A Reranking Model for Discourse Segmentation using Subtree Features

Ngo Xuan Bach, Nguyen Le Minh, Akira Shimazu
School of Information Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa, 923-1292, Japan
{bachnx,nguyenml,shimazu}@jaist.ac.jp

Abstract

This paper presents a discriminative reranking model for the discourse segmentation task, the first step in a discourse parsing system. Our model exploits subtree features to rerank N-best outputs of a base segmenter, which uses syntactic and lexical features in a CRF framework. Experimental results on the RST Discourse Treebank corpus show that our model outperforms existing discourse segmenters in both settings that use gold standard Penn Treebank parse trees and Stanford parse trees.

1 Introduction

Discourse structure has been shown to have an important role in many natural language applications, such as text summarization (Marcu, 2000; Louis et al., 2010), information presentation (Bateman et al., 2001), question answering (Sun and Chai, 2007), and dialogue generation (Hernault et al., 2008). To produce such kinds of discourse structure, several attempts have been made to build discourse parsers in the framework of Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), one of the most widely used theories of text structure.

In the RST framework, a text is first divided into several elementary discourse units (EDUs). Each EDU may be a simple sentence or a clause in a complex sentence. Consecutive EDUs are then put in relation with each other to build a discourse tree. Figure 1 shows an example of a discourse tree with three EDUs. The goal of the discourse segmentation task is to divide the input text into such EDUs.

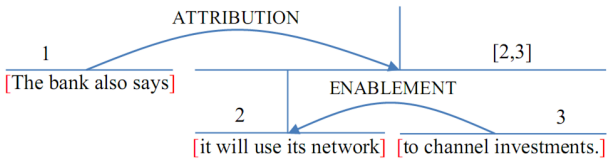

Figure 1: A discourse tree (Soricut and Marcu, 2003).

The quality of the discourse segmenter contributes a significant part to the overall accuracy of every discourse parsing system. If a text is wrongly segmented, no discourse parsing algorithm can build a correct discourse tree.

Existing discourse segmenters usually exploit lexical and syntactic features to label each word in a sentence with one of two labels, *boundary* or *no-boundary*. The limitation of this approach is that it only focuses on the boundaries of EDUs. It cannot capture features that describe whole EDUs.

Recently, discriminative reranking has been used successfully in some NLP tasks such as POS tagging, chunking, and statistical parsing (Collins and Koo, 2005; Kudo et al., 2005; Huang, 2008; Fraser et al., 2009). The advantage of the reranking method is that it can exploit the output of a base model to learn. Based on that output, we can extract long-distance non-local features to rerank.

In this paper, we present a reranking model for the discourse segmentation task. We show how to use subtree features, features extracted from whole EDUs, to rerank outputs of a base model. Experimental results on RST Discourse Treebank (RST-DT) (Carlson et al., 2002) show that our model out-

Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pages 160–168,
Seoul, South Korea, 5-6 July 2012.

performs existing systems.

The rest of this paper is organized as follows. Section 2 summarizes related work. Section 3 presents our method. Experimental results on RST-DT are described in Section 4. Finally, Section 5 gives conclusions.

2 Related Work

Several methods have been proposed to deal with the discourse segmentation task. Thanh et al. (2004) present a rule-based discourse segmenter with two steps. In the first step, segmentation is done by using syntactic relations between words. The segmentation algorithm is based on some principles, which have been presented in Corston (1998) and Carlson and Marcu (2001), as follows:

1. *The clause that is attached to a noun phrase can be recognised as an embedded unit. If the clause is a subordinate clause, it must contain more than one word.*

2. *Coordinate clauses and coordinate sentences of a complex sentence are EDUs.*

3. *Coordinate clauses and coordinate elliptical clauses of verb phrases (VPs) are EDUs. Coordinate VPs that share a direct object with the main VP are not considered as a separate discourse segment.*

4. *Clausal complements of reported verbs and cognitive verbs are EDUs.*

The segmenter then uses cue phrases to correct the output of the first step.

Tofiloski et al. (2009) describe another rule-based discourse segmenter. The core of this segmenter consists of 12 syntactic segmentation rules and some rules concerning a list of stop phrases, discourse cue phrases, and part-of-speech tags. They also use a list of phrasal discourse cues to insert boundaries not derivable from the parser's output.

Soricut and Marcu (2003) introduce a statistical discourse segmenter, which is trained on RST-DT to label words with *boundary* or *no-boundary* labels. They use lexical and syntactic features to determine the probabilities of discourse boundaries $P(b_i|w_i, t)$, where w_i is the i^{th} word of the input sentence s, t is the syntactic parse tree of s, and $b_i \in \{boundary, no\text{-}boundary\}$. Given a syntactic parse tree t, their algorithm inserts a discourse boundary after each word w for which $P(boundary|w, t) > 0.5$.

Another statistical discourse segmenter using artificial neural networks is presented in Subba and Di Eugenio (2007). Like Soricut and Marcu (2003), they formulate the discourse segmentation task as a binary classification problem of deciding whether a word is the *boundary* or *no-boundary* of EDUs. Their segmenter exploits a multilayer perceptron model with back-propagation algorithm and is also trained on RST-DT.

Hernault et al. (2010) propose a sequential model for the discourse segmentation task, which considers the segmentation task as a sequence labeling problem rather than a classification problem. They exploit Conditional Random Fields (CRFs) (Lafferty et al., 2001) as the learning method and get state-of-the-art results on RST-DT.

In our work, like Hernault et al. (2010), we also consider the discourse segmentation task as a sequence labeling problem. The final segmentation result is selected among N-best outputs of a CRF-based model by using a reranking method with subtree features.

3 Method

3.1 Discriminative Reranking

In the discriminative reranking method (Collins and Koo, 2005), first, a set of candidates is generated using a base model (GEN). GEN can be any model for the task. For example, in the part-of-speech (POS) tagging problem, GEN may be a model that generates all possible POS tags for a word based on a dictionary. Then, candidates are reranked using a linear score function:

$$score(y) = \Phi(y) \cdot W$$

where y is a candidate, $\Phi(y)$ is the feature vector of candidate y, and W is a parameter vector. The final output is the candidate with the highest score:

$$\begin{aligned} F(x) &= argmax_{y \in GEN(x)} score(y) \\ &= argmax_{y \in GEN(x)} \Phi(y) \cdot W. \end{aligned}$$

To learn the parameter W we use the average perceptron algorithm, which is presented as Algorithm 1.

Algorithm 1 Average perceptron algorithm for reranking (Collins and Koo, 2005)

Inputs: Training set $\{(x^i, y^i)|x^i \in R^n, y^i \in C, \forall i = 1, 2, \ldots, m\}$
Initialize: $W \leftarrow 0, W_{avg} \leftarrow 0$
Define: $F(x) = argmax_{y \in GEN(x)} \Phi(y) \cdot W$
for $t = 1, 2, \ldots, T$ **do**
  for $i = 1, 2, \ldots, m$ **do**
    $z^i \leftarrow F(x^i)$
    if $z^i \neq y^i$ **then**
      $W \leftarrow W + \Phi(y^i) - \Phi(z^i)$
    end if
    $W_{avg} \leftarrow W_{avg} + W$
  end for
end for
$W_{avg} \leftarrow W_{avg}/(mT)$
Output: Parameter vector W_{avg}.

In the next sections we will describe our base model and features that we use to rerank candidates.

3.2 Base Model

Similar to the work of Hernault et al. (2010), our base model uses Conditional Random Fields[1] to learn a sequence labeling model. Each label is either *beginning* of EDU (B) or *continuation* of EDU (C). Soricut and Marcu (2003) and Subba and Di Eugenio (2007) use *boundary* labels, which are assigned to words at the end of EDUs. Like Hernault et al. (2010), we use *beginning* labels, which are assigned to words at the beginning of EDUs. However, we can convert an output with *boundary*, *no-boundary* labels to an output with *beginning*, *continuation* labels and vice versa. Figure 2 shows two examples of segmenting a sentence into EDUs and their correct label sequences.

We use the following lexical and syntactic information as features: words, POS tags, nodes in parse trees and their lexical heads and their POS heads[2]. When extracting features for word w, let r be the

[1]We use the implementation of Kudo (Kudo, CRF++).

[2]Lexical heads are extracted using Collins' rules (Collins, 1999).

[The bank also says] [it will use its network] [to channel investments.] (Soricut and Marcu, 2003)

Label sequence: B C C C B C C C C B C C C

[Then,] [when it would have been easier to resist them,] [nothing was done] [and my brother was murdered by the drug mafias three years ago.] (RST Discourse Treebank)

Label sequence: B C B C C C C C C C C B C C B C C C C C C C C C C C

Figure 2: Examples of segmenting sentences into EDUs.

word on the right-hand side of w and N_p be the deepest node that belongs to both paths from the root to w and r. N_w and N_r are child nodes of N_p that belong to two paths, respectively. Figure 3 shows two partial lexicalized syntactic parse trees. In the first tree, if w = $says$ then r = it, N_p = $VP(says)$, N_w = $VBZ(says)$, and N_r = $SBAR(will)$. We also consider the parent and the right-sibling of N_p if any. The final feature set for w consists of not only features extracted from w but also features extracted from two words on the left-hand side and two words on the right-hand side of w.

Our feature extraction method is different from the method in previous work (Soricut and Marcu, 2003; Hernault et al., 2010). They define N_w as the highest ancestor of w that has lexical head w and has a right-sibling. Then N_p and N_r are defined as the parent and right-sibling of N_w. In the first example, our method gives the same results as the previous one. In the second example, however, there is no node with lexical head "*done*" and having a right-sibling. The previous method cannot extract N_w, N_p, and N_r in such cases. We also use some new features such as the head node and the right-sibling node of N_p.

3.3 Subtree Features for Reranking

We need to decide which kinds of subtrees are useful to represent a candidate, a way to segment the input sentence into EDUs. In our work, we consider two kinds of subtrees: *bound trees* and *splitting trees*.

The *bound tree* of an EDU, which spans from word u to word w, is a subtree which satisfies two conditions:

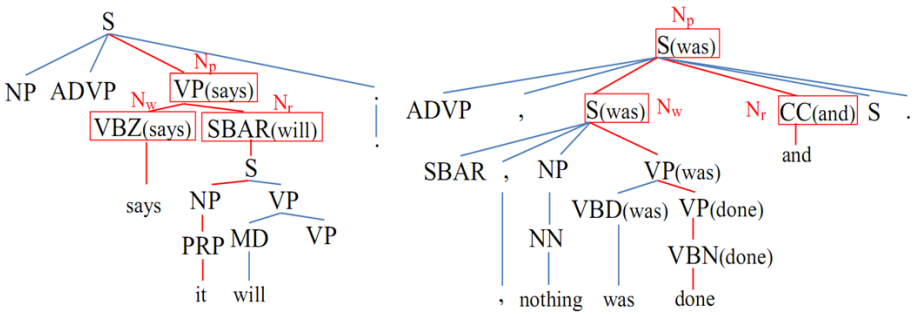

Figure 3: Partial lexicalized syntactic parse trees.

1. its root is the deepest node in the parse tree which belongs to both paths from the root of the parse tree to u and w, and

2. it only contains nodes in two those paths.

The *splitting tree* between two consecutive EDUs, from word u to word w and from word r to word v, is a subtree which is similar to a bound tree, but contains two paths from the root of the parse tree to w and r. Hence, a *splitting tree* between two consecutive EDUs is a *bound tree* that only covers two words: the last word of the first EDU and the first word of the second EDU. Bound trees will cover the whole EDUs, while splitting trees will concentrate on the boundaries of EDUs.

From a bound tree (similar to a splitting tree), we extract three kinds of subtrees: subtrees on the left path (*left tree*), subtrees on the right path (*right tree*), and subtrees consisting of a subtree on the left path and a subtree on the right path (*full tree*). In the third case, if both subtrees on the left and right paths do not contain the root node, we add a pseudo root node. Figure 4 shows the bound tree of EDU "*nothing was done*" of the second example in Figure 3, and some examples of extracted subtrees.

Each subtree feature is then represented by a string as follows:

- A left tree (or a right tree) is represented by concatenating its nodes with hyphens between nodes. For example, subtrees (b) and (e) in Figure 4 can be represented as follows:

  S-NP-NN-nothing, and

  S-VP-VP-VBN-done.

- A full tree is represented by concatenating its left tree and right tree with string ### in the middle. For example, subtrees (g) and (h) in Figure 4 can be represented as follows:

  S-NP-NN###S-VP-VP-VBN, and

  NP-NN-nothing###VP-VP-VBN.

The feature set of a candidate is the set of all subtrees extracted from bound trees of all EDUs and splitting trees between two consecutive EDUs.

Among two kinds of subtrees, splitting trees can be computed between any two adjacent words and therefore can be incorporated into the base model. However, if we do so, the feature space will be very large and contains a lot of noisy features. Because many words are not a boundary of any EDU, many subtrees extracted by this method will never become a *real* splitting tree (tree that splits two EDUs). Splitting trees extracted in the reranking model will focus on a small but compact and useful set of subtrees.

4 Experiments

4.1 Data and Evaluation Methods

We tested our model on the RST Discourse Treebank corpus. This corpus consists of 385 articles from the Penn Treebank, which are divided into a Training set and a Test set. The Training set consists of 347 articles (6132 sentences), and the Test set consists of 38 articles (991 sentences).

There are two evaluation methods that have been used in previous work. The first method measures only *beginning* labels (B labels) (Soricut and Marcu, 2003; Subba and Di Eugenio, 2007). The second

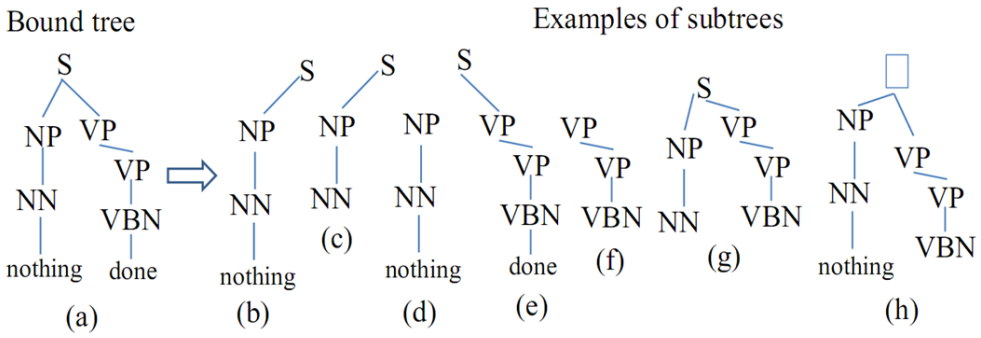

Figure 4: Subtree features.

method (Hernault et al., 2010) measures both *beginning* and *continuation* labels (B and C labels)[3]. This method first calculates scores on B labels and scores on C labels, and then produces the average of them. Due to the number of C labels being much higher than the number of B labels, the second evaluation method yields much higher results. In Hernault et al. (2010), the authors compare their systems with previous work despite using different evaluation methods. Such comparisons are not valid. In our work, we measure the performance of the proposed model using both methods.

Table 1: Performance when evaluating on B labels

Model	Trees	Pre(%)	Re(%)	F_1(%)
SPADE	Penn	84.1	85.4	84.7
NNDS	Penn	85.5	86.6	86.0
CRFSeg	Penn	92.7	89.7	91.2
Base	Penn	92.5	92.5	92.5
Reranking	Penn	**93.1**	**94.2**	**93.7**
CRFSeg	Stanford	91.0	87.2	89.0
Base	Stanford	91.4	90.1	90.7
Reranking	Stanford	**91.5**	**90.4**	**91.0**
Human	-	98.5	98.2	98.3

4.2 Experimental Results

We learned the base model on the Training set and tested on the Test set to get N-best outputs to rerank. To learn parameters of the reranking model, we conducted 5-fold cross-validation tests on the Training set. In all experiments, we set N to 20. To choose the number of iterations, we used a development set, which is about 20 percent of the Training set.

Table 1 shows experimental results when evaluating only *beginning* (B) labels, in which SPADE is the work of Soricut and Marcu (2003), NNDS is a segmenter that uses neural networks (Subba and Di Eugenio, 2007), and CRFSeg is a CRF-based segmenter (Hernault et al., 2010). When using gold parse trees, our base model got 92.5% in the F_1 score, which improves 1.3% compared to the state-of-the-art segmenter (CRFSeg). When using Stanford parse trees (Klein and Manning, 2003), our base model improved 1.7% compared to CRFSeg. It demonstrates the effectiveness of our feature extraction method in the base model. As expected, our reranking model got higher results compared to the base model in both settings. The reranking model got 93.7% and 91.0% in two settings, which improves 2.5% and 2.0% compared to CRFSeg. Also note that, when using Stanford parse trees, our reranking model got competitive results with CRFSeg when using gold parse trees (91.0% compared to 91.2%).

Table 2 shows experimental results when evaluating on both *beginning* and *continuation* labels. Our models also outperformed CRFSeg in both settings, using gold parse trees and using Stanford parse trees (96.6% compared to 95.3% in the first setting, and 95.1% compared to 94.1% in the second setting).

Both evaluation methods have a weak point in that they do not measure the ability to find EDUs exactly. We suggest that the discourse segmentation task should be measured on EDUs rather than boundaries of EDUs. Under this evaluation scheme, our model achieved 90.0% and 86.2% when using

[3]Neither evaluation method counts sentence boundaries.

Table 2: Performance when evaluating on B and C labels

Model	Trees	Pre(%)	Re(%)	F_1(%)
CRFSeg	Penn	96.0	94.6	95.3
Base	Penn	96.0	96.0	96.0
Reranking	Penn	**96.3**	**96.9**	**96.6**
CRFSeg	Stanford	95.0	93.2	94.1
Base	Stanford	95.3	94.7	95.0
Reranking	Stanford	**95.4**	**94.9**	**95.1**

gold parse trees and Stanford parse trees, respectively.

We do not compare our segmenter to systems described in Thanh et al. (2004) and Tofiloski et al. (2009). Thanh et al. (2004) evaluated their system on only 8 texts of RST-DT with gold standard parse trees. They achieved 81.4% and 79.2% in the precision and recall scores, respectively. Tofiloski et al. (2009) tested their system on only 3 texts of RST-DT and used different segmentation guidelines. They reported a precision of 82.0% and recall of 86.0% when using Stanford parse trees.

An important question is which subtree features were useful for the reranking model. This question can be answered by looking at the weights of subtree features (the parameter vector learned by the average perceptron algorithm). Table 3 shows 30 subtree features with the highest weights in absolute value. These features are thus useful for reranking candidates in the reranking model. We can see that most subtree features at the top are *splitting trees*, so *splitting trees* have a more important role than *bound trees* in our model. Among three types of subtrees (*left tree*, *right tree*, and *full tree*), *full tree* is the most important type. It is understandable because subtrees in this type convey much information; and therefore describe *splitting trees* and *bound trees* more precise than subtrees in other types.

4.3 Error Analysis

This section discusses the cases in which our model fails to segment discourses. Note that all errors belong to one of two types, *over-segmentation* type (i.e., words that are not EDU boundaries are mistaken for boundaries) and *miss-segmentation* type (i.e., words that are EDU boundaries are mistaken for not boundaries).

Table 4: Top error words

Word	Percentage among all errors (%)
to	14.5
and	5.8
that	4.6
the	4.6
“	3.5
he	2.3
it	2.3
of	2.3
without	2.3
–	1.7
as	1.7
if	1.7
they	1.7
when	1.7
a	1.2

Tabel 4 shows 15 most frequent words for which our model usually makes a mistake and their percentage among all segmentation errors. Most errors are related to coordinating conjunctions and subordinators (*and*, *that*, *as*, *if*, *when*), personal pronouns (*he*, *it*, *they*), determiners (*the*, *a*), prepositions (*of*, *without*), punctuations (quotes and hyphens), and the word *to*.

Figure 5 shows some errors made by our model. In these examples, gold (correct) EDU boundaries are marked by bracket squares ([]), while predicted boundaries made by our model are indicated by arrows (↓ or ↑). A down arrow (↓) shows a boundary which is predicted correctly, while an up arrow (↑) indicates an *over-segmentation* error. A boundary with no arrow means a *miss-segmentation* error. For example, in Sentence 1, we have a correct boundary and an *over-segmentation* error. Sentences 2 and 3 show two *over-segmentation* errors, and sentences 4 and 6 show two *miss-segmentation* errors.

We also note that many errors occur right after punctuations (commas, quotes, hyphens, brackets, and so on). We analyzed statistics on words that appear before error words. Table 5 shows 10 most frequent words and their percentage among all errors. Overall, more than 35% errors occur right after punctuations.

Table 3: Top 30 subtree features with the highest weights

Type of tree	Type of subtree	Subtree feature	Weight
Splitting tree	Full tree	NP###NP-VP	23.0125
Splitting tree	Full tree	VP###S-VP	19.3044
Splitting tree	Full tree	NP###VBN	18.3862
Splitting tree	Right tree	VP	-18.3723
Splitting tree	Full tree	NP###SBAR	17.7119
Splitting tree	Full tree	NP###NP-SBAR	17.0678
Splitting tree	Full tree	NP###,	-16.6763
Splitting tree	Full tree	NP###VP	15.9934
Splitting tree	Left tree	NP-VP	15.2849
Splitting tree	Full tree	NP###NP	15.1657
Splitting tree	Right tree	SBAR	14.6778
Splitting tree	Full tree	NP###S-NP	14.4962
Splitting tree	Full tree	NP###S	13.1656
Bound tree	Full tree	S-PP###,	12.7428
Splitting tree	Full tree	NP###NP-VP-VBN	12.5210
Bound tree	Full tree	NP###NP	-12.4723
Bound tree	Full tree	VP###VP	-12.1918
Splitting tree	Full tree	NP-VP###S	12.1367
Splitting tree	Right tree	NP-VP	12.0929
Splitting tree	Full tree	NP-SBAR###VP	12.0858
Splitting tree	Full tree	NP-SBAR-S###VP	12.0858
Splitting tree	Full tree	VP###VP-VP	-12.0338
Bound tree	Full tree	VBG###.	11.9067
Bound tree	Right tree	:	11.8833
Bound tree	Full tree	VP###S	-11.7624
Bound tree	Full tree	S###VP	-11.7596
Bound tree	Full tree	“###”	11.5524
Bound tree	Full tree	S###,	11.5274
Splitting tree	Full tree	NP###VP-VBN	11.3342
Bound tree	Left tree	0	11.2878

Sentence 1: [With the fall social season well under way, name-droppers are out in force,] [1A]↓ [trying to impress their betters ↑ and sometimes put down their lessers.] [1B]

Sentence 2: [But it's not only the stock market ↑ that has some small investors worried.] [2A]

Sentence 3: ["I am ready at any moment ↑ to compete with a state farm."] [3A]

Sentence 4: [The Department of Housing and Urban Development has used testers] [4A] [to investigate discrimination in rental housing.] [4B]

Sentence 5: [Tell us what measure, ↑short of house arrest, ↑will get this Congress under control.] [5A]

Sentence 6: [Without machines,] [6A] [good farms can't get bigger.] [6B]

Figure 5: Some errors made by our model.

Table 5: Most frequent words that appear before error words

Word	Percentage among all errors (%)
,	24.9
“	5.2
–	2.3
time	1.7
)	1.2
assets	1.2
investors	1.2
month	1.2
plan	1.2
was	1.2

5 Conclusion

This paper presented a reranking model for the discourse segmentation task. Our model exploits subtree features to rerank N-best outputs of a base model, which uses CRFs to learn. Compared with the state-of-the-art system, our model reduces 2.5% among 8.8% errors (28.4% in the term of error rate) when using gold parse trees, and reduces 2% among 11% errors (18.2% in the term of error rate) when using Stanford parse trees. In the future, we will build a discourse parser that uses the described discourse segmenter.

Acknowledgments

This work was partially supported by the 21st Century COE program ‘Verifiable and Evolvable e-Society’, and Grant-in-Aid for Scientific Research, Education and Research Center for Trustworthy e-Society.

The authors would like to thank the three anonymous reviewers for the time they spent reading and reviewing this paper and Michael Strube for his comments during the revision process of the paper.

References

J. Bateman, J. Kleinz, T. Kamps, and K. Reichenberger. 2001. Towards Constructive Text, Diagram, and Layout Generation for Information Presentation. *Computational Linguistics*, 27(3), pp. 409-449.

L. Carlson and D. Marcu. 2001. Discourse Tagging Manual. *ISI Technical Report*, ISI-TR-545.

L. Carlson, D. Marcu, and M.E. Okurowski. 2002. RST Discourse Treebank. *Linguistic Data Consortium (LDC)*.

M. Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing. *Ph.D. Thesis*, University of Pennsylvania.

S. Corston-Oliver. 1998. Computing Representations of the Structure of Written Discourse. *Ph.D. Thesis*, University of California, Santa Barbara.

M. Collins and T. Koo. 2005. Discriminative Reranking for Natural Language Parsing. *Computational Linguistics*, 31(1), pp. 25-70.

A. Fraser, R. Wang, and H. Schütze. 2009. Rich Bitext Projection Features for Parse Reranking. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL)*, pp. 282-290.

H. Hernault, P. Piwek, H. Prendinger, and M. Ishizuka. 2008. Generating Dialogues for Virtual Agents Using Nested Textual Coherence Relations. In *Proceedings of IVA*, pp. 139-145.

H. Hernault, D. Bollegala, and M. Ishizuka. 2010. A Sequential Model for Discourse Segmentation. In *Proceedings of the 11th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pp. 315-326.

L. Huang. 2008. Forest Reranking: Discriminative Parsing with Non-Local Features. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 586-594

D. Klein and C. Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 423-430.

T. Kudo. CRF++: Yet Another CRF toolkit. Available at http://crfpp.sourceforge.net/

T. Kudo, J. Suzuki, and H. Isozaki. 2005. Boosting-based parse reranking with subtree features. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 189-196.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pp.282-289.

A. Louis, A. Joshi, and A. Nenkova 2010. Discourse indicators for content selection in summarization. In *Proceedings of the 11th annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)* , pp.147-156.

W.C. Mann and S.A. Thompson. 1988. Rhetorical Structure Theory. Toward a Functional Theory of Text Organization. *Text* 8, pp. 243-281.

D. Marcu. 2000. The Theory and Practice of Discourse Parsing and Summarization. MIT Press, Cambridge.

R. Soricut and D. Marcu. 2003. Sentence Level Discourse Parsing using Syntactic and Lexical Information. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 149-156.

R. Subba and B. Di Eugenio. 2007. Automatic Discourse Segmentation using Neural Networks. In *Proceedings of the Workshop on the Semantics and Pragmatics of Dialogue (SemDial)*, pp. 189-190.

M. Sun and J.Y. Chai. 2007. Discourse Processing for Context Question Answering Based on Linguistic Knowledge. *Knowledge-Based Systems*. 20(6), pp. 511-526.

H.L. Thanh, G. Abeysinghe, and C. Huyck. 2004. Automated Discourse Segmentation by Syntactic Information and Cue Phrases. In *Proceedings of IASTED*.

M. Tofiloski, J. Brooke, and M. Taboada. 2009. A Syntactic and Lexical-Based Discourse Segmenter. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pp. 77-80.