The DCU Discourse Parser: A Sense Classification Task

Tsuyoshi Okita, Longyue Wang, Qun Liu
Dublin City University, ADAPT Centre
Glasnevin, Dublin 9, Ireland
{tokita,lwang,qliu}@computing.dcu.ie

Abstract

This paper describes the discourse parsing system developed at Dublin City University for participation in the CoNLL 2015 shared task. We participated in two tasks: a connective and argument identification task and a sense classification task. This paper focuses on the latter task and especially the sense classification for implicit connectives.

1 Introduction

This paper describes the discourse parsing system developed at Dublin City University for participation in the CoNLL 2015 shared task (Xue et al., 2015). We participated in two tasks: a connective and argument identification task and a sense classification task. This paper focuses on the latter task.

We divide the whole process into two stages: the first stage concerns an identification of triples $(Arg1, Conn, Arg2)$ and pairs $(Arg1, Arg2)$ while the second stage concerns a sense classification of the identified individual triples and pairs. The first phase of the identification of connective and arguments are described in (Wang et al., 2015), which bases on the framework of (Lin et al., 2009) and is also presented in this shared task as a different paper. Hence, we omit the detailed description of the first stage (See (Wang et al., 2015) for identification of connectives and arguments). This paper focuses on the second stage which concerns sense classification.

2 Sense Classification

We use off-the-shelf classifiers with four kinds of features: relational phrase embedding, production, word-pair and heuristic features. Among them, we test the method which incorporates relational phrase embedding features for $Arg1$ and $Arg2$ for discourse parsing. Production features are proposed in (Lin et al., 2014) and word-pair features are reported in (Lin et al., 2014; Rutherford and Xue, 2015). Heuristic features, which is specific for explicit sense classification, are described in (Lin et al., 2014).

	Rel phrase (2.1)	Prod (2.2)	Word pair (2.3)	Heuristic feat (2.4)
Implicit	yes	yes/no[1]	yes/no[2]	no
Explicit	yes	no	no	yes

Table 1: Overview of features used for implicit/explicit classification.

We consider the embedding models which lead to two different types of intermediate representations. The relational phrase embedding model considers the dependency within words uniformly without considering the second-order effect. The word-pair embedding model considers the second-order effect of specific combinations within the word-pairs in $Arg1$ and $Arg2$. If we plug in a paragraph vector model for the relational phrase embedding model, the model considers the effect of uni-gram within a sentence as a sequence. If we plug in a RNN-LSTM model (Le and Zuidema, 2015), the model considers the effect of uni-gram within a sentence as a tree.

2.1 Relational Phrase Embedding Features

Phrase embeddings (or sentence embeddings) are distributed representation in a higher level than a word level. We used a paragraph vector model to obtain these phrase embeddings (Le and Mikolov, 2014). Upon obtained the phrase embeddings for

[2] For the official score, we did not use production features due to the timing constraint. We write the result for the development set.

[3] For the official score, we did not use the word-pair feature due to the timing constraint. We write the result for the development set.

Proceedings of the Nineteenth Conference on Computational Natural Language Learning: Shared Task, pages 71–77,
Beijing, China, July 26-31, 2015. ©2014 Association for Computational Linguistics

$Arg1$, $Arg2$ (and Connectives), we used the relational phrase embedding from these triples (or pairs) based on their phrase embeddings (Bordes et al., 2013).

The first type of embedding we used in this paper is a combination of paragraph vector (Le and Mikolov, 2014) and translational embeddings (Bordes et al., 2013). First, the abstraction of each variable $Arg1$ and $Arg2$ was built independently in a vertical way, and then the relation among these ($Arg1$,$Conn$,$Arg2$) and ($Arg1$,$Arg2$) are examined in a collective way. This is shown in Figure 3. This model has two intermediate embeddings: paragraph vector embeddings of $Arg1$, $Arg2$, and $Conn$, and translational embedding of ($Arg1$,$Conn$,$Arg2$) and ($Arg1$,$Arg2$).

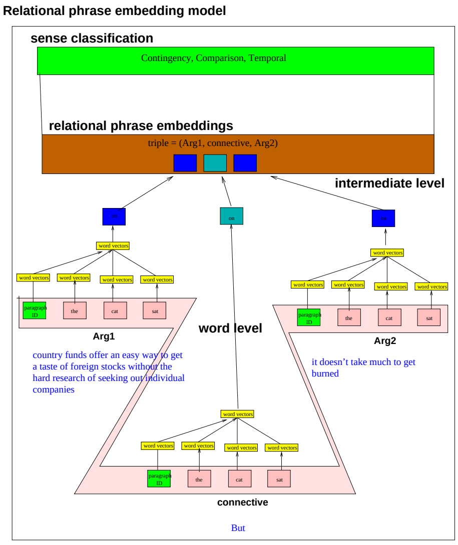

Figure 1: Figure shows relational paragraph embeddings.

We use a paragraph vector model to obtain the feature for $Arg1$ and $Arg2$ (Le and Mikolov, 2014). The paragraph vector model is an idea to obtain a real-valued vector in the similar construction with the word vector model (or word2vec) (Mikolov et al., 2013b) where the detailed explanation can be obtained.

In implicit/explicit sense classification, the participated items related to this classification are two for implicit relations of a pair $(Arg1, Arg2)$ and three for explicit relations of a triple $(Arg1, Conn, Arg2)$. This is by nature a multiple-instance learning setting (Dietterich et al., 1997), which receives a set of instances which are labeled collectively instead of individually labeled where each contains many instances. All the more, linguistic characteristics of discourse relations support this: meaning/sense is attached not to a single argument $Arg1$ or $Arg2$ but to a pair $(Arg1, Arg2)$ or a triple $(Arg1, Conn, Arg2)$.

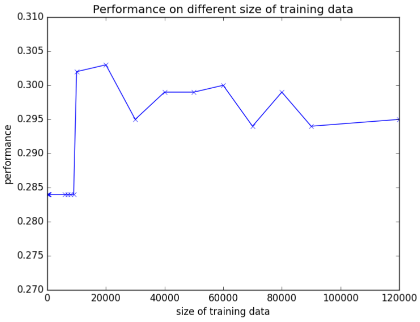

Figure 2: Figure shows a scalability of implicit classification performance based on the size of additional training data. We used dev set and used resources from WestBurry version of wikipedia corpus[6]and WMT14[7].

Followed by Bordes et al. (Bordes et al., 2011; Bordes et al., 2013), we minimized a margin-based ranking criterion over the pair of embeddings:

$$\mathcal{L} = \sum_{(Arg1,Arg2)\in S} \sum_{(Arg1,Arg2)\in S'} [\gamma + d(Arg1', Arg2) - d(Arg1, Arg2')]_+$$

where $[x]_+$ denotes the positive part of x, $\gamma > 0$ is a margin hyperparameter. S' denotes a set of corrupted pair where $Arg1$ or $Arg2$ is replaced by a random entity (but not both at the same time). Readers should see the detailed explanation in (Bordes et al., 2013).

It is noted that we tried indicator function (alternatively called discrete-valued vector, bucket function (Bansal et al., 2014), binarization of embeddings (Guo et al., 2014)) for embeddings which are converted from real-valued vector. Although we have not tested sufficiently due to the timing constraint, we did not include this method in our experiments since we could not have any gain.

2.2 Production Features for Constituent Parsing

(Lin et al., 2014) describes the method using the production features based on the parsing results.

[6] www.psych.ualberta.ca/ westburylab.

[7] www.statmt.org/wmt14.

Subtree extract	extracted		not extracted	
Exact match	16582	0.347	31265	0.653
+-1 position	39096	0.817	8751	0.183
Combi 2 elem	43031	0.899	4816	0.101
Combi 3 elem	45102	0.943	2745	0.057
Combi 4 elem	45872	0.959	1975	0.041

Table 2: Extraction of production features for constituent parsing results.

In this paper, we further process and treat these as the phrase embeddings. The algorithm is as follows. First, the subset of (constituent) parsing results which correspond to $Arg1$ and $Arg2$ are extracted. Then, all the production rules for these subtrees are derived. Third, we apply these production rules into the relational phrase embedding model that we described in 2.1. We replace all the words in 2.1 with production rules.

2.3 Word-Pair Features

Word-pair features in discourse parsing indicate the Cartesian products of all the combinations of words in $Arg1$ and $Arg2$. This feature is used in (Lin et al., 2014; Rutherford and Xue, 2015). (Rutherford and Xue, 2015) further developed this method combined with Brown clustering (Brown et al., 1992). We use this by word-pair embedding.

The second type of embedding we used in this paper is an abstraction of word-pair embedding in $Arg1$, $Arg2$ (and $Conn$) in a horizontal way. This is shown in Figure 4. The word grows their bi-gram in terms of Cartesian product of elements in different $Arg1$ and $Arg2$ which has a order from $Arg1$ to $Arg2$ where this bi-gram is embedded in the word embedding. Followed by Pitler et al. (Pitler et al., 2008) we use the 100 frequent word-pairs in training set for each category of relation. We did not delete function words/stop-words.

2.4 Heuristic Features for Explicit Connectives

Heuristic features in this paper indicate the specific features used in the explicit sense classification: (1) connective, (2) POS of connective, and (3) connective + previous word (Lin et al., 2014). These three features are employed in order to resolve the ambiguity in discourse connectives, and practically work fairly efficiently.

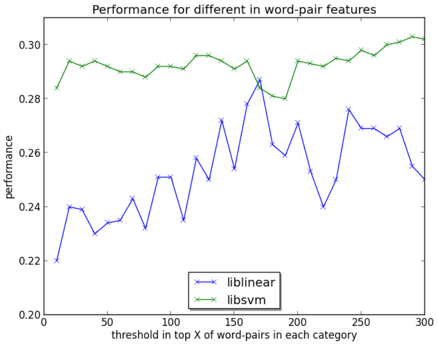

Figure 3: Figure shows the variation of the threshold in top X of word-pairs in each category. Most of the frequent word-pairs are functional word pairs, such as *the-the*, but we did not remove them.

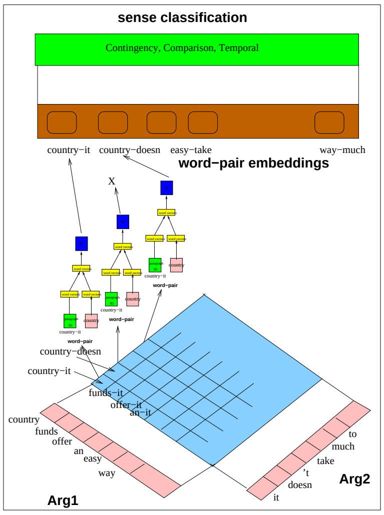

Figure 4: Figure show word-pair embeddings.

3 Experimental Settings

For the dataset, we used the CoNLL 2015 Shared task data set, i.e. LDC2015E21 (Xue et al., 2015) and Skip-gram neural word embeddings (Mikolov et al., 2013a)[8]). For the unofficial run, we used westbury version of English wikipedia dump (such as Figure 2) and WMT14 data set.[9]

We choose python as the language to develop our discourse parser. We use external tools such as libSVM (Chang and Lin, 2011), liblinear (Fan et al., 2008), wapiti (Lavergne et al., 2010), and maximum entropy model[10] for a classification task described as Section 2. Among these off-the-shelf classifiers, we used libSVM for the official re-

[8] https://code.google.com/p/word2vec

[9] www.statmt.org/wmt14.

[10] http://homepages.inf.ed.ac.uk/lzhang10/maxent.html

	Overall Task									Sense Classification					
	dev			test			blind			dev			test		
	f1	pr	rec	f1	pr	rec	f1	pr	rec	f1	pr	rec	f1	pr	rec
Overall															
Arg12	.291	.250	.348	.246	.210	.297	.215	.188	.252	1	1	1	1	1	1
Arg1	.392	.336	.469	.357	.304	.431	.317	.276	.371	1	1	1	1	1	1
Arg2	.422	.362	.505	.398	.339	.480	.382	.333	.448	1	1	1	1	1	1
conn	.863	.904	.827	.881	.903	.859	.794	.849	.746	1	1	1	1	1	1
parser	.154	.132	.184	.123	.105	.149	.107	.093	.125	.492	.812	.474	.466	.804	.458
sense	.081	.270	.099	.083	.207	.112	.041	.047	.065	.546	.546	.546	.531	.531	.531
Explicit Only															
Arg12	.186	.195	.178	.147	.150	.143	.111	.119	.104	1	1	1	1	1	1
Arg1	.263	.275	.252	.211	.216	.206	.167	.178	.157	1	1	1	1	1	1
Arg2	.373	.391	.357	.382	.392	.373	.281	.301	.264	1	1	1	1	1	1
conn	.863	.904	.827	.881	.903	.859	.794	.849	.746	1	1	1	1	1	1
parser	.158	.166	.152	.132	.136	.129	.079	.084	.074	.707	.882	.694	.727	.726	.838
sense	.138	.263	.142	.108	.175	.110	.077	.077	.084	.838	.838	.838	.873	.873	.873
Implicit Only															
Arg12	.355	.275	.501	.307	.237	.436	.276	.217	.378	1	1	1	1	1	1
Arg1	.453	.351	.640	.430	.332	.610	.392	.309	.538	1	1	1	1	1	1
Arg2	.451	.349	.638	.407	.314	.578	.441	.347	.603	1	1	1	1	1	1
conn	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
parser	.151	.117	.213	.117	.091	.166	.123	.097	.169	.283	.283	.283	.221	.221	.221
sense	.019	.699	.052	.025	.667	.061	.016	.598	.046	.105	.803	.136	.112	.891	.149

Table 3: Official results for task of identification of connectives and arguments. Table shows the results for dev set, test set and blind test set.

sults. Additionally we use word2vec (Mikolov et al., 2013b) and Theano (Bastien et al., 2012)[11] in the pipeline.

One bottleneck of our system was in a training procedure. Since a paragraph vector is currently not incrementally trainable, we were not able to separate training and test phases. Hence, we need to run it all on TIRA,[12] whose computing resource is powerless which took a considerable time such as 15 to 30 minutes where most of other participants only finish their run in 30 seconds or so.

4 Experimental Results

Table 3 shows our results. There are fifteen columns where the nine columns in the left show the overall task while the six columns in the right shows the supplementary task.[13]

In terms of the evaluation for explicit connectives, we obtained F score of 0.138, 0.108, and 0.077 for dev/test/blind sets for overall task (the lowest low in the second group) while we obtained F score of 0.707 for sense classification task. For the connectives, F score was 0.863 while Arg 1-2 was 0.186 which was fairly low. This may be result in the policy of the evaluation script which checks the correct classification results together with the correct identification of triples $(Arg1, Conn, Arg2)$. Hence, even if the classification results were correct if the triples $(Arg1, Conn, Arg2)$ were not correctly identified, the results were not correct. Thus, this explains why there is a big difference between the overall task (left nine columns) and the sense classification task (right three columns), as well as the low scores of 0.138, 0.108 and 0.077.

For the implicit only evaluation, on contrast, we obtained F score of 0.019, 0.025, and 0.016 (the lowest row in the third group) for overall task and 0.105 for sense classification task. Here, precision was high (precision of these which were 0.699, 0.667, and 0.598) for overall task and 0.803 and 0.891 for sense classification task; while recall

[11] http://deeplearning.net/tutorial/rnnslu.html

[12] http://www.tira.io

[13] Due to the unforeseen errors occurred on TIRA, we could not obtain the results for blind test set.

		dev set (official results)						dev set (unofficial results)								
		Explicit			Implicit			Implicit(30m)			Implicit (prod)			Implicit (wp)		
		pr	rec	f1	pr	rec	f1	pr	rec	f1	pr	rec	f1	pr	rec	f1
1	Comp	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
2	Comp.Conc	.13	.17	.14	1	0	0	1	0	1	0	0	0	0	0	0
3	Comp.Cont	.79	.84	.82	1	0	0	1	0	0	.05	.02	.03	.16	.2	.18
4	Cont.Cau.Rea	.96	.58	.72	1	0	0	1	0	0	.09	.10	.03	.24	.10	.14
5	Cont.Cau.Res	1	.84	.91	1	0	0	1	0	0	.08	.08	.08	.15	.08	.10
6	EntRel	–	–	–	.28	.95	.44	.29	.98	.45	.24	.91	.43	.36	.69	.47
7	Exp	–	–	–	1	0	0	1	0	0	1	0	0	1	0	0
8	Cont.Cond	1	.89	.94	–	–	–	–	–	–	–	–	–	–	–	–
9	Exp.Alt	.86	1	.92	1	0	0	1	0	0	1	0	0	1	0	0
10	Exp.Alt.C alt	1	.83	.91	1	0	0	1	0	0	0	0	0	1	0	0
11	Exp.Conj	.96	.96	.96	.26	.04	.07	.54	.11	.18	.17	.15	.16	.35	.26	.30
12	Exp.Inst	.90	1	.95	0	0	0	.75	.06	.12	.09	.23	.13	.43	.06	.11
13	Exp.Rest	1	.33	.50	.50	.04	.07	.33	.01	.02	.14	.16	.15	.11	.06	.08
14	Temp.As.Pr	.94	.96	.95	1	0	0	1	0	0	.13	.04	.06	0	0	0
15	Temp.As.Su	.95	.73	.82	1	0	0	1	0	0	0	0	0	1	0	0
16	Temp.Syn	.62	.97	.76	1	0	0	1	0	0	.08	.10	.09	0	0	0
17	Average	.88	.69	.71	.80	.14	.11	.86	.14	.11	.21	.14	.07	.39	.16	.09
18	Overall	.84	.84	.84	.28	.28	.28	.30	.30	.30	.16	.16	.16	.29	.29	.29

Table 4: Results for devset (Official and unofficial results). Implicit only includes Implicit, EntRel, and AltLex. This experiment uses the development set. The right most column *Implicit only(30m)* shows the results with additional data of 30M sentence pairs using the same setting of Figure 2.

was very low.

Table 5 shows the detailed results for sense classification under the setting that identification of connectives and arguments are correct. The first group (the left three columns) show the results for explicit classification. On contrast to implicit classification all the figures are considerably good except Comp.Conc whose F score was 0.14. The second group to the fifth group (the rightmost three columns) show four configurations of implicit classification. The third group shows the 30 million additional sentence pairs for training, the fourth group uses production feature, and the fifth group uses word-pair feature. These three groups exposed each characteristics quite clearly. Relational phrase embeddings (Implicit and Implicit(30m)) works for Expansion group (Exp.Conj, Exp.Inst, Exp.Rest), the production feature (marked as Implicit(prod)) worked for Temporal group (Temp.As.Pr and TempSyn), and the word-pair feature (marked as Implicit(wp)) worked for Comparison/Contingency groups. The effect of additional data was shown in the third group (marked as Implicit(30m)). This group was given additional data of 30M sentence pairs which improved the performance on Exp.Conj (from F score 0.07 to 0.18), and Exp.Inst (from F score 0.00 to 0.12) while Exp.Rest was down from fi score 0.07 to 0.02. The effect was limited to these categories.

It is easily observed that if the surface form of connective does not share multiple senses, such as *if* (67%) in Cont.Cond and *instead* (87%) in Exp.Alt.C, the results of sense classification performed good where Cont.Cond was F score of 0.94 and Exp.Alt.C was F score of 0.91. If the surface form of connective share multiple senses, they tend to be classified unbalancedly and one sense tends to be collected many votes. (For example, *But* has multiple senses, including Comp.Conc, Comp, and Comp.Cont. Comp.Cont collected many votes. As a result, the classification results for Comp.Cont was good but for others they were bad).

5 Discussion

A paragraph vector is proven useful for the sentiment analysis-typed task (Le and Mikolov, 2014). The word embedding is propagated towards the parent node and averaged. Our intension was that

	test set					
	Explicit			Implicit		
	pr	rec	f1	pr	rec	f1
1	1	0	0	–	–	–
2	.41	.59	.48	1	0	0
3	.91	.83	.87	1	0	0
4	1	.75	.86	1	0	0
5	1	.97	.99	1	0	0
6	–	–	–	.22	.96	.35
7	–	–	–	1	0	0
8	1	.81	.89	–	–	–
9	.83	1	.91	–	–	–
10	1	1	1	1	0	0
11	.98	.98	.98	.25	.09	.13
12	1	1	1	1	.04	.08
13	1	.29	.44	1	0	0
14	.92	1	.96	1	0	0
15	.94	.69	.79	1	0	0
16	.58	.98	.73	1	0	0
17	.84	.73	.73	.89	.15	.11
18	.87	.87	.87	.22	.22	.22

Table 5: Official results for explicit/implicit sense classification for test set.

the averaged embedding in a sentence will perform meaning establishment in the intermediate representation which capture the characteristics of $Arg1$, $Arg2$, and $Conn$. First, Comp.Cont or Comp.Conc may include sentence polarity with some additional condition that these polarities may be reversed. Against our expectation only a handful of examples were classified in these categories. However, if they are classified in these categories they were correct, i.e. precision 1. Second, if $Arg1$ and $Arg2$ are required to expose the causal relation such as Cont.Cau.Rea and Cont.Cau.Res this may be beyond the framework of a paragraph vector. Third, our implicit classification tried to classify Exp.Conj and Exp.Rest. Both of these categories of relation can be found some similarities with sentiment analysis/polarities, which can be reasonable that it worked for these categories. Four, interestingly, the word-pair feature works for Comparison/Contingency sense group while the production feature works (only slightly though) for Temporal sense group.

We used a margin-based ranking criteria to obtain relations over a paragraph vectors. First, (Mikolov et al., 2013b) observed a linear relation on two word embeddings. However, it might be too heavy expectation for two paragraph embeddings which can capture the similar phenomenon. Even if $Arg1$ consists of many words, a paragraph vector will average their word embeddings. In this sense this approach may have a crucial limit together with the fact that this is unsupervised learning. Second, we do not know yet but some small trick may improve the relation of Comp.Cont or Comp.Conc since these relations are quite similar relations with Exp.Conj, Exp.Instantiation, and Exp.Rest except that these relations are the polarities reversed.

6 Conclusion

This paper describes the discourse parsing system developed at Dublin City University for participation in the CoNLL 2015 shared task. We take an approach based on a paragraph vector. One shortcoming was that our classifier was effective only Exp.Conj, Exp.Inst and Exp.Rest despite our expectation that this model will work for Comp.Cont and Comp.Conc as well. The relation of the latter is in an opposite direction. We provided the word-pair model which works for these categories but in a different perspective.

Further work includes the mechanism how to make it work for Comp.Cont and Comp.Conc. Although a paragraph vector did not work efficiently, our model has a tentative model which does not have interaction between relational, paragraph, and word embeddings such as in (Denil et al., 2015), which is one immediate challenge. Then, other challenge includes replacement of a paragraph vector model with a convolutional sentence vector model (Kalchbrenner et al., 2014) and RNN-LSTM model (Le and Zuidema, 2015). The former approach is related to the supervised learning instead of unsupervised learning. The latter approach is to employ the structure of tree instead of a sequence.

Acknowledgment

This research is supported by the Science Foundation Ireland (Grant 12/CE/I2267) as part of the ADAPT at Dublin City University. We would also like to thank ICHEC, the CoNLL 2015 Shared Task Committee, and Martin Potthast.

References

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. *In Proceedings of the Association for Computational Linguistics (ACL 2014).*

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. *Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.*

Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. 2011. Learning structured embeddings of knowledge bases. *In Proceeding at the Learning Workshop.*

A Bordes, N Usunier, A Garcia-Duran, J Weston, and O Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, pages 2787–2795.

Peter F. Brown, P.V. deSouza, Robert L. Mercer, Vincent J.D Pietra, and J.C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, pages 467–479.

C.-C. Chang and C.-J. Lin. 2011. Libsvm : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

Misha Denil, Alban Demiraj, and Nando de Freitas. 2015. Extraction of salient sentences from labelled documents. *Technical Report at Oxford University.*

Thomas G. Dietterich, Richard H. Lathrop, and Tomas Lozano-Perez. 1997. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(12):3171.

R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, and C. J. Lin. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Revisiting embedding features for simple semi-supervised learning. *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2014).*

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, June.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513, July.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *In Proceedings of ICML.*

Phong Le and Willem Zuidema. 2015. Compositional distributional semantics with long short term memory. *In Proceedings of SemEval (*SEM 2015).*

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 (EMNLP 2009).*

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

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *ArXiv:1309.4168.*

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *In Proceedings of NIPS conference.*

Emily Pitler, Annie Louis, and Ani Nenkova. 2008. Automatic sense prediction for implicit discourse relations in text. *In Proceedings of the 14th Conference of the Association for Computational Linguistics (ACL 2009).*

Attapol Te Rutherford and Nianwen Xue. 2015. Improving the inference of implicit discourse relations via classifying explicit discourseconnectives. *In Proceedings of the NAACL-HLT.*

Langyue Wang, Chris Hokamp, Tsuyoshi Okita, Xiaojun Zhang, and Qun Liu. 2015. The dcu discourse parser for connective, argument identification and explicit sense classification. *In Proceedings of the Nineteenth Conference on Computational Natural Language Learning: Shared Task.*

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.*