an F-measure of only 55.11% and thus leaves much room for improvement. Most work in this domain make use of a variety of hand-crafted features that do not handle long-distance dependencies well. However, the great majority of discourse arguments are not adjacent to one another and long distance features are important for this task.

In this paper, we investigate the use of recurrent neural networks for discourse labeling. Specifically, we investigate the use of Long Short Term Memory (LSTMs) to better handle long term dependencies and automatically extract and embed the features without prior input. We show that a widely applicable model can be produced by learning from the input data alone, and learning the features directly. This allows more generalized applications of our model across multiple text genres as well as languages. We also show that the approach does not suffer from long distance dependencies and achieves stable results regardless of the distance between `Arg1` and `Arg2`. To our knowledge, this is the first attempt at argument labelling using Deep Learning architectures that uses no hand-crafted features and achieves good results in contrast to the existing systems which rely on hand-crafted features during the learning process of the model.

2 Related Work

Due to the CoNLL 2015 and 2016 shared tasks (Xue et al., 2015, 2016), much recent work has addressed the problem of discourse parsing. However, much work is still needed in order to reach the human performance of 82.8% reported by (Miltsakaki et al., 2004). Discourse parsing consists of both: (1) argument labeling and (2) relation labeling (e.g. (Lin et al., 2014; Laali et al., 2015; Kong et al., 2014; Xue et al., 2015, 2016)). The approaches used for argument labeling at CoNLL 2015 (e.g. (Laali et al., 2015; Son et al., 2015; Zhou, 2015) largely consisted of standard supervised machine learning techniques such as Support Vector Machines (SVMs), Conditional Random Fields, Naive Bayes and Maximum Entropy (MaxEnt) and viewed the problem as a sequence labeling task. These models used syntactic positional and lexical features (such as the first word after the discourse connective) to learn to identify discourse arguments. These traditional methods achieved F-scores in the order of 45-55%. In 2016, several Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) were introduced for relation labeling (Qin et al., 2016). However, for argument labeling, only (Wang et al., 2015) used RNNs with word embeddings as input which were learned specifically from the training corpus and combined these with hand engineered features based on part of speech tags as well as other linguistic information. The group also used a classifier to determine whether a discourse relation spanned over more than one sentence and based on its output used separate classifiers for same-sentence and multiple-sentence relations. However their F1 score of 20.52% was still significantly below the F1 scores of traditional methods. While applying hand engineered features in a machine learning model does provide good results, it however, forces the model to be specific to the dataset that it is applied to. As a result, the model is unable to perform well with other datasets where either the content of the dataset differs or the language (Prasad et al., 2011). In contrast, if a machine learning model is configured to learn features solely on the training dataset, it would be able to generalize over larger datasets of different domain with relative ease.

3 Motivation

A major problem in argument labeling is that arguments may be dependent on long distance features. For example, in:

(2) *These are all market excesses* (putting aside the artificial boosts that the tax code gives to debt over equity), <u>and</u> **what we've seen is the market reining them in**.

`Arg1` and `Arg2` are separated by 14 words[2]. These words, known as *attribution*, are a challenge for standard features. For example, features from (Wang et al., 2015) such as "1st Previous Word of Connective", "2nd Previous Word of Connective", "1st Previous POS of Connective " or "2nd Previous POS of Connective", do not handle non-adjacent arguments well. Table 1 shows the number of instances in the PDTB dataset (Prasad et al., 2007) used at CoNLL 2015 and 2016 that do not have consecutive arguments. As Table 1 shows, 78% of `Arg1` and `Arg2` are not located consecutively and 24% are separated by at least 5 words. Standard machine learning techniques have much difficulty to account for these and a standard RNN approach (as used by (Wang et al., 2015)) also suffers from long distance dependencies between `Arg1` and `Arg2` due to the problem of vanishing/exploding gradients (Hochreiter et al., 2001). When attempting to incorporate information between words in arguments that are non-consecutive, as the distance between argument increases, the information collected either converges to zero or to infinity resulting in over- or under-estimating the learning of the neural network. On the other hand, LSTMs can control this behaviour and have been shown to model long distance dependencies much better (Hochreiter and Schmidhuber, 1997). For this reason, we experimented with such an approach.

4 Experiment

4.1 Corpus

To train and validate our LSTM approach, we used the Penn Discourse Treebank Corpus (PDTB) (Prasad et al., 2007). The PDTB has become

[2]The connective acts as a marker for the start of `Arg2`, and therefore it is not included in this calculation.

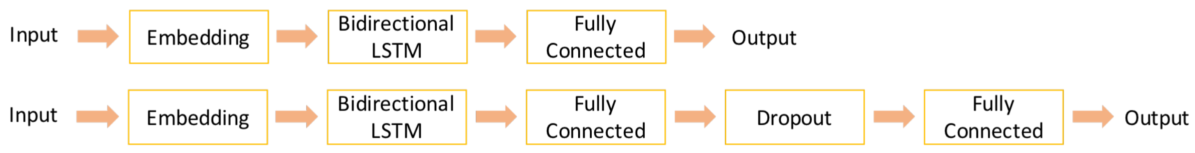

Figure 1: Architecture of model `m1` (top) and model `m2` (bottom)

Table 1: Statistics on the distance (in number of word tokens) between `Arg1` and `Arg2` in the explicit relations in the training and test set of the PDTB Dataset (Prasad et al., 2008)

Distance	**Number of instances**	**Percentage**
0	3,554	22.29%
1	8,582	53.82%
2-10	1,743	10.93%
>10	2,066	12.96%
Total	**15,945**	**100.00%**

Table 2: Number of instances in the PDTB Dataset

Dataset	**Explicit**	**Non-Explicit**	**Total**
Training	15,246	17,289	32,535
Testing	699	737	1,436
Total	**15,945**	**18,026**	**33,971**

the standard dataset in discourse parsing, thanks, in part, to the CoNLL shared tasks (Xue et al., 2015, 2016). Using this corpus allowed us to compare our work with the state of the art systems. The PDTB contains both explicit relations (marked with discourse connectives such as *because* or *but*) as well as non-explicit relations. Table 2 shows statistics of the dataset.

Since we focused on explicit relations only, the dataset was first cleaned by removing all non-explicit relations. As is standard in the field, we used sections 2-21 of the PDTB for training and section 22 was used for testing. Thus, about 15,246 instances such as example (1) in Section 1 were used for the training process and 699 were used for testing.

4.2 Network Architecture

For our experiments, we used a neural network composed of Long Short-Term Memory (LSTM) cells (Hochreiter and Schmidhuber, 1997). An LSTM is a specialized form of a Recurrent Neural Network (RNN) where a neuron is replaced with a memory cell. The memory cell is able to learn and hold information to take into account long term dependencies, thus allowing to overcome the problem of vanishing and exploding gradients with RNNs (Hochreiter et al., 2001).

In order to learn the position of `Arg1` and `Arg2` and the length of these segments from the data only, we experimented with two main architectures. The first architecture shown in Figure 1 (top) is composed of an embedding layer that feeds directly into a Bidirectional LSTM layer. The Bidirectional LSTM layer is composed of 100 LSTM cells (for each direction) and the initialization is performed via the Glorot Uniform technique (Glorot and Bengio, 2010). The Bidirectional LSTM outputs are then fed into a fully connected layer which outputs the probability of 4 possible labels: `Arg1`, `Arg2`, `connective` or `none` for each input word. In the second architecture, shown in Figure 1 (bottom), we added a dropout layer as well as a fully connected layer at the end of the first model. This is shown in Figure 1. We tested our architectures with different dimensions of word embeddings and decided to use a value of 300 as it resulted in a higher accuracy with the training set (see Section 4.3). Thus, this created the 2 models below:

1. `m1`: Bidirectional LSTM layer + vectors of 300 words

2. `m2`: Bidirectional LSTM layers + Dense + Dropout + Dense + vectors of 300 words

Using both models, we also experimented with randomly generated embeddings as well as precomputed word embeddings from (Pennington et al., 2014) (see Section 4.3). All models were learned over 50 epochs. The cost function was minimized via the Adam Optimizer (Kingma and Ba, 2015), a memory efficient stochastic optimizer that relies on first order differentials only and calculates updates via the first and second order moments of the gradients thereby resulting in a linear update process. The cost was minimized over the

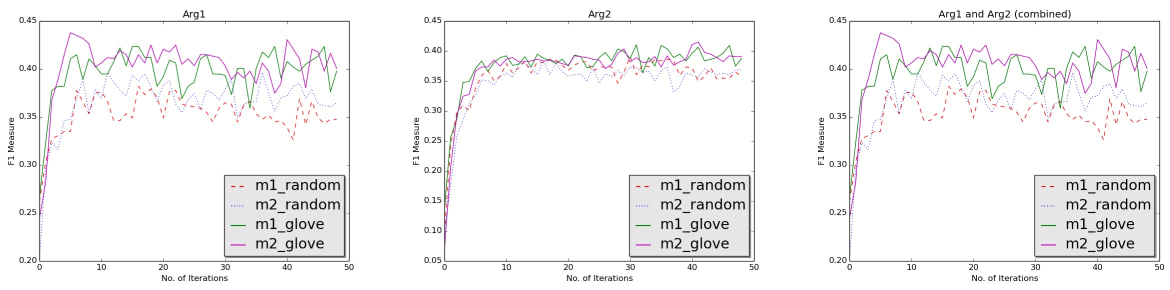

Figure 2: F1 score on the test set as a function of the number of iterations on the training set for `Arg1` (top), `Arg2` (middle) and `Arg1+Arg2` (bottom)

We would have to wait until **we have collected on those assets**
[1, 2, 3, 4, 5, 6, 1, 7, 8, 9, 10, 11, 0, 0,..., 0]

Figure 3: Example with words in a training instance labelled with the corresponding numeric value

mean of the labels for an entire mini batch provided in a single iteration.

4.3 Data Preparation

To provide as input to the neural networks, the training instances were converted into a numeric matrix structure. Since word embeddings are updated dynamically, we used pre-computed `GloVe` embeddings (Global Vectors for Word Representation) (Pennington et al., 2014) in one set of experiments and random values in another set. This was done by creating a dictionary of words and assigning each word to a random numeric value. As shown in Figure 3, a "zero word" was added to the vocabulary as a placeholder to pad sentences to an equal length. This length was set to 1,170 words, which is the size of the longest discourse segment containing both `Arg1` and `Arg2` segments in the PDTB training dataset. Thus the input data was a 2 dimensional matrix of a fixed size of 300 by 1,170. The use of fixed size vectors was not a necessary requirement for the network, but it was mechanically easier to have consistency within the dataset. The label vectors were also correspondingly padded with the `none` class. This allowed the network to learn the end of the `Arg1+Arg2` sequence.

5 Results and Analysis

To evaluate our approach, we used the official CoNLL scoring module[3] and modified it to calculate the performance for explicit relations only. Specifically, the scoring module provides the scores for the exact match for `Arg1` only, `Arg2` only and `Arg1+Arg2`, for every instance in the test set.

For both models, the performance was evaluated at every epoch for a total of 50 evaluation points. Figure 2 shows the F1 scores of the models for `Arg1` only, `Arg2` only and `Arg1+Arg2`. As the graphs show, after about 10 epochs all models seem to stabilize and learn at a much slower rate hence reaching a saturation point.

Table 3 shows the performance of our approaches compared to the state of the art systems. As the table shows, hand-engineered approaches still out perform our LSTM methods with F-measures between 55% to 46% for both `Arg1+Arg2`. However, compared to (Wang et al., 2015), both `m1` and `m2` outperform their RNN approach which did incorporate some hand-engineered features. It is also worthwhile to note that pre-computed embeddings result in slightly higher F1 measures for `Arg1+Arg2` than the random embeddings (25.75% versus 23.75% for `m2` and 24.89% versus 22.75% for `m2`) . This is most likely because of the sparsity of the words used

[3] available at https://github.com/attapol/conll16st

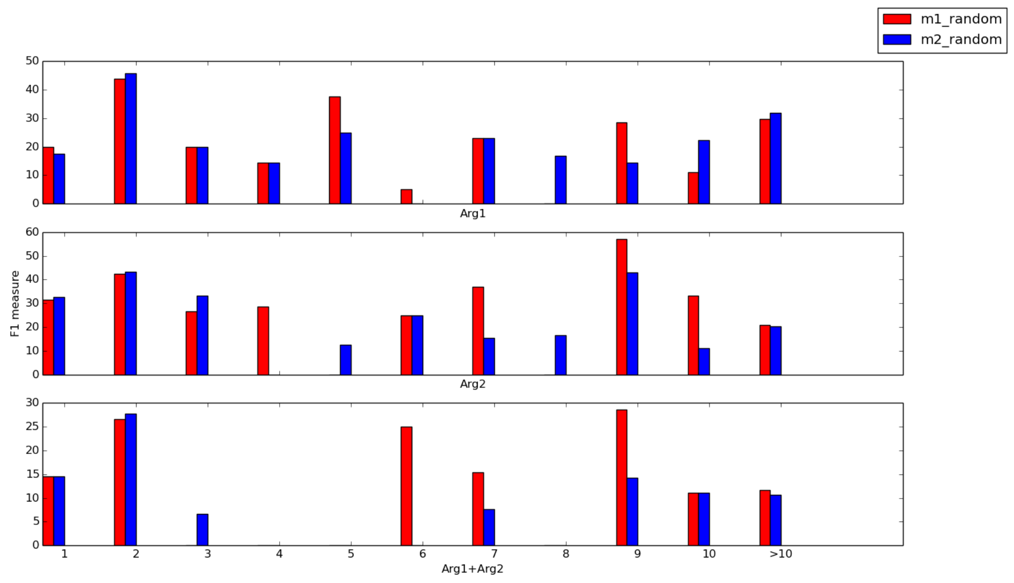

Figure 4: Plot of the distance-based F1 scores for Arg1 (top), Arg2 (middle) and Arg1+Arg2 (bottom)

in the dataset. Since not all words are equally weighted, the system is unable to learn the relationship of those words in a given argument directly from the dataset. Therefore, having pre-computed embeddings assist in optimizing the learning process for those words.

Because `Arg2` is structurally bound to the connective, in the case of explicit relations, identifying the connective gives strong evidence to locate `Arg2`. On the other hand, `Arg1` is much harder to identify as it can be located in various positions relative to `Arg2`. In the case of our LSTM based approach, it is interesting to note that while the F1 scores of `Arg1` and `Arg2` independently are quite lower than the state of the art, this difference diminishes drastically for the combined `Arg1+Arg2` F1 scores. This is because the neural network optimizes over an entire instance and hence tries to maximize the score for both arguments combined as opposed to independently optimizing `Arg1` and `Arg2` labeling.

To verify how our LSTM based approach handled long term dependencies, we separated the test dataset by distance and computed a distance-based F1 score. Recall from Section 2 that the distance is measured by the number of words between the closest words of `Arg1` and `Arg2` excluding the connective. Thus we count from the end of `Arg1` to the start of `Arg2` or the connective whichever comes first when `Arg1` precedes `Arg2` and from the end of `Arg2` or the connective whichever comes last to the start of `Arg1` when `Arg2` precedes `Arg1`. Figure 4 shows the F1 scores for both the models learned with randomly initialized embeddings, calculated at their last epoch, as a function of the distance between `Arg1` and `Arg2`. It is interesting to note that while model `m1_random` seems to perform better on the longer distance based relations (greater than 9), model `m2_random` still gets a better `Arg1+Arg2` accuracy score. Moreover, both models do not show any correlation in their F1 measure as the distance increases. This indicates that both models are unaffected by long and short distances between the arguments of a discourse relation.

6 Conclusion and Future Work

This paper has presented a novel approach for argument labeling based on LSTMs. To our knowledge, this is the first attempt at using Deep Learning for this task that achieves good results without any features in comparison to the existing systems which rely entirely or partially on hand-crafted features. The approach adds value to this domain by decoupling the feature specificity of the PDTB dataset with the problem at hand. Thus, by using LSTM networks, it is possible to generalize the accuracy over different dataset. However, further research is required to prove this hypothesis ¿¿. We experimented with two configurations of our model and showed that using the PDTB training set, our best model achieved 23.05% F1 mea-

Table 3: F1 scores of our LSTM models for explicit relations compared to the best (hand-crafted) approaches and to (Wang et al., 2015)

Model	`Arg1+Arg2`	`Arg1`	`Arg2`	Method
(Wang and Lan, 2016)	55.11%	62.01%	81.26%	Linear classification
(Schenk et al., 2016)	54.41%	61.97%	78.87%	CRF
(Qin et al., 2016)	53.44%	60.99%	79.94%	SVM
(Oepen et al., 2016)	51.37%	60.72%	75.83%	SVM
(Kong et al., 2016)	46.37%	52.89%	74.81%	MaxEnt
`m2_GloVe`	25.75%	42.06%	41.49%	Bidirectional LSTM
`m1_GloVe`	24.89%	42.35%	39.48%	Bidirectional LSTM
`m2_random`	23.75%	39.63%	37.34%	Bidirectional LSTM
`m1_random`	22.75%	36.62%	38.63%	Bidirectional LSTM
(Wang et al., 2015)	20.52%	28.55%	41.78%	RNN

sure without feature engineering. We have shown that our LSTM-based models deal well with the long term dependencies of explicit discourse relations, an important problem with standard machine learning techniques.

A number of improvements can be suggested over this approach. As shown in Table 3, at this point feature-engineered approaches still provide a better performance than our LSTM-based method especially for labeling `Arg2`. To address this, it would be interesting to explore the use of a cascading network where a first network identifies the relative location of `Arg1` with respect to `Arg2` then forwards the discourse to a specialized network capable of learning only that specific type of location. In order to see how much weight is assigned to the discourse connective by the model, one could also test this approach on the implicit relations of the PDTB dataset. Finally, applying this approach on the Chinese Discourse Tree Dataset (Zhou and Xue, 2015) would also be helpful in providing stronger evidence that the features that are learned are independent of the context and language.

References

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* pages 113–120.

Joyce Y Chai and Rong Jin. 2004. Discourse structure for context question answering. In *Proceedings of the Workshop on Pragmatics of Question Answering at HLT-NAACL*. pages 23–30.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. pages 249–256.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. *A field guide to dynamical recurrent neural networks* pages 237–244.

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

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR-2015*. San Diego, CA, USA.

Fang Kong, Sheng Li, Junhui Li, Muhua Zhu, and Guodong Zhou. 2016. SoNLP-DP System for ConLL-2016 English Shallow Discourse Parsing. In (Xue et al., 2016), pages 65–69.

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 EMNLP-2014*. Doha, Qatar, pages 68–77.

Majid Laali, Elnaz Davoodi, and Leila Kosseim. 2015. The CLaC Discourse Parser at CoNLL-2015. In (Xue et al., 2015), pages 56–60.

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.

Eleni Miltsakaki, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2004. Annotating discourse connectives and their arguments. In *Proceedings of the HLT/NAACL Workshop on Frontiers in Corpus Annotation*. Boston, MA, USA, pages 9–16.

Stephan Oepen, Jonathon Read, Tatjana Scheffler, Uladzimir Sidarenka, Manfed Stede, Erik Velldal, and Lilja Øvrelid. 2016. OPT: OsloPotsdamTeesside Pipelining Rules, Rankers, and Classifier Ensembles for Shallow Discourse Parsing. In (Xue et al., 2016), pages 20–26.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of LREC-2008*. Marrakech, Morocco.

Rashmi Prasad, Susan McRoy, Nadya Frid, Aravind Joshi, and Hong Yu. 2011. The biomedical discourse relation bank. *BMC bioinformatics* 12(1):188.

Rashmi Prasad, Eleni Miltsakaki, Nikhil Dinesh, Alan Lee, Aravind Joshi, Livio Robaldo, and Bonnie L Webber. 2007. The Penn Discourse Treebank 2.0 Annotation manual. https://www.seas.upenn.edu/ pdtb/PDTBAPI/pdtb-annotation-manual.pdf.

Lianhui Qin, Zhisong Zhang, and Hai Zhao. 2016. Shallow discourse parsing using convolutional neural network. In (Xue et al., 2016), pages 70–77.

Niko Schenk, Christian Chiarcos, Kathrin Donandt, Samuel Rönnqvist, Evgeny A Stepanov, and Giuseppe Riccardi. 2016. Do We Really Need All Those Rich Linguistic Features? A Neural Network-Based Approach to Implicit Sense Labeling. In (Xue et al., 2016), pages 41–49.

Nguyen Truong Son, Ho Bao Quoc, and Nguyen Le Minh. 2015. Jaist: A two-phase machine learning approach for identifying discourse relations in newswire texts. In (Xue et al., 2015), pages 66–70.

Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2007. Evaluating discourse-based answer extraction for why-question answering. In *Proceedings of the ACM SIGIR*. Amsterdam, Netherlands, pages 735–736.

Jianxiang Wang and Man Lan. 2016. Two End-to-End Shallow Discourse Parsers for English and Chinese in CoNLL-2016 Shared Task. In (Xue et al., 2016), pages 33–40.

Longyue Wang, Chris Hokamp, Tsuyoshi Okita, Xiaojun Zhang, and Qun Liu. 2015. The DCU discourse parser for connective, argument identification and explicit sense classification. In (Xue et al., 2015), pages 89–94.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Rashmi Prasad, Christopher Bryant, and Attapol Rutherford, editors. 2015. *The CoNLL-2015 shared task on shallow discourse parsing*. Beijing, China.

Nianwen Xue, Hwee Tou Ng, Sameer Pradhan, Attapol Rutherford, Bonnie Webber, Chuan Wang, and Hongmin Wang, editors. 2016. *The CoNLL-2016 Shared Task on Shallow Discourse Parsing*. Berlin, Germany.

Yasuhisa Yoshida, Jun Suzuki, Tsutomu Hirao, and Masaaki Nagata. 2014. Dependency-based discourse parser for single-document summarization. In *EMNLP*. Doha, Qatar, pages 1834–1839.

Fang Kong Sheng Li Guodong Zhou. 2015. The SoNLP-DP system in the CoNLL-2015 shared task. In (Xue et al., 2015), pages 32–36.

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.