CODRA: A Novel Discriminative Framework for Rhetorical Analysis

Shafiq Joty*
Qatar Computing Research Institute

Giuseppe Carenini**
University of British Columbia

Raymond T. Ng†
University of British Columbia

Clauses and sentences rarely stand on their own in an actual discourse; rather, the relationship between them carries important information that allows the discourse to express a meaning as a whole beyond the sum of its individual parts. Rhetorical analysis seeks to uncover this coherence structure. In this article, we present CODRA*— a COmplete probabilistic Discriminative framework for performing Rhetorical Analysis in accordance with Rhetorical Structure Theory, which posits a tree representation of a discourse.*

CODRA *comprises a discourse segmenter and a discourse parser. First, the discourse segmenter, which is based on a binary classifier, identifies the elementary discourse units in a given text. Then the discourse parser builds a discourse tree by applying an optimal parsing algorithm to probabilities inferred from two Conditional Random Fields: one for intra-sentential parsing and the other for multi-sentential parsing. We present two approaches to combine these two stages of parsing effectively. By conducting a series of empirical evaluations over two different data sets, we demonstrate that* CODRA *significantly outperforms the state-of-the-art, often by a wide margin. We also show that a reranking of the k-best parse hypotheses generated by* CODRA *can potentially improve the accuracy even further.*

1. Introduction

A well-written text is not merely a sequence of independent and isolated sentences, but instead a sequence of structured and related sentences, where the meaning of a sentence relates to the previous and the following ones. In other words, a well-written

* Arabic Language Technologies, Qatar Computing Research Institute, Qatar Foundation, Doha, Qatar. E-mail: sjoty@qf.org.qa.
** Computer Science Department, University of British Columbia, Vancouver, BC, Canada, V6T 1Z4. E-mail: carenini@cs.ubc.ca.
† Computer Science Department, University of British Columbia, Vancouver, BC, Canada, V6T 1Z4. E-mail: rng@cs.ubc.ca.

Submission received: 11 May 2014; revised version received: 29 January 2015; accepted for publication: 18 March 2015.

doi:10.1162/COLI_a_00226

text has a **coherence structure** (Halliday and Hasan 1976; Hobbs 1979), which logically binds its clauses and sentences together to express a meaning as a whole. **Rhetorical analysis** seeks to uncover this coherence structure underneath the text; this has been shown to be beneficial for many Natural Language Processing (NLP) applications, including text summarization and compression (Marcu 2000b; Daumé and Marcu 2002; Sporleder and Lapata 2005; Louis, Joshi, and Nenkova 2010), text generation (Prasad et al. 2005), machine translation evaluation (Guzmán et al. 2014a, 2014b; Joty et al. 2014), sentiment analysis (Somasundaran 2010; Lazaridou, Titov, and Sporleder 2013), information extraction (Teufel and Moens 2002; Maslennikov and Chua 2007), and question answering (Verberne et al. 2007). Furthermore, rhetorical structures can be useful for other discourse analysis tasks, including co-reference resolution using Veins theory (Cristea, Ide, and Romary 1998).

Different formal theories of discourse have been proposed from different viewpoints to describe the coherence structure of a text. For example, Martin (1992) and Knott and Dale (1994) propose discourse relations based on the usage of discourse connectives (e.g., *because, but*) in the text. Asher and Lascarides (2003) propose Segmented Discourse Representation Theory, which is driven by sentence semantics. Webber (2004) and Danlos (2009) extend sentence grammar to formalize discourse structure. **Rhetorical Structure Theory (RST)**, proposed by Mann and Thompson (1988), is perhaps the most influential theory of discourse in computational linguistics. Although it was initially intended to be used in text generation, later it became popular as a framework for parsing the structure of a text (Taboada and Mann 2006). RST represents texts by labeled hierarchical structures, called Discourse Trees (DTs). For example, consider the DT shown in Figure 1 for the following text:

> *But he added: "Some people use the purchasers' index as a leading indicator, some use it as a coincident indicator. But the thing it's supposed to measure—manufacturing strength—it missed altogether last month."*

The leaves of a DT correspond to contiguous atomic text spans, called **elementary discourse units** (**EDUs**; six in the example). EDUs are clause-like units that serve as building blocks. Adjacent EDUs are connected by coherence relations (e.g., *Elaboration*, *Contrast*), forming larger discourse units (represented by internal nodes), which in turn are also subject to this relation linking. Discourse units linked by a rhetorical relation are

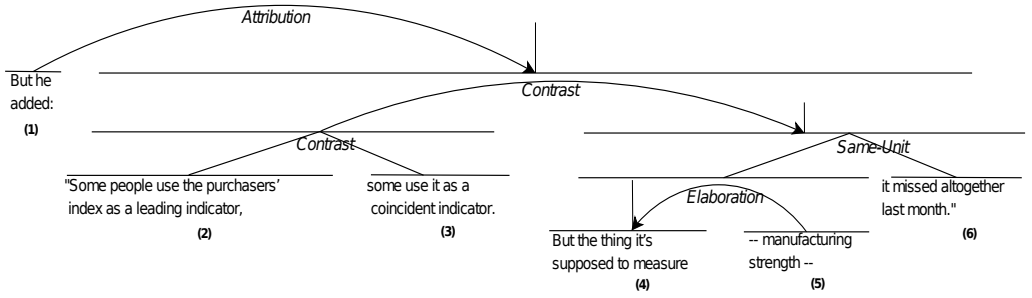


Figure 1
Discourse tree for two sentences in RST–DT. Each sentence contains three EDUs. Horizontal lines indicate text segments; satellites are connected to their nuclei by curved arrows and two nuclei are connected with straight lines.

further distinguished based on their relative importance in the text: **nuclei** are the core parts of the relation and **satellites** are peripheral or supportive ones. For example, in Figure 1, *Elaboration* is a relation between a nucleus (EDU 4) and a satellite (EDU 5), and *Contrast* is a relation between two nuclei (EDUs 2 and 3). Carlson, Marcu, and Okurowski (2002) constructed the first large RST-annotated corpus (**RST–DT**) on *Wall Street Journal* articles from the Penn Treebank. Whereas Mann and Thompson (1988) had suggested about 25 relations, the RST–DT uses 53 mono-nuclear and 25 multi-nuclear relations. The relations are grouped into 16 coarse-grained categories; see Carlson and Marcu (2001) for a detailed description of the relations. Conventionally, rhetorical analysis in RST involves two subtasks: **discourse segmentation** is the task of breaking the text into a sequence of EDUs, and **discourse parsing** is the task of linking the discourse units (EDUs and larger units) into a labeled tree. In this article, we use the terms *discourse parsing* and *rhetorical parsing* interchangeably.

While recent advances in automatic discourse segmentation have attained high accuracies (an F-score of 90.5% reported by Fisher and Roark [2007]), discourse parsing still poses significant challenges (Feng and Hirst 2012) and the performance of the existing discourse parsers (Soricut and Marcu 2003; Subba and Di-Eugenio 2009; Hernault et al. 2010) is still considerably inferior compared with the human gold standard. Thus, the impact of rhetorical structure in downstream NLP applications is still very limited. The work we present in this article aims to reduce this performance gap and take discourse parsing one step further. To this end, we address three key limitations of existing discourse parsers.

First, existing discourse parsers typically model the structure and the labels of a DT separately, and also do not take into account the sequential dependencies between the DT constituents. However, for several NLP tasks, it has recently been shown that joint models typically outperform independent or pipeline models (Murphy 2012, page 687). This is also supported in a recent study by Feng and Hirst (2012), in which the performance of a greedy bottom–up discourse parser improved when sequential dependencies were considered by using *gold* annotations for the neighboring (i.e., previous and next) discourse units as contextual features in the parsing model. To address this limitation of existing parsers, as the first contribution, we propose a novel discourse parser based on probabilistic discriminative parsing models, expressed as Conditional Random Fields (CRFs) (Sutton, McCallum, and Rohanimanesh 2007), to infer the probability of all possible DT constituents. The CRF models effectively represent the structure and the label of a DT constituent jointly, and, whenever possible, capture the sequential dependencies.

Second, existing discourse parsers typically apply greedy and sub-optimal parsing algorithms to build a DT. To cope with this limitation, we use the inferred (posterior) probabilities from our CRF parsing models in a probabilistic CKY-like bottom–up parsing algorithm (Jurafsky and Martin 2008), which is non-greedy and optimal. Furthermore, a simple modification of this parsing algorithm allows us to generate k-best (i.e., the k highest probability) parse hypotheses for each input text that could then be used in a **reranker** to improve over the initial ranking using additional (global) features of the discourse tree as evidence, a strategy that has been successfully explored in syntactic parsing (Charniak and Johnson 2005; Collins and Koo 2005).

Third, most of the existing discourse parsers do not discriminate between **intra-sentential parsing** (i.e., building the DTs for the individual sentences) and **multi-sentential parsing** (i.e., building the DT for the whole document). However, we argue that distinguishing between these two parsing conditions can result in more effective parsing. Two separate parsing models could exploit the fact that rhetorical relations

are distributed differently intra-sententially versus multi-sententially. Also, they could independently choose their own informative feature sets. As another key contribution of our work, we devise two different parsing components: one for intra-sentential parsing, the other for multi-sentential parsing. This provides for scalable, modular, and flexible solutions that can exploit the strong correlation observed between the text structure (i.e., sentence boundaries) and the structure of the discourse tree.

In order to develop a complete and robust discourse parser, we combine our intra-sentential and multi-sentential parsing components in two different ways. Because most sentences have a well-formed discourse sub-tree in the full DT (e.g., the second sentence in Figure 1), our first approach constructs a DT for every sentence using our intra-sentential parser, and then runs the multi-sentential parser on the resulting sentence-level DTs to build a complete DT for the whole document. However, this approach would fail in those cases where discourse structures violate sentence boundaries, also called "leaky" boundaries (Vliet and Redeker 2011). For example, consider the first sentence in Figure 1. It does not have a well-formed discourse sub-tree because the unit containing EDUs 2 and 3 merges with the next sentence and only then is the resulting unit merged with EDU 1. Our second approach, in order to deal with these leaky cases, builds sentence-level sub-trees by applying the intra-sentential parser on a sliding window covering two adjacent sentences and by then consolidating the results produced by overlapping windows. After that, the multi-sentential parser takes all these sentence-level sub-trees and builds a full DT for the whole document.

Our discourse parser assumes that the input text has already been segmented into elementary discourse units. As an additional contribution, we propose a novel discriminative approach to discourse segmentation that not only achieves state-of-the-art performance, but also reduces time and space complexities by using fewer features. Notice that the combination of our segmenter with our parser forms a COmplete probabilistic Discriminative framework for Rhetorical Analysis (CODRA).

Whereas previous systems have been tested on only one corpus, we evaluate our framework on texts from two very different genres: news articles and instructional how-to manuals. The results demonstrate that our approach to discourse parsing provides consistent and statistically significant improvements over previous methods both at the sentence level and at the document level. The performance of our final system compares very favorably to the performance of state-of-the-art discourse parsers. Finally, the oracle accuracy computed based on the *k*-best parse hypotheses generated by our parser demonstrates that a reranker could potentially improve the accuracy further.

After discussing related work in Section 2, we present our rhetorical analysis framework in Section 3. In Section 4, we describe our discourse parser. Then, in Section 5 we present our discourse segmenter. The experiments and analysis of results are presented in Section 6. Finally, we summarize our contributions with future directions in Section 7.

2. Related Work

Rhetorical analysis has a long history—dating back to Mann and Thompson (1988), when RST was initially proposed as a useful linguistic method for describing natural texts, to more recent attempts to automatically extract the rhetorical structure of a given text (Hernault et al. 2010). In this section, we provide a brief overview of the computational approaches that follow RST as the theory of discourse, and that are related to our work; see the survey by Stede (2011) for a broader overview that also includes other theories of discourse.

2.1 Unsupervised and Rule-Based Approaches

Although the most effective approaches to rhetorical analysis to date rely on supervised machine learning methods trained on human-annotated data, unsupervised methods have also been proposed, as they do not require human-annotated data and can be more easily applied to new domains.

Often, discourse connectives like *but, because,* and *although* convey clear information on the kind of relation linking the two text segments. In his early work, Marcu (2000a) presented a shallow rule-based approach relying on discourse connectives (or cues) and surface patterns. He used hand-coded rules, derived from an extensive corpus study, to break the text into EDUs and to build DTs for sentences first, then for paragraphs, and so on. Despite the fact that this work pioneered the field of rhetorical analysis, it has many limitations. First, identifying discourse connectives is a difficult task on its own, because (depending on the usage), the same phrase may or may not signal a discourse relation (Pitler and Nenkova 2009). For example, *but* can either signal a *Contrast* discourse relation or can simply perform non-discourse acts. Second, discourse segmentation using only discourse connectives fails to attain high accuracy (Soricut and Marcu 2003). Third, DT structures do not always correspond to paragraph structures; for example, Sporleder and Lapata (2004) report that more than 20% of the paragraphs in the RST–DT corpus (Carlson, Marcu, and Okurowski 2002) do not correspond to a discourse unit in the DT. Fourth, discourse cues are sometimes ambiguous; for example, *but* can signal *Contrast, Antithesis* and *Concession*, and so on.

Finally, a more serious problem with the rule-based approach is that often rhetorical relations are not explicitly signaled by discourse cues. For example, in RST–DT, Marcu and Echihabi (2002) found that only 61 out of 238 *Contrast* relations and 79 out of 307 *Cause–Explanation* relations were explicitly signaled by cue phrases. In the British National Corpus, Sporleder and Lascarides (2008) report that half of the sentences lack a discourse cue. Other studies (Schauer and Hahn 2001; Stede 2004; Taboada 2006; Subba and Di-Eugenio 2009) report even higher figures: About 60% of discourse relations are not explicitly signaled. Therefore, rather than relying on hand-coded rules based on discourse cues and surface patterns, recent approaches use *machine learning* techniques with a large set of informative features.

While some rhetorical relations need to be explicitly signaled by discourse cues (e.g., *Concession*) and some do not (e.g., *Background*), there is a large middle ground of relations that may be signaled or not. For these "middle ground" relations, can we exploit features present in the signaled cases to automatically identify relations when they are not explicitly signaled? The idea is to use unambiguous discourse cues (e.g., *although* for *Contrast*, *for example* for *Elaboration*) to automatically label a large corpus with rhetorical relations that could then be used to train a supervised model.[1]

A series of previous studies have explored this idea. Marcu and Echihabi (2002) first attempted to identify four broad classes of relations: *Contrast, Elaboration, Condition,* and *Cause–Explanation–Evidence*. They used a naive Bayes classifier based on word pairs (w_1, w_2), where w_1 occurs in the left segment, and w_2 occurs in the right segment. Sporleder and Lascarides (2005) included other features (e.g., words and their stems, Part-of-Speech [POS] tags, positions, segment lengths) in a boosting-based classifier (i.e., BoosTexter [Schapire and Singer 2000]) to further improve relation classification accuracy. However, these studies evaluated classification performance on the instances

1 We categorize this approach as unsupervised because it does not rely on human-annotated data.

where rhetorical relations were originally signaled (i.e., the discourse cues were artificially removed), and did not verify how well this approach performs on the instances that are not originally signaled. Subsequent studies (Blair-Goldensohn, McKeown, and Rambow 2007; Sporleder 2007; Sporleder and Lascarides 2008) confirm that classifiers trained on instances stripped of their original discourse cues do not generalize well to implicit cases because they are linguistically quite different.

Note that this approach to identifying discourse relations in the absence of manually labeled data does not fully solve the parsing problem (i.e., building DTs); rather, it only attempts to identify a small subset of coarser relations between two (flat) text segments (i.e., a tagging problem). Arguably, to perform a complete rhetorical analysis, one needs to use supervised machine learning techniques based on human-annotated data.

2.2 Supervised Approaches

Marcu (1999) applies supervised machine learning techniques to build a discourse segmenter and a shift–reduce discourse parser. Both the segmenter and the parser rely on C4.5 decision tree classifiers (Poole and Mackworth 2010) to learn the rules automatically from the data. The discourse segmenter mainly uses discourse cues, shallow-syntactic (i.e., POS tags) and contextual features (i.e., neighboring words and their POS tags). To learn the shift–reduce actions, the discourse parser encodes five types of features: lexical (e.g., discourse cues), shallow-syntactic, textual similarity, operational (previous n shift–reduce operations), and rhetorical sub-structural features. Despite the fact that this work has pioneered many of today's machine learning approaches to discourse parsing, it has all the limitations mentioned in Section 1.

The work of Marcu (1999) is considerably improved by Soricut and Marcu (2003). They present the publicly available **SPADE** system,[2] which comes with probabilistic models for discourse segmentation and *sentence-level* discourse parsing. Their segmentation and parsing models are based on lexico-syntactic patterns (or features) extracted from the lexicalized syntactic tree of a sentence. The discourse parser uses an optimal parsing algorithm to find the most probable DT structure for a sentence. SPADE was trained and tested on the RST–DT corpus. This work, by showing empirically the connection between syntax and discourse structure at the sentence level, has greatly influenced all major contributions in this area ever since. However, it is limited in several ways. First, SPADE does not produce a full-text (i.e., document-level) parse. Second, it applies a *generative* parsing model based on only lexico-syntactic features, whereas discriminative models are generally considered to be more accurate, and can incorporate arbitrary features more effectively (Murphy 2012). Third, the parsing model makes an independence assumption between the label and the structure of a DT constituent, and it ignores the sequential and the hierarchical dependencies between the DT constituents.

Subsequent research addresses the question of how much syntax one really needs in rhetorical analysis. Sporleder and Lapata (2005) focus on the **discourse chunking** problem, comprising two subtasks: discourse segmentation and (flat) nuclearity assignment. They formulate discourse chunking in two alternative ways. First, **one-step classification**, where the discourse chunker, a multi-class classifier, assigns to each token one of the four labels: (1) B–NUC (beginning of a nucleus), (2) I–NUC (inside a nucleus), (3) B–SAT (beginning of a satellite), and (4) I–SAT (inside a satellite). Therefore, this approach performs discourse segmentation and nuclearity assignment simultaneously. Second,

2 http://www.isi.edu/licensed-sw/spade/.

two-step classification, where in the first step, the discourse segmenter (a binary classifier) labels each token as either B (beginning of an EDU) or I (inside an EDU). Then, in the second step, a nuclearity labeler (another binary classifier) assigns a nuclearity status to each segment. The two-step approach avoids illegal chunk sequences like a B–NUC followed by an I–SAT or a B–SAT followed by an I–NUC, and in this approach, it is easier to incorporate sentence-level properties like the constraint that a sentence must contain at least one nucleus. They examine whether shallow-syntactic features (e.g., POS and phrase tags) would be sufficient for these purposes. The evaluation on the RST–DT shows that the two-step approach outperforms the one-step approach, and its performance is comparable to that of SPADE, which requires relatively expensive full syntactic parses.

In follow–up work, Fisher and Roark (2007) demonstrate over 4% absolute performance gain in discourse segmentation, by combining the features extracted from the syntactic tree with the ones derived via POS tagging and shallow syntactic parsing (i.e., chunking). Using quite a large number of features in a binary log-linear model, they achieve state-of-the-art performance in discourse segmentation on the RST–DT test set.

In a different approach, Regneri, Egg, and Koller (2008) propose to use **Underspecified Discourse Representation (UDR)** as an intermediate representation for discourse parsing. Underspecified representations offer a single compact representation to express possible ambiguities in a linguistic structure, and have been primarily used to deal with scope ambiguity in semantic structures (Reyle 1993; Egg, Koller, and Niehren 2001; Althaus et al. 2003; Koller, Regneri, and Thater 2008). Assuming that a UDR of a DT is already given in the form of a dominance graph (Althaus et al. 2003), Regneri, Egg, and Koller (2008) convert it into a more expressive and complete UDR representation called **regular tree grammar** (Koller, Regneri, and Thater 2008), for which efficient algorithms (Knight and Graehl 2005) already exist to derive the best configuration (i.e., the best discourse tree).

Hernault et al. (2010) present the publicly available **HILDA** system,[3] which comes with a discourse segmenter and a parser based on Support Vector Machines (SVMs). The discourse segmenter is a binary SVM classifier that uses the same lexico-syntactic features used in SPADE, but with more context (i.e., the lexico-syntactic features for the previous two words and the following two words). The discourse parser iteratively uses two SVM classifiers in a pipeline to build a DT. In each iteration, a binary classifier first decides which of the adjacent units to merge, then a multi-class classifier connects the selected units with an appropriate relation label. Using this simple method, they report promising results in document-level discourse parsing on the RST–DT.

For a different genre, *instructional* texts, Subba and Di-Eugenio (2009) propose a shift–reduce discourse parser that relies on a classifier for relation labeling. Their classifier uses **Inductive Logic Programming (ILP)** to learn first-order logic rules from a large set of features including the linguistically rich *compositional semantics* coming from a semantic parser. They demonstrate that including compositional semantics with other features improves the performance of the classifier, thus, also improves the performance of the parser.

Both HILDA and the ILP-based approach of Subba and Di-Eugenio (2009) are limited in several ways. First, they do not differentiate between intra- and multi-sentential

3 `http://nlp.prendingerlab.net/hilda/`.

parsing, and both scenarios use a single uniform parsing model. Second, they take a greedy (i.e., sub-optimal) approach to construct a DT. Third, they disregard sequential dependencies between DT constituents. Furthermore, HILDA considers the structure and the labels of a DT separately. Our discourse parser CODRA, as described in the next section, addresses all these limitations.

More recent work than ours also attempts to address some of the above-mentioned limitations of the existing discourse parsers. Similar to us, Feng and Hirst (2014) generate a document-level DT in two stages, where a multi-sentential parsing follows an intra-sentential one. At each stage, they iteratively use two separate linear-chain CRFs (Lafferty, McCallum, and Pereira 2001) in a cascade: one for predicting the presence of rhetorical relations between adjacent discourse units in a sequence, and the other to predict the relation label between the two most probable adjacent units to be merged as selected by the previous CRF. While they use CRFs to take into account the sequential dependencies between DT constituents, they use them greedily during parsing to achieve efficiency. They also propose a greedy *post-editing* step based on an additional feature (i.e., depth of a discourse unit) to modify the initial DT, which gives them a significant gain in performance. In a different approach, Li et al. (2014) propose a discourse-level dependency structure to capture direct relationships between EDUs rather than deep hierarchical relationships. They first create a discourse dependency treebank by converting the *deep* annotations in RST–DT to shallow *head-dependent* annotations between EDUs. To find the dependency parse (i.e., an optimal spanning tree) for a given text, they apply Eisner (1996) and Maximum Spanning Tree (McDonald et al. 2005) dependency parsing algorithms with the Margin Infused Relaxed Algorithm online learning framework (McDonald, Crammer, and Pereira 2005).

With the successful application of **deep learning** to numerous NLP problems including syntactic parsing (Socher et al. 2013a), sentiment analysis (Socher et al. 2013b), and various tagging tasks (Collobert et al. 2011), a couple of recent studies in discourse parsing also use deep neural networks (DNNs) and related feature representation methods. Inspired by the work of Socher et al. (2013a, 2013b), Li, Li, and Hovy (2014) propose a recursive DNN for discourse parsing. However, as in Socher et al. (2013a, 2013b), word vectors (i.e., embeddings) are not learned explicitly for the task, rather they are taken from Collobert et al. (2011). Given the vectors of the words in an EDU, their model first composes them hierarchically based on a syntactic parse tree to get the vector representation for the EDU. Adjacent discourse units are then merged hierarchically to get the vector representations for the higher order discourse units. In every step, the merging is done using one binary (structure) and one multi-class (relation) classifier, each having a three-layer neural network architecture. The cost function for training the model is given by these two cascaded classifiers applied at different levels of the DT. Similar to our method, they use the classifier probabilities in a CKY-like parsing algorithm to find the global optimal DT. Finally, Ji and Eisenstein (2014) present a feature representation learning method in a shift–reduce discourse parser (Marcu 1999). Unlike DNNs, which learn non-linear feature transformations in a maximum likelihood model, they learn linear transformations of features in a max margin classification model.

3. Overview of Our Rhetorical Analysis Framework

CODRA takes as input a raw text and produces a discourse tree that describes the text in terms of coherence relations that hold between adjacent discourse units (i.e., clauses, sentences) in the text. An example DT generated by an online demo of CODRA

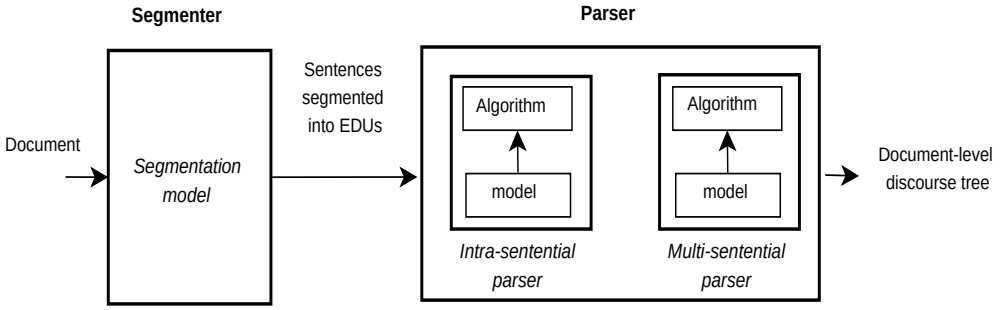


Figure 2
CODRA architecture.

is shown in Appendix A.[4] The color of a node represents its nuclearity status: blue denoting nucleus and yellow denoting satellite. The demo also allows some useful user interactions—for example, collapsing or expanding a node, highlighting an EDU, and so on.[5]

CODRA follows a pipeline architecture, shown in Figure 2. Given a raw text, the first task in the rhetorical analysis pipeline is to break the text into a sequence of EDUs (i.e., discourse segmentation). Because it is taken for granted that sentence boundaries are also EDU boundaries (i.e., EDUs do not span across multiple sentences), the discourse segmentation task boils down to finding EDU boundaries inside sentences. CODRA uses a **maximum entropy** model for discourse segmentation (see Section 5).

Once the EDUs are identified, the discourse parsing problem is determining which discourse units (EDUs or larger units) to relate (i.e., the structure), and what relations (i.e., the labels) to use in the process of building the DT. Specifically, discourse parsing requires: (1) **a parsing model** to explore the search space of possible structures and labels for their nodes, and (2) **a parsing algorithm** for selecting the best parse tree(s) among the candidates. A probabilistic parsing model like ours assigns a probability to every possible DT. The parsing algorithm then picks the most probable DTs.

The existing discourse parsers (Marcu 1999; Soricut and Marcu 2003; Subba and Di-Eugenio 2009; Hernault et al. 2010) described in Section 2 use parsing models that disregard the structural interdependencies between the DT constituents. However, we hypothesize that, like syntactic parsing, discourse parsing is also a structured prediction problem, which involves predicting multiple variables (i.e., the structure and the relation labels) that depend on each other (Smith 2011). Recently, Feng and Hirst (2012) also found these interdependencies to be critical for parsing performance. To capture the structural dependencies between the DT constituents, CODRA uses undirected conditional graphical models (i.e., CRFs) as its parsing models.

To find the most probable DT, unlike most previous studies (Marcu 1999; Subba and Di-Eugenio 2009; Hernault et al. 2010), which adopt a greedy solution, CODRA applies an optimal CKY parsing algorithm to the inferred posterior probabilities (obtained from the CRFs) of all possible DT constituents. Furthermore, the parsing algorithm allows CODRA to generate a list of *k*-best parse hypotheses for a given text.

4 The demo of CODRA is available at `http://109.228.0.153/Discourse_Parser_Demo/`. The source code of CODRA is available from `http://alt.qcri.org/tools/`.
5 The input text in the demo in Appendix A is taken from `www.bbc.co.uk/news/world-asia-26106490`.

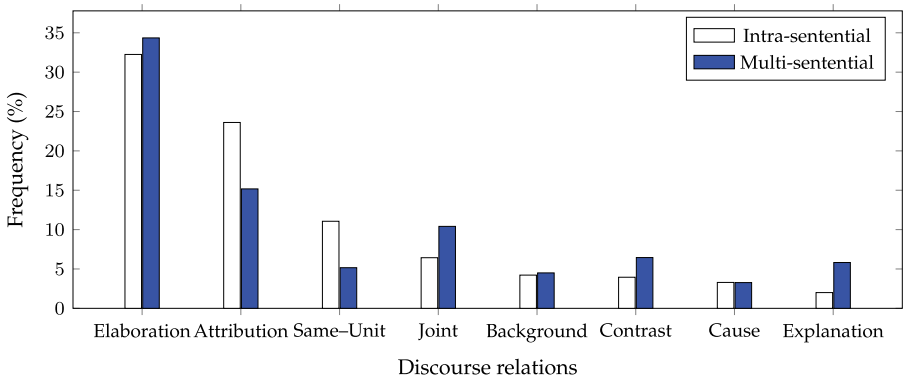


Figure 3
Distributions of the eight most frequent relations in intra-sentential and multi-sentential parsing scenarios on the RST–DT training set.

Note that the way CRFs and CKY are used in CODRA is quite different from the way they are used in syntactic parsing. For example, in the CRF-based constituency parsing proposed by Finkel, Kleeman, and Manning (2008), the conditional probability distribution of a parse tree given a sentence decomposes across factors defined over *productions*, and the standard *inside–outside* algorithm is used for inference on possible trees. In contrast, CODRA first uses the standard *forward–backward* algorithm in a "fat" chain structured[6] CRF (to be discussed in Section 4.1.1) to compute the posterior probabilities of all possible DT constituents for a given text (i.e., EDUs); then it uses a CKY parsing algorithm to combine those probabilities and find the most probable DT.

Another crucial question related to parsing models is whether to use a single model or two different models for parsing at the sentence-level (i.e., intra-sentential) and at the document-level (i.e., multi-sentential). A simple and straightforward strategy would be to use a single unified parsing model for both intra- and multi-sentential parsing without distinguishing the two cases, as was previously done (Marcu 1999; Subba and Di-Eugenio 2009; Hernault et al. 2010). That approach has the advantages of making the parsing process easier, and the model gets more data to learn from. However, for a solution like ours, which tries to capture the interdependencies between constituents, this would be problematic with respect to scalability and inappropriate because of two modeling issues.

More specifically, for scalability note that the number of valid trees grows exponentially with the number of EDUs in a document.[7] Therefore, an exhaustive search over all the valid DTs is often infeasible, even for relatively small documents.

For modeling, a single unified approach is inappropriate for two reasons. On the one hand, it appears that discourse relations are distributed differently intra- versus multi-sententially. For example, Figure 3 shows a comparison between the two distributions of the eight most frequent relations in the RST–DT training set. Notice that *Same–Unit* is more frequent than *Joint* in the intra-sentential case, whereas *Joint* is more frequent than *Same–Unit* in the multi-sentential case. Similarly, the relative distributions

6 By the term "fat" we refer to CRFs with multiple (interconnected) chains of output variables.
7 For $n + 1$ EDUs, the number of valid discourse tree structures (i.e., not counting possible variations in the nuclearity and relation labels) is the *Catalan number* C_n.

of *Background*, *Contrast*, *Cause*, and *Explanation* are different in the two parsing scenarios. On the other hand, different kinds of features are applicable and informative for intra- versus multi-sentential parsing. For example, syntactic features like **dominance sets** (Soricut and Marcu 2003) are extremely useful for parsing at the sentence-level, but are not even applicable in the multi-sentential case. Likewise, **lexical chain features** (Sporleder and Lapata 2004), which are useful for multi-sentential parsing, are not applicable at the sentence level.

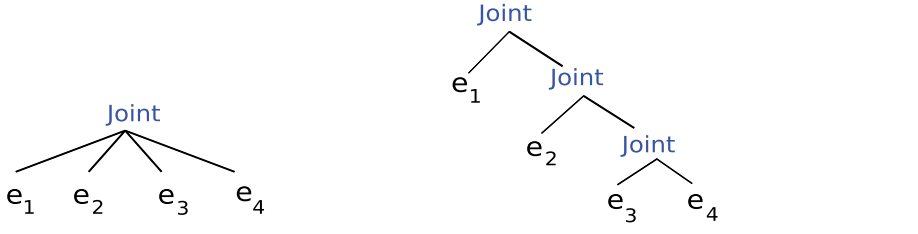
Based on these above observations, CODRA comprises two separate modules: an **intra-sentential parser** and a **multi-sentential parser**, as shown in Figure 2. First, the intra-sentential parser produces one or more discourse sub-trees for each sentence. Then, the multi-sentential parser generates a full DT for the document from these sub-trees. Both of our parsers have the same two components: a *parsing model* and a *parsing algorithm*. Whereas the two parsing models are rather different, the same parsing algorithm is shared by the two modules. Staging multi-sentential parsing on top of intra-sentential parsing in this way allows CODRA to explicitly exploit the strong correlation observed between the text structure and the DT structure, as explained in detail in Section 4.3.

4. The Discourse Parser

Before describing the parsing models and the parsing algorithm of CODRA in detail, we introduce some terminology that we will use throughout this article.

A DT can be formally represented as a set of constituents of the form $R[i, m, j]$, where $i \leq m < j$. This refers to a rhetorical relation R between the discourse unit containing EDUs i through m and the discourse unit containing EDUs $m{+}1$ through j. For example, the DT for the second sentence in Figure 1 can be represented as {*Elaboration–NS[4,4,5]*, *Same–Unit–NN[4,5,6]*}. Notice that in this representation, a relation R also specifies the nuclearity status of the discourse units involved, which can be one of *Nucleus–Satellite (NS)*, *Satellite–Nucleus (SN)*, *or Nucleus–Nucleus (NN)*. Attaching nuclearity status to the relations allows us to perform the two subtasks of discourse parsing, *relation identification* and *nuclearity assignment*, simultaneously.

A common assumption made for generating DTs effectively is that they are *binary* trees (Soricut and Marcu 2003; Hernault et al. 2010). That is, multi-nuclear relations (e.g., *Joint*, *Same–Unit*) involving more than two discourse units are mapped to a hierarchical right-branching binary tree. For example, a flat $Joint(e_1, e_2, e_3, e_4)$ (Figure 4a) is mapped to a right-branching binary tree $Joint(e_1, Joint(e_2, Joint(e_3, e_4)))$ (Figure 4b).



(a) A multi-nuclear relation. (b) The corresponding binary tree.

Figure 4
Multi-nuclear relation and its corresponding binary tree representation.

4.1 Parsing Models

As mentioned before, the job of the intra- and multi-sentential parsing models of CODRA is to assign a probability to each of the constituents of all possible DTs at the sentence level and at the document level, respectively. Formally, given the model parameters Θ at a particular parsing scenario (i.e., sentence-level or document-level), for each possible constituent $R[i, m, j]$ in a candidate DT at that parsing scenario, the parsing model estimates $P(R[i, m, j]|\Theta)$, which specifies a joint distribution over the label R and the structure $[i, m, j]$ of the constituent. For example, when applied to the sentences in Figure 1 separately, the intra-sentential parsing model (with learned parameters Θ_s) estimates $P(R[1, 1, 2]|\Theta_s)$, $P(R[2, 2, 3]|\Theta_s)$, $P(R[1, 2, 3]|\Theta_s)$, and $P(R[1, 1, 3]|\Theta_s)$ for the first sentence, and $P(R[4, 4, 5]|\Theta_s)$, $P(R[5, 5, 6]|\Theta_s)$, $P(R[4, 5, 6]|\Theta_s)$, and $P(R[4, 4, 6]|\Theta_s)$ for the second sentence, respectively, for all R ranging over the set of relations.

4.1.1 Intra-Sentential Parsing Model. Figure 5 shows the parsing model of CODRA for intra-sentential parsing. The observed nodes U_j (at the bottom) in a sequence represent the discourse units (EDUs or larger units). The first layer of hidden nodes are the structure nodes, where $S_j \in \{0, 1\}$ denotes whether two adjacent discourse units U_{j-1} and U_j should be connected or not. The second layer of hidden nodes are the relation nodes, with $R_j \in \{1 \dots M\}$ denoting the relation between two adjacent units U_{j-1} and U_j, where M is the total number of relations in the relation set. The connections between adjacent nodes in a hidden layer encode sequential dependencies between the respective hidden nodes, and can enforce constraints such as the fact that a node must have a unique mother, namely, a $S_j = 1$ must not follow a $S_{j-1} = 1$. The connections between the two hidden layers model the structure and the relation of DT constituents jointly.

Notice that the probabilistic graphical model shown in Figure 5 is a chain-structured undirected graphical model (also known as **Markov Random Field** or **MRF** [Murphy 2012]) with two hidden layers, i.e., structure chain and relation chain. It becomes a **Dynamic Conditional Random Field (DCRF)** (Sutton, McCallum, and Rohanimanesh 2007) when we directly model the hidden (output) variables by conditioning the clique potentials (i.e., factors) on the observed (input) variables:

$$P(R_{2:t}, S_{2:t}|\mathrm{x}, \Theta_s) = \frac{1}{Z(\mathrm{x}, \Theta_s)} \prod_{i=2}^{t-1} \phi(R_i, R_{i+1}|\mathrm{x}, \Theta_{s,r}) \psi(S_i, S_{i+1}|\mathrm{x}, \Theta_{s,s}) \omega(R_i, S_i|\mathrm{x}, \Theta_{s,c}) \quad (1)$$

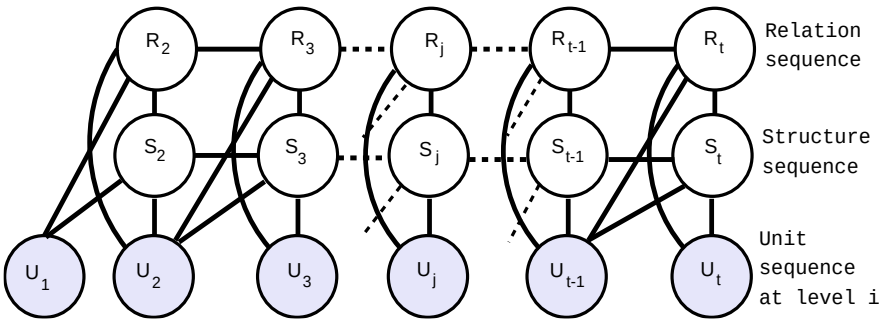


Figure 5
The intra-sentential parsing model of CODRA.

where $\{\phi\}$ and $\{\psi\}$ are the factors over the edges of the relation and structure chains, respectively, and $\{\omega\}$ are the factors over the edges connecting the relation and structure nodes (i.e., between-chain edges). Here, x represents input features extracted from the observed variables, $\Theta_s = [\Theta_{s,r}, \Theta_{s,s}, \Theta_{s,c}]$ are model parameters, and $Z(\mathrm{x}, \Theta_s)$ is the partition function. We use the standard log-linear representation of the factors:

$$\phi(R_i, R_{i+1}|\mathrm{x}, \Theta_{s,r}) = \exp(\Theta_{s,r}^T f(R_i, R_{i+1}, \mathrm{x})) \quad (2)$$

$$\psi(S_i, S_{i+1}|\mathrm{x}, \Theta_{s,s}) = \exp(\Theta_{s,s}^T f(S_i, S_{i+1}, \mathrm{x})) \quad (3)$$

$$\omega(R_i, S_i|\mathrm{x}, \Theta_{s,c}) = \exp(\Theta_{s,c}^T f(R_i, S_i, \mathrm{x})) \quad (4)$$

where $f(Y, Z, \mathrm{x})$ is a feature vector derived from the input features x and the local labels Y and Z, and $\Theta_{s,y}$ is the corresponding weight vector—that is, $\Theta_{s,r}$ and $\Theta_{s,s}$ are the weight vectors for the factors over the relation edges and the structure edges, respectively, and $\Theta_{s,c}$ is the weight vector for the factors over the between-chain edges.

A DCRF is a generalization of linear-chain CRFs (Lafferty, McCallum, and Pereira 2001) to represent complex interactions between output variables (i.e., labels), such as when performing multiple labeling tasks on the same sequence. Recently, there has been an explosion of interest in CRFs for solving structured output classification problems, with many successful applications in NLP including syntactic parsing (Finkel, Kleeman, and Manning 2008), syntactic chunking (Sha and Pereira 2003), and discourse chunking (Ghosh et al. 2011) in accordance with the Penn Discourse Treebank (Prasad et al. 2008).

DCRFs, being a discriminative approach to sequence modeling, have several advantages over their generative counterparts such as **Hidden Markov Models (HMMs)** and MRFs, which first model the joint distribution $p(\mathrm{y}, \mathrm{x}|\Theta)$, and then infer the conditional distribution $p(\mathrm{y}|\mathrm{x}, \Theta)$. It has been advocated that discriminative models are generally more accurate than generative ones because they do not "waste resources" modeling complex distributions that are observed (i.e., $p(\mathrm{x})$); instead, they focus directly on modeling what we care about, namely, the distribution of labels given the data (Murphy 2012).

Other key advantages include the ability to incorporate arbitrary overlapping local and global features, and the ability to relax strong independence assumptions. Furthermore, CRFs surmount the *label bias* problem (Lafferty, McCallum, and Pereira 2001) of the **Maximum Entropy Markov Model** (McCallum, Freitag, and Pereira 2000), which is considered to be a discriminative version of the HMM.

4.1.2 Training and Applying the Intra-Sentential Parsing Model. In order to obtain the probability of the constituents of all candidate DTs for a sentence, CODRA applies the intra-sentential parsing model (with learned parameters Θ_s) recursively to sequences at different levels of the DT, and computes the posterior marginals over the relation–structure pairs. It uses the standard *forward–backward* algorithm to compute the posterior marginals. To illustrate the process, let us assume that the sentence contains four EDUs, $e_1, \cdots, e_4$ (see Figure 6). At the first (i.e., bottom) level of the DT, when all the discourse units are EDUs, there is only one unit sequence (e_1, e_2, e_3, e_4) to which CODRA applies the DCRF model. Figure 6a at the top left shows the corresponding DCRF model. For this sequence it computes the posterior marginals $P(R_2, S_2{=}1|e_1, e_2, e_3, e_4, \Theta_s)$,

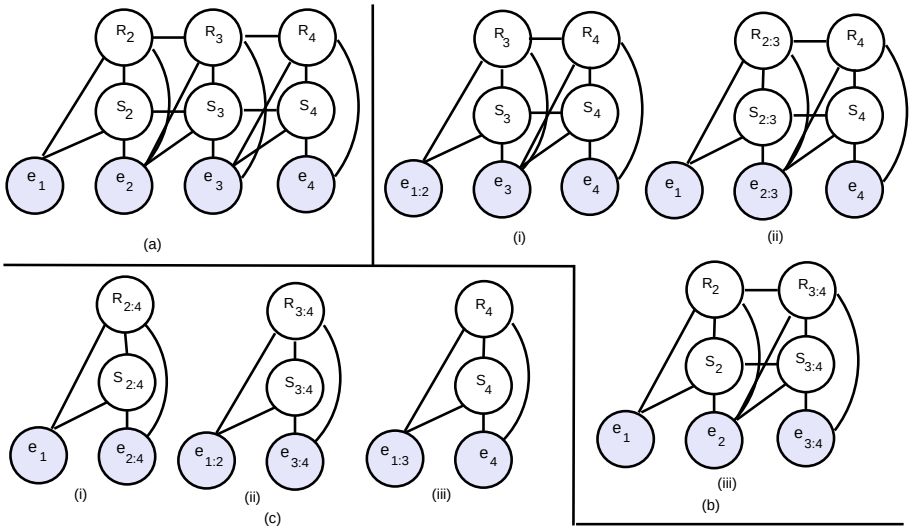


Figure 6
The intra-sentential parsing model is applied to (a) the only possible sequence at the first level, (b) the three possible sequences at the second level, and (c) the three possible sequences at the third level.

$P(R_3, S_3{=}1|e_1, e_2, e_3, e_4, \Theta_s)$, and $P(R_4, S_4{=}1|\, e_1, e_2, e_3, e_4, \Theta_s)$ to obtain the probability of the DT constituents $R[1, 1, 2]$, $R[2, 2, 3]$, and $R[3, 3, 4]$, respectively.

At the second level, there are three unit sequences: $(e_{1:2}, e_3, e_4)$, $(e_1, e_{2:3}, e_4)$, and $(e_1, e_2, e_{3:4})$. Figure 6b shows their corresponding DCRF models. Notice that each of these sequences has a discourse unit that connects two EDUs, and the probability of this connection has already been computed at the previous level. CODRA computes the posterior marginals $P(R_3, S_3{=}1|e_{1:2}, e_3, e_4, \Theta_s)$, $P(R_{2:3}S_{2:3}{=}1|e_1, e_{2:3}, e_4, \Theta_s)$, $P(R_4, S_4{=}1|e_1, e_{2:3}, e_4, \Theta_s)$, and $P(R_{3:4}, S_{3:4}{=}1|e_1, e_2, e_{3:4}, \Theta_s)$ from these three sequences, which correspond to the probability of the constituents $R[1, 2, 3]$, $R[1, 1, 3]$, $R[2, 3, 4]$, and $R[2, 2, 4]$, respectively. Similarly, it attains the probability of the constituents $R[1, 1, 4]$, $R[1, 2, 4]$, and $R[1, 3, 4]$ by computing their respective posterior marginals from the three sequences at the third (i.e., top) level of the candidate DTs (see Figure 6c).

Algorithm 1 describes how CODRA generates the unit sequences at different levels of the candidate DTs for a given number of EDUs in a sentence. Specifically, to compute the probability of a DT constituent $R[i, k, j]$, CODRA generates sequences like $(e_1, \cdots, e_{i-1}, e_{i:k}, e_{k+1:j}, e_{j+1}, \cdots, e_n)$ for $1 \leq i \leq k < j \leq n$. However, in doing so, it may generate some duplicate sequences. Clearly, the sequence $(e_1, \cdots, e_{i-1}, e_{i:i}, e_{i+1:j}, e_{j+1}, \cdots, e_n)$ for $1 \leq i \leq k < j < n$ is already considered for computing the probability of the constituent $R[i + 1, j, j + 1]$. Therefore, it is a duplicate sequence that CODRA excludes from the list of sequences. The algorithm has a complexity of $O(n^3)$, where n is the number of EDUs in the sentence.

Once CODRA acquires the probability of all possible intra-sentential DT constituents, the discourse sub-trees for the sentences are built by applying an optimal parsing algorithm (Section 4.2) using one of the methods described in Section 4.3.

Algorithm 1 is also used to generate sequences for training the model (i.e., learning Θ_s). For example, Figure 7 demonstrates how we generate the training instances (right) from a gold DT with four EDUs (left). To find the relevant labels for the sequences

Algorithm 1 Generating unit sequences for a sentence with n EDUs.

Input: Sequence of EDUs: $(e_1, e_2, \cdots, e_n)$
Output: List of sequences: L

```
for i = 1 → n − 1 do                          // all possible starting positions for the subsequence
    for j = i + 1 → n do                      // all possible ending positions for the subsequence
        if j == n then                        // sequences at top and bottom levels
            for k = i → j − 1 do              // all possible cut points within the subsequence
                L.append ((e_1, ··· , e_{i−1}, e_{i:k}, e_{k+1:j}, e_{j+1}, ··· , e_n))
            end
        else                                  // sequences at intermediate levels
            for k = i + 1 → j − 1 do          // cut points excluding duplicate sequences
                L.append ((e_1, ··· , e_{i−1}, e_{i:k}, e_{k+1:j}, e_{j+1}, ··· , e_n))
            end
        end
    end
end
```

generated by the algorithm, we consult the gold DT and see if two discourse units are connected by a relation r (i.e., the corresponding labels are $S = 1, R = r$) or not (i.e., the corresponding labels are $S = 0, R =$NR). We train the model by maximizing the *conditional likelihood* of the labels in each of these training examples (see Equation (1)).

4.1.3 Multi-Sentential Parsing Model. Given the discourse units (sub-trees) for all the individual sentences in a document, a simple approach to build the DT of the document would be to apply a new DCRF model, similar to the one in Figure 5 (with different parameters), to all the possible sequences generated from these units by Algorithm 1 to infer the probability of all possible higher-order (multi-sentential) constituents. However, the number of possible sequences and their length increase with the number of sentences in a document. For example, assuming that each sentence has a well-formed DT, for a document with n sentences, Algorithm 1 generates $O(n^3)$ sequences, where

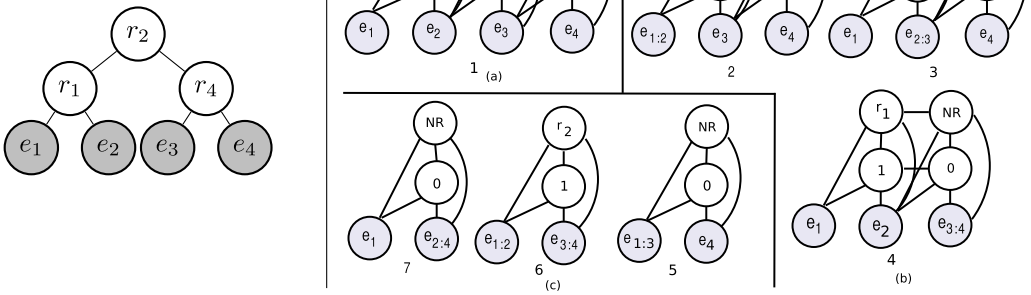


Figure 7
A gold discourse tree (left), and the 7 training instances it generates (right). NR = No Relation.

the sequence at the bottom level has n units, each of the sequences at the second level has n-1 units, and so on. Because the DCRF model in Figure 5 has a "fat" chain structure, one could use the forward–backward algorithm for exact inference in this model (Murphy 2012). Forward–backward on a sequence containing T units costs $O(TM^2)$ time, where M is the number of relations in our relation set. This makes the chain-structured DCRF model impractical for multi-sentential parsing of long documents, since learning requires running inference on every training sequence with an overall time complexity of $O(TM^2n^3) = O(M^2n^4)$ per document (Sutton and McCallum 2012).

To address this problem, we have developed a simplified parsing model for multi-sentential parsing. Our model is shown in Figure 8. The two observed nodes U_{t-1} and U_t are two adjacent (multi-sentential) discourse units. The (hidden) structure node $S \in \{0, 1\}$ denotes whether the two discourse units should be linked or not. The other hidden node $R \in \{1 \dots M\}$ represents the relation between the two units. Notice that similar to the model in Figure 5, this is also an undirected graphical model and becomes a CRF model if we directly model the labels by conditioning the clique potential ϕ on the input features x, derived from the observed variables:

$$P(R_t, S_t|\mathrm{x}, \Theta_d) = \frac{1}{Z(\mathrm{x}, \Theta_d)} \phi(R_t, S_t|\mathrm{x}, \Theta_d) \tag{5}$$

$$\phi(R_t, S_t|\mathrm{x}, \Theta_d) = \exp(\Theta_d^T f(R_t, S_t, \mathrm{x})) \tag{6}$$

where $f(R_t, S_t, \mathrm{x})$ is a feature vector derived from the input features x and the labels R_t and S_t, and Θ_d is the corresponding weight vector. Although this model is similar in spirit to the parsing model in Figure 5, it now breaks the chain structure, which makes the inference much faster (i.e., a complexity of $O(M^2)$). Breaking the chain structure also allows CODRA to balance the data for training (an equal number of instances with S=1 and S=0), which dramatically reduces the learning time of the model.

CODRA applies this parsing model to all possible adjacent units at all levels in the multi-sentential case, and computes the posterior marginals of the relation–structure pairs $P(R_t, S_t{=}1|U_{t-1}, U_t, \Theta_d)$ using the forward–backward algorithm to obtain the probability of all possible DT constituents. Given the sentence-level discourse units, Algorithm 2, which is a simplified variation of Algorithm 1, extracts all possible adjacent discourse units for multi-sentential parsing. Similar to Algorithm 1, Algorithm 2 also has a complexity of $O(n^3)$, where n is the number of sentence-level discourse units.

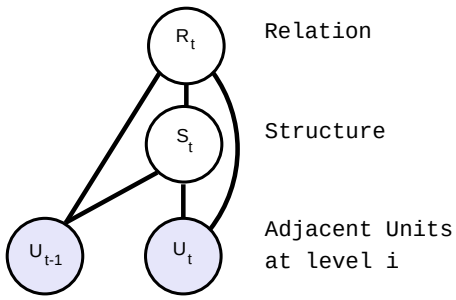


Figure 8
The multi-sentential parsing model of CODRA.

Algorithm 2 Generating all possible adjacent discourse units at all levels of a document-level discourse tree.

Input: Sequence of units: $(U_1, U_2, \cdots U_n)$, where $U_x[0]$:= start EDU ID of unit x, and $U_x[1]$:= end EDU ID of unit x.
Output: List of adjacent units: L

```
for i = 1 → n − 1 do                          // all possible starting positions for the subsequence
    for j = i + 1 → n do                      // all possible ending positions for the subsequence
        for k = i → j − 1 do                  // all possible cut points within the subsequence
            Left = U_i[0] : U_k[1]
            Right = U_{k+1}[0] : U_j[1]
            L.append ((Left, Right))
        end
    end
end
```

Both our intra- and multi-sentential parsing models are designed using MALLET's graphical model toolkit GRMM (McCallum 2002). In order to avoid overfitting, we regularize the CRF models with l_2 regularization and learn the model parameters using the limited-memory BFGS (L-BFGS) fitting algorithm.

4.1.4 Features Used in the Parsing Models. Crucial to parsing performance is the set of features used in the parsing models, as summarized in Table 1. We categorize the features into seven groups and specify which groups are used in what parsing model. Notice that some of the features are used in both models. Most of the features have been explored in previous studies (e.g., Soricut and Marcu 2003; Sporleder and Lapata 2005; Hernault et al. 2010). However, we improve some of these as explained subsequently.

The features are extracted from two adjacent discourse units U_{t-1} and U_t. **Organizational** features encode useful information about text organization as shown by duVerle and Prendinger (2009). We measure the length of the discourse units as the number of *EDUs* and *tokens* in it. However, in order to better adjust to the length variations, rather than computing their absolute numbers in a unit, we choose to measure their *relative numbers* with respect to their total numbers in the two units. For example, if the two discourse units under consideration contain three EDUs in total, a unit containing two of the EDUs will have a relative EDU number of 0.67. We also measure the *distances* of the units in terms of the number of EDUs from the beginning and end of the sentence (or text in the multi-sentential case). **Text structural** features capture the correlation between text structure and rhetorical structure by counting the number of *sentence* and *paragraph* boundaries in the discourse units.

Discourse cues (e.g., *because, but*), when present, signal rhetorical relations between two text segments, and have been used as a primary source of information in earlier studies (Knott and Dale 1994; Marcu 2000a). However, recent studies (Hernault et al. 2010; Biran and Rambow 2011) suggest that an empirically acquired *lexical* **N-gram** dictionary is more effective than a fixed list of cue phrases, since this approach is domain independent and capable of capturing non-lexical cues such as punctuation.

In order to build a lexical N-gram dictionary empirically from the training corpus, we extract the first and last N tokens ($N \in \{1, 2, 3\}$) of each discourse unit and rank them according to their *mutual information* with the two labels, *Structure (S)* and *Relation (R)*.

Table 1
Features used in our intra- and multi-sentential parsing models.

8 Organizational features	***Intra & Multi-Sentential***
Number of EDUs in *unit 1* (or *unit 2*).	
Number of tokens in *unit 1* (or *unit 2*).	
Distance of unit 1 in EDUs to the *beginning* (or to the *end*).	
Distance of unit 2 in EDUs to the *beginning* (or to the *end*).	
4 Text structural features	***Multi-Sentential***
Number of sentences in *unit 1* (or *unit 2*).	
Number of paragraphs in *unit 1* (or *unit 2*).	
8 N-gram features $N \in \{1, 2, 3\}$	***Intra & Multi-Sentential***
Beginning (or *end*) lexical N-grams in unit 1.	
Beginning (or *end*) lexical N-grams in unit 2.	
Beginning (or *end*) POS N-grams in unit 1.	
Beginning (or *end*) POS N-grams in unit 2.	
5 Dominance set features	***Intra-Sentential***
Syntactic labels of the *head* node and the *attachment* node.	
Lexical heads of the *head* node and the *attachment* node.	
Dominance relationship between the two units.	
9 Lexical chain features	***Multi-Sentential***
Number of chains spanning unit 1 and unit 2.	
Number of chains start in unit 1 and end in unit 2.	
Number of chains *start* (or *end*) in *unit 1* (or in *unit 2*).	
Number of chains skipping both unit 1 and unit 2.	
Number of chains skipping *unit 1* (or *unit 2*).	
2 Contextual features	***Intra & Multi-Sentential***
Previous and *next* feature vectors.	
2 Sub-structural features	***Intra & Multi-Sentential***
Root nodes of the *left* and *right* rhetorical sub-trees.	

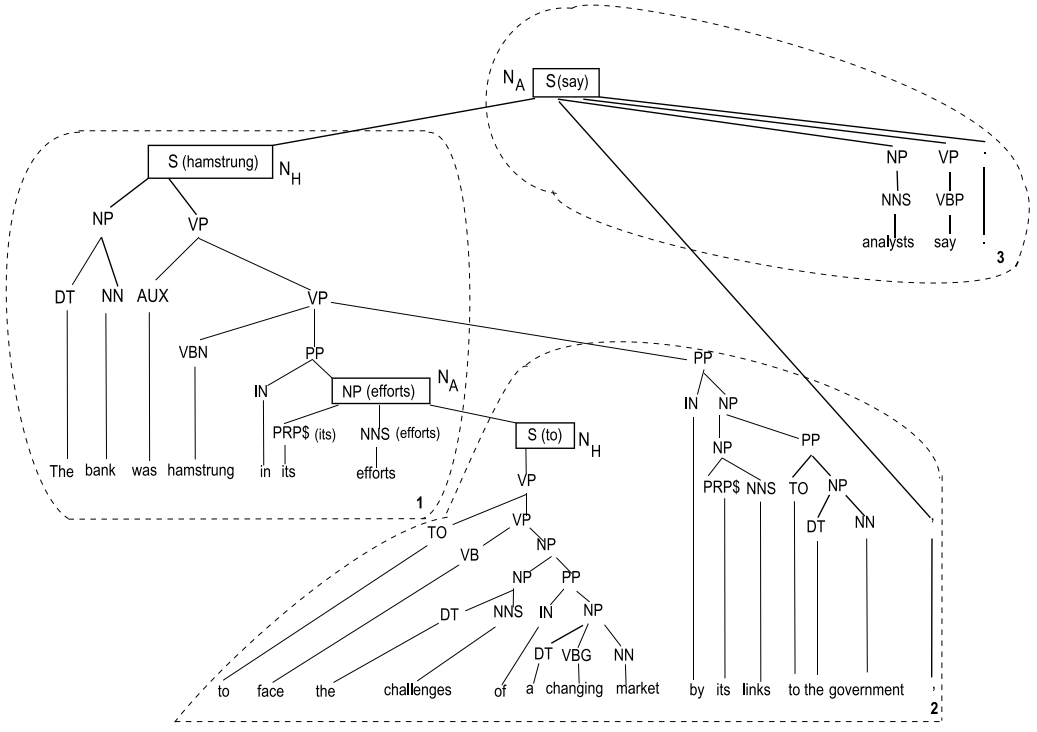
More specifically, given an N-gram x, we compute its *conditional entropy* H with respect to S and R as follows:[8]

$$H(S, R|x) = -\sum_{s \in S} \sum_{r \in R} \log \frac{c(x, s, r)}{c(x)} \tag{7}$$

where $c(x)$ is the empirical count of N-gram x, and $c(x, s, r)$ is the joint empirical count of N-gram x with the labels s and r. This is in contrast to HILDA (Hernault et al. 2010), which ranks the N-grams by their frequencies in the training corpus. However, Blitzer

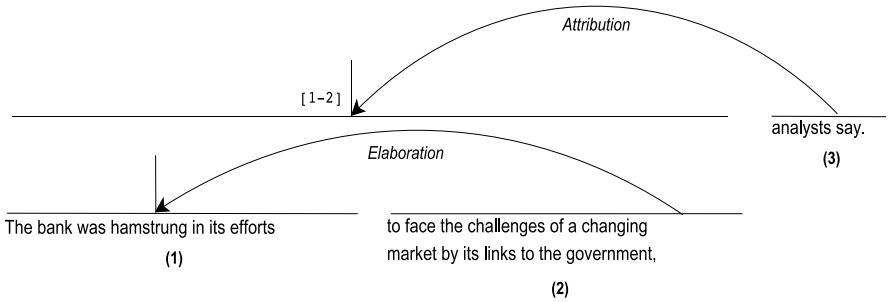
8 The higher the conditional entropy, the lower the mutual information, and vice versa.

(2008) found mutual information to be more effective than frequency as a method for feature selection. Intuitively, the most informative discourse cues are not only the most frequent, but also the ones that are indicative of the labels in the training data. In addition to the lexical N-grams we also encode the *POS* tags of the first and last N tokens ($N \in \{1,2,3\}$) in a discourse unit as shallow-syntactic features in our models.



D = { *((1, efforts/NP) > (2, to/S)), ((3, say/S) > (1, hamstrung/S))* }

(a) The discourse segmented lexicalized syntactic tree (DS-LST) for a sentence in RST–DT. Boxed nodes form the dominance set D as shown at the bottom.



(b) Discourse tree for the given sentence.

Figure 9
Dominance set features for intra-sentential discourse parsing.

Lexico-syntactic features **dominance sets** extracted from the Discourse Segmented Lexicalized Syntactic Tree (DS-LST) of a sentence have been shown to be extremely effective for intra-sentential discourse parsing in SPADE (Soricut and Marcu 2003). Figure 9a shows the DS-LST (i.e., lexicalized syntactic tree with EDUs identified) for a sentence with three EDUs from the RST–DT corpus, and Figure 9b shows the corresponding discourse tree. In a DS-LST, each EDU except the one with the root node must have a *head node* N_H that is attached to an *attachment node* N_A residing in a separate EDU. A dominance set D (shown at the bottom of Figure 9a) contains these *attachment* points (shown in boxes) of the EDUs in a DS-LST. In addition to the syntactic and lexical information of the head and attachment nodes, each element in the dominance set also includes a dominance relationship between the EDUs involved; the EDU with the attachment node dominates (represented by ">") the EDU with the head node.

Soricut and Marcu (2003) hypothesize that the dominance set (i.e., lexical heads, syntactic labels, and dominance relationships) carries the most informative clues for intra-sentential parsing. For instance, the dominance relationship between the EDUs in our example sentence is $3 > 1 > 2$, which favors the DT structure $[1, 1, 2]$ over $[2, 2, 3]$. In order to extract dominance set features for two adjacent discourse units U_{t-1} and U_t, containing EDUs $e_{i:j}$ and $e_{j+1:k}$, respectively, we first compute the dominance set from the DS-LST of the sentence. We then extract the element from the set that holds across the EDUs j and $j+1$. In our example, for the two units, containing EDUs e_1 and e_2, respectively, the relevant dominance set element is *(1, efforts/NP)>(2, to/S)*. We encode the syntactic labels and lexical heads of N_H and N_A, and the dominance relationship as features in our intra-sentential parsing model.

Lexical chains (Morris and Hirst 1991) are sequences of semantically related words that can indicate topical boundaries in a text (Galley et al. 2003; Joty, Carenini, and Ng 2013). Features extracted from lexical chains are also shown to be useful for finding paragraph-level discourse structure (Sporleder and Lapata 2004). For example, consider the text with four paragraphs (P_1 to P_4) in Figure 10a. Now, let us assume that there is a lexical chain that spans the whole text, skipping paragraphs P_2 and P_3, while a second chain only spans P_2 and P_3. This situation makes it more likely that P_2 and P_3 should be linked in the DT before either of them is linked with another paragraph. Therefore, the DT structure in Figure 10b should be more likely than the structure in Figure 10c.

One challenge in computing lexical chains is that words can have multiple senses, and semantic relationships depend on the sense rather than the word itself. Several methods have been proposed to compute lexical chains (Barzilay and Elhadad 1997; Hirst and St. Onge 1997; Silber and McCoy 2002; Galley and McKeown 2003). We follow the state-of-the-art approach proposed by Galley and McKeown (2003), which extracts lexical chains after performing Word Sense Disambiguation (WSD).

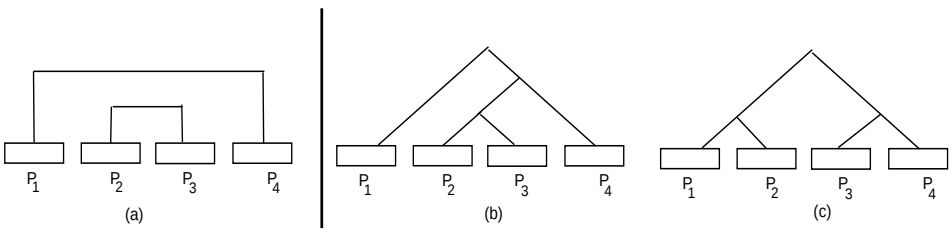


Figure 10
Correlation between lexical chains and discourse structure. (a) Lexical chains spanning paragraphs. (b) and (c) Two possible DT structures.

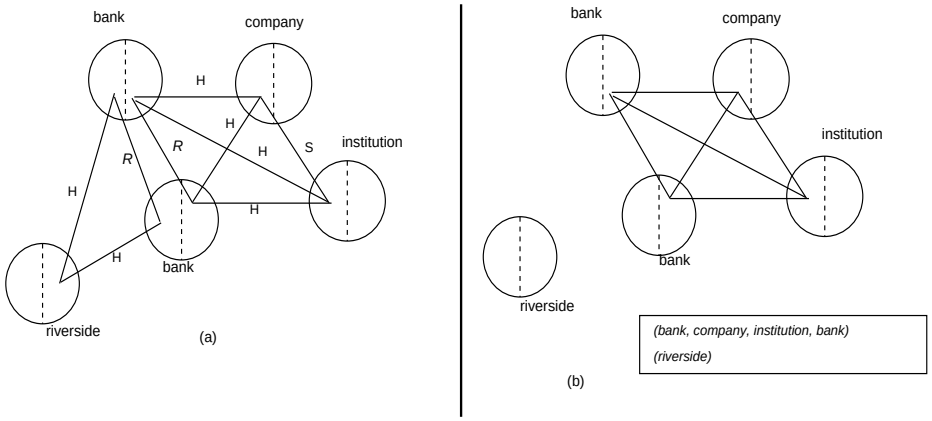


Figure 11
Extracting lexical chains. (a) A Lexical Semantic Relatedness Graph (LSRG) for five noun-tokens. (b) Resultant graph after performing WSD. The box at the bottom shows the lexical chains.

In the preprocessing step, we extract the nouns from the document and lemmatize them using WordNet's built-in *morphy* function (Fellbaum 1998). Then, by looking up in WordNet we expand each noun to all of its senses, and build a Lexical Semantic Relatedness Graph (LSRG) (Galley and McKeown 2003; Chali and Joty 2007). In an LSRG, the nodes represent noun-tokens with their candidate senses, and the weighted edges between senses of two different tokens represent one of the three semantic relations: *repetition, synonym*, and *hypernym*. For example, Figure 11a shows a partial LSRG, where the token *bank* has two possible senses, namely, *money bank* and *river bank*. Using the *money bank* sense, *bank* is connected with *institution* and *company* by hypernymy relations (edges marked with *H*), and with another *bank* by a repetition relation (edges marked with *R*). Similarly, using the *river bank* sense, it is connected with *riverside* by a hypernymy relation and with *bank* by a repetition relation. Nouns that are not found in WordNet are considered as proper nouns having only one sense, and are connected by only *repetition* relations.

We use this LSRG first to perform WSD, then to construct lexical chains. For WSD, the weights of all edges leaving the nodes under their different senses are summed up and the one with the highest score is considered to be the right sense for the word-token. For example, if repetition and synonymy are weighted equally, and hypernymy is given half as much weight as either of them, the score of *bank*'s two senses are: $1 + 0.5 + 0.5 = 2$ for the sense *money bank* and $1 + 0.5 = 1.5$ for the sense *river bank*. Therefore, the selected sense for *bank* in this context is *river bank*. In case of a tie, we select the sense that is most frequent (i.e., the first sense in WordNet). Note that this approach to WSD is different from that of Sporleder and Lapata (2004), which takes a greedy approach.

Finally, we prune the graph by only keeping the links that connect words with the selected senses. At the end of the process, we are left with the edges that form the actual lexical chains. For example, Figure 11b shows the result of pruning the graph in Figure 11a. The lexical chains extracted from the pruned graph are shown in the box at the bottom. Following Sporleder and Lapata (2004), for each chain element, we keep track of the location (i.e., sentence ID) in the text where that element was found, and exclude chains containing only one element. Given two discourse units, we count the number of chains that: hit the two units, exclusively hit the two units, skip both units, skip one of the units, start in a unit, and end in a unit.

We also consider more **contextual** information by including the above features computed for the neighboring adjacent discourse unit pairs in the current feature vector. For example, the contextual features for units U_{t-1} and U_t include the feature vector computed from U_{t-2} and U_{t-1} and the feature vector computed from U_t and U_{t+1}.

We incorporate *hierarchical dependencies* between the constituents in a DT by rhetorical **sub-structural** features. For two adjacent units U_{t-1} and U_t, we extract the roots of the two rhetorical sub-trees. For example, the root of the rhetorical sub-tree spanning over EDUs $e_{1:2}$ in Figure 9b is *Elaboration–NS*. However, extraction of these features assumes the presence of labels for the sub-trees, which is not the case when we apply the parser to a new text (sentence or document) in order to build its DT in a non-greedy fashion. One way to deal with this is to loop twice through the parsing process using two different parsing models—one trained with the complete feature set, and the other trained without the sub-structural features. We first build an initial, sub-optimal DT using the parsing model that is trained without the sub-structural features. This intermediate DT will now provide labels for the sub-structures. Next we can build a final, more accurate DT by using the complete parsing model. This idea of two-pass discourse parsing, where the second pass performs *post-editing* using additional features, has recently been adopted by Feng and Hirst (2014) in their greedy parser.

One could even continue doing post-editing multiple times until the DT converges. However, this could be very time consuming as each post-editing pass requires: (1) applying the parsing model to every possible unit sequence and computing the posterior marginals for all possible DT constituents, and (2) using the parsing algorithm to find the most probable DT. Recall from our earlier discussion in Section 4.1.3 that for n discourse units and M rhetorical relations, the first step requires $O(M^2n^4)$ and $O(M^2n^3)$ for intra- and multi-sentential parsing, respectively; we will see in the next section that the second step requires $O(Mn^3)$. In spite of the computational cost, the gain we attained in the subsequent passes was not significant for our development set. Therefore, we restrict our parser to only one-pass post-editing.

Note that in parsing models where the score (i.e., likelihood) of a parse tree decomposes across local factors (e.g., the CRF-based syntactic parser of Finkel, Kleeman, and Manning [2008]), it is possible to define a *semiring* using the factors and the local scores (e.g., given by the inside algorithm). The CKY algorithm could then give the optimal parse tree in a single post-editing pass (Smith 2011). However, because our intra-sentential parsing model is designed to capture sequential dependencies between DT constituents, the score of a DT does not directly decompose across factors over discourse productions. Therefore, designing such a semiring was not possible in our case.

In addition to these features, we also experimented with other features including *WordNet-based lexical semantics*, *subjectivity*, and *TF.IDF-based cosine similarity*. However, because such features did not improve parsing performance on our development set, they were excluded from our final set of features.

4.2 Parsing Algorithm

The intra- and multi-sentential parsing models of CODRA assign a probability to every possible DT constituent in their respective parsing scenarios. The job of the parsing algorithm is then to find the k most probable DTs for a given text. We implement a probabilistic CKY-like bottom–up parsing algorithm that uses dynamic programming to compute the most likely parses (Jurafsky and Martin 2008). For simplicity, we first

describe the specific case of generating the single most probable DT, then we describe how to generalize this algorithm to produce the k most probable DTs for a given text.

Formally, the search problem for finding the most probable DT can be written as

$$DT^* = \underset{DT}{\operatorname{argmax}}\, P(DT|\Theta) \tag{8}$$

where Θ specifies the parameters of the parsing model (intra- or multi-sentential). Given n discourse units, our parsing algorithm uses the upper-triangular portion of the $n \times n$ dynamic programming table D, where cell $D[i,j]$ (for $i < j$) stores:

$$D[i,j] = P(r^*[U_i(0), U_{m^*}(1), U_j(1)]) \tag{9}$$

where $U_x(0)$ and $U_x(1)$ are the start and end EDU Ids of discourse unit U_x, and

$$(m^*, r^*) = \underset{i \le m < j\ ;\ R \in \{1 \cdots M\}}{\operatorname{argmax}} P(R[U_i(0), U_m(1), U_j(1)]) \times D[i,m] \times D[m+1,j] \tag{10}$$

Recall that the notation $R[U_i(0), U_m(1), U_j(1)]$ in this expression refers to a rhetorical relation R that holds between the discourse unit containing EDUs $U_i(0)$ through $U_m(1)$ and the unit containing EDUs $U_m(1)+1$ through $U_j(1)$.

In addition to D, which stores the *probability* of the most probable constituents of a DT, the algorithm also simultaneously maintains two other $n \times n$ dynamic programming tables S and R for storing the structure (i.e., $U_{m^*}(1)$) and the relations (i.e., r^*) of the corresponding DT constituents, respectively. For example, given four EDUs $e_1 \cdots e_4$, the S and R dynamic programming tables at the left side in Figure 12 together represent the DT shown at the right. More specifically, to find the DT, we first look at the top-right entries in the two tables, and find $S[1,4] = 2$ and $R[1,4] = r_2$, which specify that the two discourse units $e_{1:2}$ and $e_{3:4}$ should be connected by the relation r_2 (the root in the DT). Then, we see how EDUs e_1 and e_2 should be connected by looking at the entries $S[1,2]$ and $R[1,2]$, and find $S[1,2] = 1$ and $R[1,2] = r_1$, which indicates that these two units should be connected by the relation r_1 (the left pre-terminal in the DT). Finally, to see how EDUs e_3 and e_4 should be linked, we look at the entries $S[3,4]$ and $R[3,4]$, which tell us that they should be linked by the relation r_4 (the right pre-terminal). The algorithm

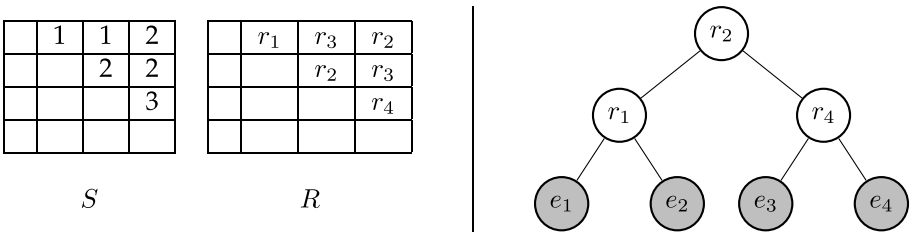


Figure 12
The S and R dynamic programming tables (left), and the corresponding discourse tree (right).

works in polynomial time. Specifically, for n discourse units and M number of relations, the time and space complexities are $O(n^3M)$ and $O(n^2)$, respectively.

A key advantage of using a probabilistic parsing algorithm like the one we use is that it allows us to generate a list of k most probable parse trees. It is straightforward to generalize the above algorithm to produce k most probable DTs. Specifically, when filling up the dynamic programming tables, rather than storing a single best parse for each sub-tree, we store and keep track (i.e., using back-pointers) of k-best candidates simultaneously. One can show that the time and space complexities of the k-best version of the algorithm are $O(n^3Mk^2 \log k)$ and $O(k^2n)$, respectively (Huang and Chiang 2005).

Note that, in contrast to other document-level discourse parsers (Marcu 2000b; Subba and Di-Eugenio 2009; Hernault et al. 2010; Feng and Hirst 2012, 2014), which use a greedy algorithm, CODRA finds a discourse tree that is globally optimal.[9] This approach of CODRA is also different from the sentence-level discourse parser SPADE (Soricut and Marcu 2003). SPADE first finds the *tree structure* that is globally optimal, then it assigns the most probable *relations* to the internal nodes. More specifically, the cell $D[i, j]$ in SPADE's dynamic programming table stores

$$D[i, j] = P([U_i(0), U_{m^*}(1), U_j(1)]) \tag{11}$$

where $m^* = \underset{i \leq m < j}{\operatorname{argmax}} P([U_i(0), U_m(1), U_j(1)])$. Disregarding the relation label R while populating D, this approach may find a discourse tree that is not globally optimal.

4.3 Document-Level Parsing Approaches

Now that we have presented our intra-sentential and multi-sentential parsing components, we are ready to describe how they can be effectively combined in a unified framework (Figure 2) to perform document-level rhetorical analysis. Recall that a key motivation for a *two-stage*[10] parsing is that it allows us to capture the strong correlation between text structure and discourse structure in a scalable, modular, and flexible way. In the following, we describe two different approaches to model this correlation.

4.3.1 1S–1S (1 Sentence–1 Sub-tree). A key finding from previous studies on sentence-level discourse analysis is that most sentences have a well-formed discourse sub-tree in the full document-level DT (Soricut and Marcu 2003; Fisher and Roark 2007). For example, Figure 13a shows 10 EDUs in three sentences (see boxes), where the DTs for the sentences obey their respective sentence boundaries.

Our first approach, called 1S–1S (1 Sentence–1 Sub-tree), aims to maximally exploit this finding. It first constructs a DT for every sentence using our intra-sentential parser, and then it provides our multi-sentential parser with the sentence-level DTs to build the rhetorical parse for the whole document.

4.3.2 Sliding Window. Although the assumption made by 1S–1S clearly simplifies the parsing process, it completely ignores the cases where rhetorical structures violate

9 We agree that with potentially sub-optimal, sub-structural features in the parsing model, CKY may end up finding a sub-optimal DT. But that is a separate issue.

10 Do not confuse the term *two-stage* with the term *two-pass*.

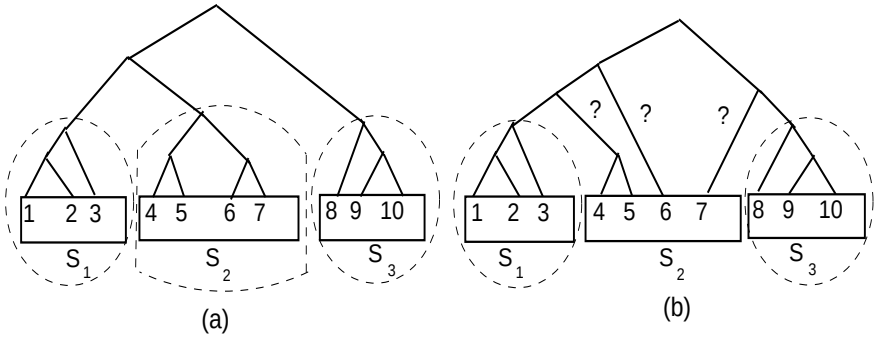


Figure 13
Two possible DTs for three sentences.

sentence boundaries. For example, in the DT shown in Figure 13b, sentence S_2 does not have a well-formed sub-tree because some of its units attach to the left (i.e., 4–5 and 6) and some to the right (i.e., 7). Vliet and Redeker (2011) call these cases "leaky" boundaries.

Although we find fewer than 5% of the sentences in the RST–DT have leaky boundaries, in other corpora this can be true for a larger portion of the sentences. For example, we observe that over 12% of the sentences in the instructional corpus of Subba and Di-Eugenio (2009) have leaky boundaries. However, we notice that in most cases where DT structures violate sentence boundaries, its units are merged with the units of its adjacent sentences, as in Figure 13b. For example, this is true for 75% of the leaky cases in our development set containing 20 news articles from the RST–DT and for 79% of the leaky cases in our development set containing 20 how-to manuals from the instructional corpus. Based on this observation, we propose a sliding window approach.

In this approach, our intra-sentential parser works with a window of two consecutive sentences, and builds a DT for the two sentences. For example, given the three sentences in Figure 13, our intra-sentential parser constructs a DT for S_1–S_2 and a DT for S_2–S_3. In this process, each sentence in a document except the boundary sentences (i.e., the first and the last) will be associated with two DTs: one with the previous sentence (say, DT_p) and one with the next (say, DT_n). In other words, for each non-boundary sentence, we will have two decisions: one from DT_p and one from DT_n. Our parser consolidates the two decisions and generates one or more sub-trees for each sentence by checking the following three mutually exclusive conditions one after another:

- *Same in both:* If the sentence under consideration has the same (in both structure and labels) well-formed sub-tree in both DT_p and DT_n, we take this sub-tree. For example, in Figure 14a, S_2 has the same sub-tree in the two DTs (one for S_1–S_2 and one for S_2–S_3). The two decisions agree on the DT for the sentence.
- *Different but no cross:* If the sentence under consideration has a well-formed sub-tree in both DT_p and DT_n, but the two sub-trees vary either in structure or in labels, we pick the most probable one. For example, consider the DT for S_1–S_2 (at the left) in Figure 14a and the DT for S_2–S_3

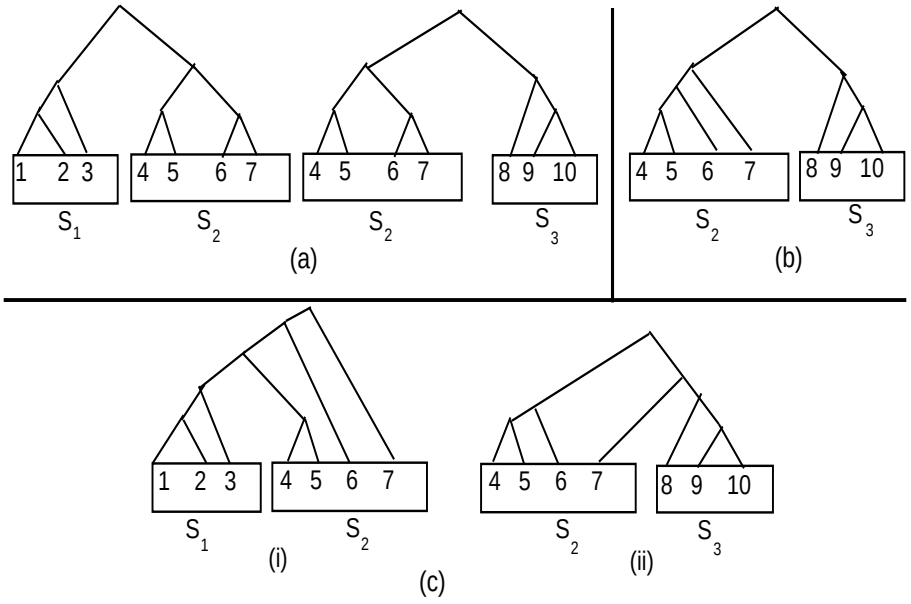


Figure 14
Extracting sub-trees for S_2.

in Figure 14b. In both cases S_2 has a well-formed sub-tree, but they differ in structure. We pick the sub-tree which has the higher probability in the two dynamic programming tables.

- *Cross:* If either or both of DT_p and DT_n segment the sentence into multiple sub-trees, we pick the one having more sub-trees. For example, consider the two DTs in Figure 14c. In the DT for S_1–S_2 on the left, S_2 has three sub-trees (4–5, 6, 7), whereas in the DT for S_2–S_3 on the right, it has two (4–6, 7). So, we extract the three sub-trees for S_2 from the first DT. If the sentence has the same number of sub-trees in both DT_p and DT_n, we pick the one with higher probability in the dynamic programming tables. Note that our choice of picking the DT with more sub-trees is intended to allow the parser to find more leaky cases. However, other heuristics are also possible. For example, another simple heuristic that one could try is: When both DTs segment the sentence into multiple sub-trees, pick the one with fewer sub-trees, and when only one of the DTs segment the sentence into multiple sub-trees, pick that one.

At the end, the multi-sentential parser takes all these sentence-level sub-trees for a document, and builds a full rhetorical parse for the whole document.

5. The Discourse Segmenter

Our discourse parser assumes that the input text has been already segmented into a sequence of EDUs. However, discourse segmentation is also a challenging problem, and previous studies (Soricut and Marcu 2003; Fisher and Roark 2007) have identified

it as a primary source of inaccuracy for discourse parsing. Regardless of its importance in discourse parsing, discourse segmentation itself can be useful in several NLP applications, including sentence compression (Sporleder and Lapata 2005) and textual alignment in statistical machine translation (Stede 2011). Therefore, in CODRA, we have developed our own discourse segmenter, which not only achieves state-of-the-art performance as shown later, but also reduces the time complexity by using fewer features.

5.1 Segmentation Model

The discourse segmenter in CODRA implements a binary classifier to decide for each word–token (except the last) in a sentence, whether to place an EDU boundary after that token. We use a **maximum entropy** model to build a discriminative classifier. More specifically, we use a Logistic Regression classifier with parameter Θ:

$$P(y|w, \Theta) = \text{Ber}(y|\, \text{Sigm}(\Theta^T \text{x})) \tag{12}$$

where the output $y \in \{0, 1\}$ denotes whether to put an EDU boundary (i.e., $y = 1$) or not (i.e., $y = 0$) after the word–token w, which is represented by a feature vector x. In the equation, Ber(η) and Sigm(η) refer to the *Bernoulli* distribution and the *Sigmoid* (also known as *logistic*) function, respectively. The negative log-likelihood (NLL) of the model with l_2 regularization for N data points (i.e., word–tokens) is given by

$$NLL(\theta) = -\sum_{i=1}^{N} y_i \log\ \text{Sigm}(\Theta^T \text{x}_\text{i}) + (1 - y_i) \log\ (1 - \text{Sigm}(\Theta^T \text{x}_\text{i})) + \lambda \Theta^T \Theta \tag{13}$$

where y_i is the gold label for word–token w_i (represented by feature vector x_i). We learn the model parameters Θ using the L-BFGS fitting algorithm, which is time- and space-efficient. To avoid overfitting, we use 5-fold cross validation to learn the regularization strength parameter λ from the training data. We also use a simple *bagging* technique (Breiman 1996) to deal with the sparsity of *boundary* (i.e., $y = 1$) tags.

Note that our first attempt at the discourse segmentation task implemented a linear-chain CRF model (Lafferty, McCallum, and Pereira 2001) to capture the sequential dependencies between the tags in a discriminative way. However, the binary Logistic Regression classifier, using the same set of features, not only outperforms the CRF model, but also reduces time and space complexity. One possible explanation for the low performance of the CRF model is that Markov dependencies between tags cannot be effectively captured due to the sparsity of boundary tags. Also, because we could not balance the data by using techniques like bagging in the CRF model, this further degrades the performance.

5.2 Features Used in the Segmentation Model

Our set of features for discourse segmentation are mostly inspired from previous studies but used in a novel way, as we describe here.

Our first subset of features, which we call **SPADE features**, includes the lexico-syntactic patterns extracted from the lexicalized syntactic tree of the given sentence.

These features replicate the features used in SPADE's segmenter, but used in a discriminative way. In order to decide on an EDU boundary after a word–token w_k, we search for the lowest constituent in the lexicalized syntactic tree that spans over tokens $w_i \dots w_j$ such that $i \leq k < j$. The production that expands this constituent in the tree, with the potential EDU boundary marked, forms the primary feature. For instance, to determine the existence of an EDU boundary after the word *efforts* in our sample sentence shown in Figure 9, the production *NP(efforts)* $\rightarrow$ *PRP$(its) NNS(efforts)* $\uparrow$ *S(to)* extracted from the lexicalized syntactic tree in Figure 9a constitutes the primary feature, where $\uparrow$ denotes the potential EDU boundary.

SPADE predicts an EDU boundary if the relative frequency (i.e., maximum likelihood estimate) of a potential boundary given the production in the training data is greater than 0.5. If the production has not been observed frequently enough, the unlexicalized version of the production (e.g., *NP* $\rightarrow$ *PRP$ NNS* $\uparrow$ *S*) is used for prediction. If the unlexicalized version is also found to be rare, other variations of the production, depending on whether they include the lexical heads and how many non-terminals (one or two) they consider before and after the potential boundary, are examined one after another (see Fisher and Roark [2007] for details). In contrast, we compute the maximum likelihood estimates for a primary production (feature) and its other variations, and use those directly as features with/without binarizing the values.

Shallow syntactic features like **Chunk** and **POS** tags have been shown to possess valuable clues for discourse segmentation (Fisher and Roark 2007; Sporleder and Lapata 2005). For example, it is less likely that an EDU boundary occurs within a chunk. We annotate the tokens of a sentence with chunk and POS tags using the state-of-the-art Illinois tagger[11] and encode these as features in our model. Note that the chunker assigns each token a tag using the *BIO* notation, where *B* stands for beginning of a particular phrase (e.g., noun or verb phrase), *I* stands for inside of a particular phrase, and *O* stands for outside of a particular phrase. The rationale for using the Illinois chunker is that it uses a larger set of tags (23 in total); thus it is more informative than most of the other existing taggers, which typically use only five tags (B–NP, I–NP, B–VP, I–VP, and O).

EDUs are normally multi-word strings. Thus, a token near the beginning or end of a sentence is unlikely to be the end of a segment. Therefore, for each token we include its **relative position** (i.e., absolute position/total number of tokens) in the sentence and **distances** to the beginning and end of the sentence as features.

It is unlikely that two consecutive tokens are tagged with EDU boundaries. Therefore, we incorporate **contextual** information for a token into our model by including the above features computed for its neighboring tokens.

We also experimented with different N-gram ($N \in \{1, 2, 3\}$) features extracted from the token sequence, POS sequence, and chunk sequence. However, because such features did not improve segmentation accuracy on the development set, they were excluded from our final set of features.

6. Experiments

In this section we present our experimental results. First, we describe the corpora on which the experiments were performed and the evaluation metrics used to measure the

11 Available at `http://cogcomp.cs.illinois.edu/page/software`.

performance of the discourse segmenter and the parser. Then we show the performance of our discourse segmenter, followed by the performance of our discourse parser.

6.1 Corpora

Whereas previous studies on discourse analysis only report their results on a particular corpus, to demonstrate the generality of our method, we experiment with texts from two very different genres: news articles and instructional how-to manuals.

Our first corpus is the standard *RST–DT* (Carlson, Marcu, and Okurowski 2002), which contains discourse annotations for 385 *Wall Street Journal* news articles taken from the Penn Treebank corpus (Marcus, Marcinkiewicz, and Santorini 1994). The corpus is partitioned into a training set of 347 documents and a test set of 38 documents. A total of 53 documents selected from both training and test sets were annotated by two human annotators. We measure human agreements based on this doubly annotated data set. We used 25 documents from the training set as our *development set*. In RST–DT, the original 25 rhetorical relations defined by Mann and Thompson (1988) are further divided into a set of 18 coarser relation classes with 78 finer-grained relations (see Carlson and Marcu [2001] for details). Our second corpus is the *instructional* corpus prepared by Subba and Di-Eugenio (2009), which contains discourse annotations for 176 how-to manuals on home repair. The corpus was annotated with 26 informational relations (e.g., *Preparation–Act*, *Act–Goal*).

For our experiments with the intra-sentential discourse parser, we extracted a sentence-level DT from a document-level DT by finding the sub-tree that exactly spans over the sentence. In RST–DT, by our count, 7,321 out of 7,673 sentences in the training set, 951 out of 991 sentences in the test set, and 1,114 out of 1,208 sentences in the doubly-annotated set have a well-formed DT. On the other hand, 3,032 out of 3,430 sentences in the instructional corpus have a well-formed DT. This forms the corpora for our experiments with intra-sentential discourse parsing. However, the existence of a well-formed DT is not a necessity for discourse segmentation; therefore, we do not exclude any sentence in our discourse segmentation experiments.

6.2 Evaluation (and Agreement) Metrics

In this subsection we describe the metrics used to measure both how much the annotators agree with each other, and how well the systems perform when their outputs are compared with human annotations for the discourse analysis tasks.

6.2.1 Metrics for Discourse Segmentation. Because sentence boundaries are considered to also be the EDU boundaries, we measure segmentation accuracy with respect to the intra-sentential segment boundaries, which is a standard method (Soricut and Marcu 2003; Fisher and Roark 2007). Specifically, if a sentence contains n EDUs, which corresponds to $n-1$ intra-sentential segment boundaries, we measure the segmenter's ability to correctly identify these $n-1$ boundaries. Let h be the total number of intra-sentential segment boundaries in the human annotation, m be the total number of intra-sentential segment boundaries in the model output, and c be the total number of correct segment boundaries in the model output. Then, we measure Precision (P), Recall (R), and F-score for segmentation performance as follows:

$$\mathrm{P} = \frac{\mathrm{c}}{\mathrm{m}}, \qquad \mathrm{R} = \frac{\mathrm{c}}{\mathrm{h}}, \qquad \text{and} \quad \mathrm{F{-}score} = \frac{2\mathrm{PR}}{\mathrm{P}+\mathrm{R}} = \frac{2\mathrm{c}}{\mathrm{h}+\mathrm{m}} \tag{14}$$

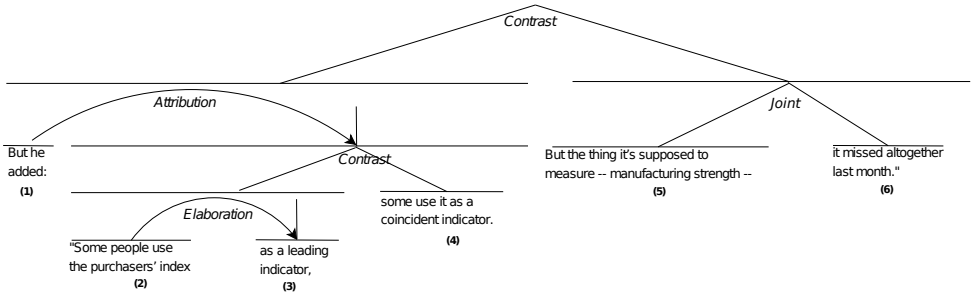


Figure 15
A hypothetical system-generated DT for the two sentences in Figure 1.

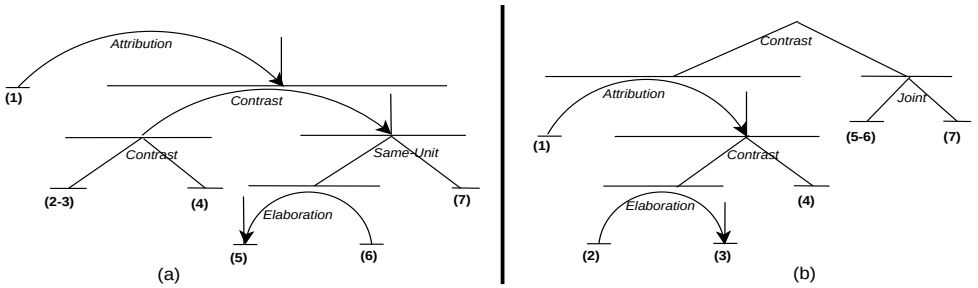


Figure 16
Measuring the accuracy of a discourse parser. (a) The human-annotated discourse tree. (b) The system-generated discourse tree.

6.2.2 Metrics for Discourse Parsing. To evaluate parsing performance, we use the standard unlabeled and labeled precision, recall, and F-score as proposed by Marcu (2000b). The unlabeled metric measures how accurate the discourse parser is in finding the right structure (i.e., the skeleton) of the DT, while the labeled metrics measure the parser's ability to find the right labels (i.e., nuclearity statuses or relation labels) in addition to the right structure. Assume, for example, that given the two sentences of Figure 1, our system generates the DT shown in Figure 15. In Figure 16, we show the same gold DT shown in Figure 1 (on the left), and the same system-generated DT shown in Figure 15 (on the right), when the two trees are aligned. For the sake of illustration, instead of showing the real EDUs, we only show their IDs. Notice that the automatic segmenter made two mistakes: (1) it broke the EDU marked 2–3 (*Some people use the purchasers' index as a leading indicator*) in the human annotation into two separate EDUs, and (2) it could not identify EDU 5 (*But the thing it's supposed to measure*) and EDU 6 (— *manufacturing strength* —) as two separate EDUs. Therefore, when we align the two annotations, we obtain seven EDUs in total.

In Table 2, we list all constituents of the two DTs and their associated labels at the span, nuclei, and relation levels. The recall (R) and precision (P) figures are shown at the bottom of the table. Notice that, following (Marcu 2000b), the relation labels are assigned to the children nodes rather than to the parent nodes in the evaluation process to deal with non-binary trees in human annotations. To our knowledge, no implementation of

Table 2
Measuring parsing accuracy (P = Precision, R = Recall).

	Spans		Nuclearity		Relations	
Constituents	Human	System	Human	System	Human	System
1–1	*	*	S	S	Attribution	Attribution
2–2		*		S		Elaboration
3–3		*		N		Span
4–4	*	*	N	N	Contrast	Contrast
5–5	*		N		Span	
6–6	*		S		Elaboration	
7–7	*	*	N	N	Same–Unit	Joint
2–3	*	*	N	N	Contrast	Contrast
5–6	*	*	N	N	Same–Unit	Joint
2–4	*	*	S	N	Contrast	Span
5–7	*	*	N	N	Contrast	Contrast
1–4		*		N		Contrast
2–7	*		N		Span	
	R = 7/10, P = 7/10		R = 6/10, P = 6/10		R = 4/10, P = 4/10	

the evaluation metrics was made publicly available. Therefore, to help other researchers, we have made our source code of the evaluation metrics publicly available.[12]

Given this evaluation setup, it is easy to understand that if the number of EDUs is the same in the human and system annotations (e.g., when the discourse parser uses gold discourse segmentation), and the discourse trees are binary, then we get the same figures for precision, recall, and F-score.

6.3 Discourse Segmentation Evaluation

In this section we present our experiments on discourse segmentation.

6.3.1 Experimental Set-up for Discourse Segmentation. We compare the performance of our discourse segmenter with the performance of the two publicly available discourse segmenters, namely, the discourse segmenters of the HILDA (Hernault et al. 2010) and SPADE (Soricut and Marcu 2003) systems. We also compare our results with the state-of-the-art results reported by Fisher and Roark (2007) on the RST–DT test set. In all our experiments when comparing two systems, we use *paired t-test* on the F-scores to measure statistical significance and report the p-value.

We ran HILDA with its default settings. For SPADE, we applied the same modifications to its default settings as described in Fisher and Roark (2007), which delivers significantly improved performance over its original version. Specifically, in our experiments on the RST–DT corpus, we trained SPADE using the human-annotated syntactic trees extracted from the Penn Treebank (Marcus, Marcinkiewicz, and Santorini 1994), and, during testing, we replaced the Charniak parser (Charniak 2000) with a more

12 Available from `alt.qcri.org/tools/`.

Table 3
Discourse segmentation results of different models on the two corpora. Performances significantly superior to SPADE are denoted by *.

	RST–DT							Instructional	
	Standard Test Set				Doubly	10-fold		10-fold	10-fold
	HILDA	SPADE	F&R	DS	Human	SPADE	DS	SPADE	DS
Precision	77.9	83.8	**91.3***	88.0*	98.5	83.7	**87.5***	65.1	**73.9***
Recall	70.6	86.8	89.7*	**92.3***	98.2	86.2	**89.9***	82.8	**89.7***
F-score	74.1	85.2	**90.5***	90.1*	98.3	84.9	**88.7***	72.8	**80.9***

accurate reranking parser (Charniak and Johnson 2005). However, because of the lack of gold syntactic trees in the instructional corpus, we trained SPADE in this corpus using the syntactic trees produced by the reranking parser. To avoid using the gold syntactic trees, we used the reranking parser in our system for both training and testing purposes. This syntactic parser was trained on the sections of the Penn Treebank not included in our test set. We applied the same canonical lexical head projection rules (Magerman 1995; Collins 2003) to lexicalize the syntactic trees as done in HILDA and SPADE.

Note that previous studies (Fisher and Roark 2007; Soricut and Marcu 2003; Hernault et al. 2010) on discourse segmentation only report their performance on the RST–DT test set. To compare our results with them, we evaluated our model on the RST–DT test set. In addition, we showed a more general performance of SPADE and our system on the two corpora based on 10-fold cross validation.[13] However, SPADE does not come with a training module for its segmenter. We reimplemented this module and verified its correctness by reproducing the results on the RST–DT test set.

6.3.2 Results for Discourse Segmentation. Table 3 shows the discourse segmentation results of different systems in Precision, Recall, and F-score on the two corpora. On the RST–DT corpus, HILDA's segmenter delivers the weakest performance, having an F-score of only 74.1. Note that the high segmentation accuracy reported by Hernault et al. (2010) is due to a less stringent evaluation metric. SPADE performs much better than HILDA with an absolute F-score improvement of 11.1%. Our segmenter DS outperforms SPADE with an absolute F-score improvement of 4.9% (p-value $<$ 2.4e-06), and also achieves comparable results to the ones of Fisher and Roark (2007) (F&R), even though we use fewer features.[14] Notice that human agreement for this task is quite high—namely, an F-score of 98.3 computed on the doubly-annotated portion of the RST–DT corpus mentioned in Section 6.1.

Because Fisher and Roark (2007) only report their results on the RST–DT test set and we did not have access to their system, we compare our approach only with SPADE when evaluating on a whole corpus based on 10-fold cross validation. On the RST–DT corpus, our segmenter delivers an absolute F-score improvement of 3.8 percentage points, which represents a more than 25% relative error rate reduction.

13 Because the two tasks—discourse segmentation and intra-sentential parsing—operate at the sentence level, the cross validation was performed over sentences for their evaluation.

14 Because we did not have access to the system or to the complete output/results of Fisher and Roark (2007), we were not able to perform a statistical significance test.

The improvement is higher on the instructional corpus with an absolute F-score improvement of 8.1 percentage points, which corresponds to a relative error reduction of 30%. The improvements for both corpora are statistically significant (p-value $<$ 3.0e-06). When we compare our results on the two corpora, we observe a substantial decrease in performance on the instructional corpus. This could be because of a smaller amount of data in this corpus and/or to the inaccuracies in the syntactic parser and taggers, which are trained on news articles. A promising future direction would be to apply effective domain adaptation methods (e.g., `easyadapt` [Daumé 2007]) to improve discourse segmentation performance in the instructional domain by leveraging the rich data in the news domain (i.e., RST–DT).

6.4 Discourse Parsing Evaluation

In this section we present our experiments on discourse parsing. First, we describe the experimental set-up. Then, we present the results of the parsers. While presenting the performance of our discourse parser, we show a breakdown of intra-sentential versus inter-sentential results, in addition to the aggregated results at the document level.

6.4.1 Experimental Set-up for Discourse Parsing. In our experiments on sentence-level (i.e., intra-sentential) discourse parsing, we compare our approach with SPADE (Soricut and Marcu 2003) on the RST–DT corpus, and with the ILP-based approach of Subba and Di-Eugenio (2009) on the instructional corpus, because they are the state of the art in their respective genres. For SPADE, we applied the same modifications to its default settings as described in Section 6.3.1, which leads to improved performance. Similarly, in our experiments on document-level (i.e., multi-sentential) parsing, we compare our approach with HILDA (Hernault et al. 2010) on the RST–DT corpus, and with the ILP-based approach (Subba and Di-Eugenio 2009) on the instructional corpus. The results for HILDA were obtained by running the system with default settings on the same inputs we provided to our system. Because we could not run the ILP-based system (not publicly available), we report the performance presented in their paper.

Our experiments on the RST–DT corpus use the same 18 coarser coherence relations (see Figure 18 later in this article), defined by Carlson and Marcu (2001) and also used in SPADE and HILDA systems. More specifically, the relation set consists of 16 relation categories and two *pseudo-relations*, namely, *Textual–Organization* and *Same–Unit*. After attaching the nuclearity statuses (NS, SN, NN) to these relations, we obtain 41 distinct relations.[15] Our experiments on the instructional corpus consider the same 26 primary relations (e.g., *Goal:Act*, *Cause:Effect*) used by Subba and Di-Eugenio (2009) and also treat the reversals of non-commutative relations as separate relations. That is, *Goal–Act* and *Act–Goal* are considered to be two different coherence relations. Attaching the nuclearity statuses to these relations provides 76 distinct relations.

Based on our experiments on the development set, the size of the automatically built bi-gram and tri-gram dictionaries was set to 95% of their total number of items, and the size of the unigram dictionary was set to 100%. Note that the unigram dictionary contains only special tags denoting EDU, sentence, and paragraph boundaries.

15 Not all relations take all the possible nuclearity statuses. For example, *Elaboration* and *Attribution* are mono-nuclear relations, and *Same–Unit* and *Joint* are multi-nuclear relations.

6.4.2 Evaluation of the Intra-Sentential Discourse Parser. This section presents our experimental evaluation on intra-sentential discourse parsing. First, we show the performance of the sentence-level parsers when they are provided with manual (or gold) discourse segmentations. This allows us to judge the parsing performance independently of the segmentation task. Then, we show the end-to-end performance of our intra-sentential framework, that is, the intra-sentential parsing performance based on automatic discourse segmentation.

Intra-sentential parsing results based on manual segmentation

Table 4 presents the intra-sentential discourse parsing results when manual discourse segmentation is used. Recall from our discussion on evaluation metrics in Section 6.2.2 that precision, recall, and F-score are the same when manual segmentation is used. Therefore, we report only one of them. Notice that our sentence-level discourse parser PAR-S consistently outperforms SPADE on the RST–DT test set in all three metrics, and the improvements are statistically significant (p-value < 0.01). Especially, on the relation labeling task, which is the hardest among the three tasks, we achieve an absolute F-score improvement of 12.2 percentage points, which represents a relative error rate reduction of 37.7%.

To verify our claim that capturing the sequential dependencies between DT constituents using a DCRF model actually contributes to the performance gain, we also compare our results with an intra-sentential parser (see CRF-NC in Table 4) that uses a simplified CRF model similar to the one shown in Figure 8. Although the simplified model has two hidden variables to model the structure and the relation of a DT constituent jointly, it does not have a chain structure, thus it ignores the sequential dependencies between DT constituents. The comparison in all three measures demonstrates that the improvements are indeed partly due to the DCRF model (p–value $<$ 0.01).[16] A comparison between CRF-NC and SPADE shows that CRF-NC significantly outperforms SPADE in all three measures (p-value < 0.01). This could be due to the fact that CRF-NC is trained discriminatively with a large number of features, whereas SPADE is trained generatively with only lexico-syntactic features.

Notice that the scores of our parser (PAR-S) are close to the human agreement on the doubly-annotated data, and these results on the RST–DT test set are also consistent with the mean scores over 10-folds on the whole RST–DT corpus.[17]

The improvements are higher on the instructional corpus, where we compare our mean results over 10-folds with the reported results of the ILP-based system of Subba and Di-Eugenio (2009), giving absolute F-score improvements of 5.4 percentage points, 17.6 percentage points, and 12.8 percentage points in span, nuclearity, and relations, respectively.[18] Our parser PAR-S reduces the errors by 76.1%, 62.4%, and 34.6% in span, nuclearity and relations, respectively.

If we compare the performance of our intra-sentential discourse parser on the two corpora, we notice that our parser PAR-S is more accurate in finding the right tree

16 The parsing performance reported in Table 4 for CRF-NC is when the CRF parsing model is trained on a balanced data set (an equal number of instances with S=1 and S=0); Training on full but imbalanced data set gives slightly lower results.

17 Our EMNLP and ACL publications (Joty, Carenini, and Ng 2012; Joty et al. 2013) reported slightly lower parsing accuracies. Fixing a bug in the parsing algorithm accounts for the difference.

18 Subba and Di-Eugenio (2009) report their results based on an arbitrary split between training and test sets. Because we did not have access to their particular split, we compare our model's performance based on 10-fold cross validation with their reported results. Also, because we did not have access to their system/output, we could not perform a significance test on the instructional corpus.

Table 4
Intra-sentential parsing results based on manual discourse segmentation. Performances significantly superior to SPADE are denoted by *.

	RST–DT					Instructional	
	Standard Test Set			10-fold	Doubly	Reported	10-fold
Scores	SPADE	CRF-NC	PAR-S	PAR-S	Human	ILP	PAR-S
Span	93.5	95.1*	**96.5***	95.4	95.7	92.9	**98.3**
Nuclearity	85.8	87.7*	**89.4***	88.6	90.4	71.8	**89.4**
Relation	67.6	76.6*	**79.8***	78.9	83.0	63.0	**75.8**

structure (see *Span* row in the table) on the instructional corpus. This may be due to the fact that sentences in the instructional domain are relatively short and contain fewer EDUs than sentences in the news domain, thus making it easier to find the right tree structure. However, when we compare the performance on the relation labeling task, we observe a decrease on the instructional corpus. This may be due to the small amount of data available for training and the imbalanced distribution of a large number of discourse relations (i.e., 76 with nuclearity attached) in this corpus.

Intra-sentential parsing results based on automatic segmentation

In order to evaluate the performance of the fully automatic sentence-level discourse analysis systems, we feed the intra-sentential discourse parsers the output of their respective discourse segmenters. Table 5 shows the (P)recision, (R)ecall, and (F)–score results for different evaluation metrics. We compare our intra-sentential parser PAR-S with SPADE on the RST–DT test set. We achieve absolute F-score improvements of 5.7 percentage points, 6.4 percentage points, and 9.5 percentage points in span, nuclearity, and relation, respectively. These improvements are statistically significant (p-value$<$0.001). Our system, therefore, reduces the errors by 24.5%, 21.4%, and 22.6% in span, nuclearity, and relations, respectively. These results are also consistent with the mean results over 10-folds on the whole RST–DT corpus.

The rightmost column in the table shows our mean results over 10-folds on the instructional corpus. We could not compare our system with the ILP-based approach of Subba and Di-Eugenio (2009) because no results were reported using an automatic segmenter. It is interesting to observe how much our parser is affected by an automatic segmenter on the two corpora (see Tables 4 and 5). Nevertheless, taking into account the

Table 5
Intra-sentential parsing results using automatic discourse segmentation. Performances significantly superior to SPADE are denoted by *.

	RST–DT									Instructional		
	Test set						10-fold			10-fold		
Scores	SPADE			PAR-S			PAR-S			PAR-S		
	P	R	F	P	R	F	P	R	F	P	R	F
Span	75.9	77.4	76.7	**80.8***	**84.0***	**82.4***	79.6	80.7	80.1	73.5	80.7	76.9
Nuclearity	69.8	70.5	70.2	**75.2***	**78.1***	**76.6***	73.9	76.5	75.2	64.6	71.0	67.6
Relation	57.4	58.5	58.0	**66.1***	**68.8***	**67.5***	65.2	67.4	66.8	54.8	60.4	57.5

Table 6
Parsing results of document-level parsers using manual segmentation. Performances significantly superior to HILDA (p-value <0.0001) are denoted by *. Significant differences between TSP 1-1 and TSP SW (p-value <0.01) are denoted by †.

	RST–DT						Instructional		
Metrics	HILDA	CRF-O	CRF-T	TSP 1-1	TSP SW	Human	ILP	TSP 1-1	TSP SW
Span	74.68	77.02*	81.34*	**82.56***	**83.84***†	88.70	70.35	**80.67**	**82.88**†
Nuc.	58.99	63.84*	66.52*	**68.32***	**68.90***	77.72	49.47	**63.03**	**64.13**
Rel.	44.32	48.46*	53.01*	**55.83***	**55.87***	65.75	35.44	**43.52**	**44.20**

segmentation results in Table 3, this is not surprising because previous studies (Soricut and Marcu 2003) have already shown that automatic segmentation is the primary impediment to high accuracy discourse parsing. This demonstrates the need for a more accurate discourse segmentation model in the instructional genre.

6.4.3 Evaluation of the Complete Parser. We experiment with our full document-level discourse parser on the two corpora using the two parsing approaches described in Section 4.3, namely, 1S-1S and the sliding window. On RST–DT, the standard split was used for training and testing. On the instructional corpus, Subba and Di-Eugenio (2009) used 151 documents for training and 25 documents for testing. Because we did not have access to their particular split, we took five random samples of 151 documents for training and 25 documents for testing, and report the average performance over the five test sets.

Table 6 presents results for our two-stage discourse parser (TSP) using approaches 1S-1S (TSP 1-1) and the sliding window (TSP SW) on manually segmented texts. Recall that precision, recall, and F-score are the same when manual segmentation is used. We compare our parser with the state-of-the-art on the two corpora: HILDA (Hernault et al. 2010) on RST–DT, and the ILP-based approach (Subba and Di-Eugenio 2009) on the instructional domain. On both corpora, our systems outperform existing systems by a wide margin (p-value <7.1e-05 on RST–DT).[19] On RST–DT, our parser TSP 1-1 achieves absolute improvements of 7.9 percentage points, 9.3 percentage points, and 11.5 percentage points in span, nuclearity, and relation, respectively, over HILDA. This represents relative error reductions of 31.2%, 22.7%, and 20.7% in span, nuclearity, and relation, respectively.

Beside HILDA, we also compare our results with two baseline parsers on RST–DT: (1) CRF-O, which uses a single unified CRF-based parsing model shown in Figure 8 (the one used for multi-sentential parsing) without distinguishing between intra- and multi-sentential parsing, and (2) CRF-T, which uses two different CRF-based parsing models for intra- and multi-sentential parsing in the two-stage approach 1S-1S, both models having the same structure as in Figure 8. Thus, CRF-T is a variation of TSP 1-1, where the DCRF-based (chain-structured) intra-sentential parsing model is replaced with a simpler CRF-based parsing model.[20] Note that although CRF-O does not *explicitly*

19 Because we did not have access to the output or to the system of Subba and Di-Eugenio (2009), we were not able to perform a significance test on the instructional corpus.

20 The performance of this model for intra-sentential parsing is reported in Table 4 under the name CRF-NC.

discriminate between intra- and multi-sentential parsing, it uses *N-gram features* that include sentence and EDU boundaries to encode this information into the model.

Table 6 shows that both CRF-O and CRF-T outperform HILDA by a good margin (p-value <0.0001). This improvement can be attributed to the optimal parsing algorithm and better feature selection strategy. When we compare CRF-T with CRF-O, we notice significant performance gains for CRF-T (p-value <0.001). The absolute gains are 4.32 percentage points, 2.68 percentage points, and 4.55 percentage points in span, nuclearity, and relation, respectively. This comparison clearly demonstrates the benefit of using a two-stage approach with two different parsing models over a framework with one single unified parsing model. Finally, when we compare our best results with the human agreements, we still observe room for further improvement in all three measures.

On the instructional genre, our parser TSP 1-1 delivers absolute F-score improvements of 10.3 percentage points, 13.6 percentage points, and 8.1 percentage points in span, nuclearity, and relations, respectively, over the ILP-based approach of Subba and Di-Eugenio (2009). Our parser, therefore, reduces errors by 34.7%, 26.9%, and 12.5% in span, nuclearity, and relations, respectively.

If we compare the performance of our discourse parsers on the two corpora, we observe lower results on the instructional corpus. There could be two reasons for this. First, the instructional corpus has a smaller amount of data with a larger set of relations (76 with nuclearity attached). Second, some of the frequent relations are semantically very similar (e.g., *Preparation-Act, Step1-Step2*), which makes it difficult even for the human annotators to distinguish them (Subba and Di-Eugenio 2009).

Comparison between our two document-level parsing approaches reveals that TSP SW significantly outperforms TSP 1-1 only in finding the right structure on both corpora (p-value <0.01). Not surprisingly, the improvement is higher on the instructional corpus. A likely explanation is that the instructional corpus contains more leaky boundaries (12%), allowing the sliding window approach to be more effective in finding those, without inducing much noise for the labels. This demonstrates the potential of TSP SW for data sets with even more leaky boundaries, e.g., the Dutch (Vliet and Redeker 2011) and the German Potsdam (Stede 2004) corpora. However, it would be interesting to see how other heuristics to do consolidation in the *cross* condition (Section 4.3.2) perform.

To analyze errors made by TSP SW, we looked at some poorly parsed examples and found that although TSP SW finds more correct structures, a corresponding improvement in labeling relations is not present because in some cases, it tends to induce noise from the neighboring sentences for the labels. For example, when parsing is performed on the first sentence in Figure 1 in isolation using 1S-1S, our parser rightly identifies the *Contrast* relation between EDUs 2 and 3. But, when it is considered with its neighboring sentences by the sliding window, the parser labels it as *Elaboration*. A promising strategy to deal with this and similar problems would be to apply both approaches to each sentence and combine them by consolidating three probabilistic decisions, namely, the one from 1S-1S and the two from the sliding window.

6.4.4 k-best Parsing Results Based on Manual Segmentation. As described in Section 4.2, a straight-forward modification of our probabilistic parsing algorithm allows us to generate a list of *k*-best parse hypotheses for a given text. We adapt our parsing algorithm accordingly to produce *k* most probable DTs for each text, and measure the oracle accuracy based on the F-scores of the *Relation* metric which gives aggregated evaluation

Table 7
Oracle scores as a function of k of k-best sentence-level parses on RST–DT.

k	1	2	3	4	5	10	15	20	25	30
PAR-S	79.77	84.42	86.55	87.68	88.09	90.37	91.74	92.57	92.95	93.22

on structure and relation labels (see Table 2). Specifically, the oracle accuracy *O−score* for k-best discourse parsing is measured as follows:

$$\text{O}-\text{score} = \frac{\sum_{\text{i}=1}^{\text{N}} \max_{\text{j}=1}^{\text{k}} \text{F}-\text{score}_{\text{r}}(\text{g}_{\text{i}}, \text{h}_{\text{i}}^{\text{j}})}{\text{N}} \tag{15}$$

where N is the total number of texts (sentences or documents) evaluated, g_i is the gold DT annotation for text i, h_i^j is the j^{th} parse hypothesis generated by the parser for text i, and F-$score_r(g_i, h_i^j)$ is the F-score accuracy of hypothesis h_i^j on the *Relation* metric, which essentially measures how similar h_i^j is to g_i in terms of its structure and labels.

Table 7 presents the oracle scores of our intra-sentential parser PAR-S on the RST–DT test set as a function of k of k-best parsing. The 1-best result tells that the parser has the base accuracy of 79.8%. The 2-best shows dramatic oracle-rate improvements (i.e., 4.65% absolute), meaning that often our parser generates the best tree as its top two outputs. 3-best and 4-best also show moderate improvements (about 2%). Things start to slow down afterwards, and we achieve oracle rates of 90.37% and 92.57% at 10−best and 20-best, respectively. The 30-best parsing gives an oracle score of 93.2%.

The results of our k-best intra-sentential discourse parser demonstrate that a k-best reranking approach like that of Collins and Koo (2005) and Charniak and Johnson (2005) used for syntactic parsing can potentially improve the parsing accuracy even further by exploiting additional global features of the candidate discourse trees as evidence.

The scenario is quite different at the document-level; Table 8 shows the k-best parsing results of TSP 1S-1S on the RST–DT test set. The improvements in oracle-rate are small at the document-level when compared with the sentence-level parsing. For example, the 2-best and the 5-best improve over the base accuracy by only 0.7 percentage points and 1.0 percentage points, respectively. The improvements get even slower after that. However, this is not surprising because generally document-level DTs are big with many constituents, and only a very few of these constituents change from k-best to $k+1$-best parsing. These small changes among the candidate DTs do not contribute much to the overall F-score accuracy (for further clarification see how F-score is calculated in Section 6.2.2).

Table 8
Oracle scores as a function of k of k-best document-level parses on RST–DT.

k	1	2	3	4	5	10	15	20	25	30
TSP 1S-1S	55.83	56.52	56.67	56.80	56.91	57.23	57.54	57.65	57.67	57.74

Table 9
Parsing results using different subsets of features on RST–DT test set.

	Sentence-level					Document-level (TSP 1-1)				
Scores	Dom	+Org	+N-gr	+Con	+Sub	Org	+N-gr	+L-ch	+Con	+Sub
Span	91.3	92.1	93.3	94.6	96.5	74.2	75.8	78.5	80.9	82.6
Nuclearity	78.2	80.3	83.8	86.8	89.4	60.6	63.7	65.6	66.9	68.3
Relation	66.2	68.1	74.1	76.3	79.8	46.3	50.1	52.4	53.6	55.8

The results of our *k*-best document-level parsing suggest that often the best tree is missing in the top *k* parses. Thus, a reranking of *k*-best document-level parses may not be a suitable option for further improvement at the document-level. An alternative to *k*-best reranking is to use a sampling-based parsing strategy (Wick et al. 2011) to explore the space of possible trees, as recently used for dependency parsing (Zhang et al. 2014). However, note that the potential gain we may obtain by using a reranker at the sentence level will also improve the (combined) accuracy of the document-level parser.

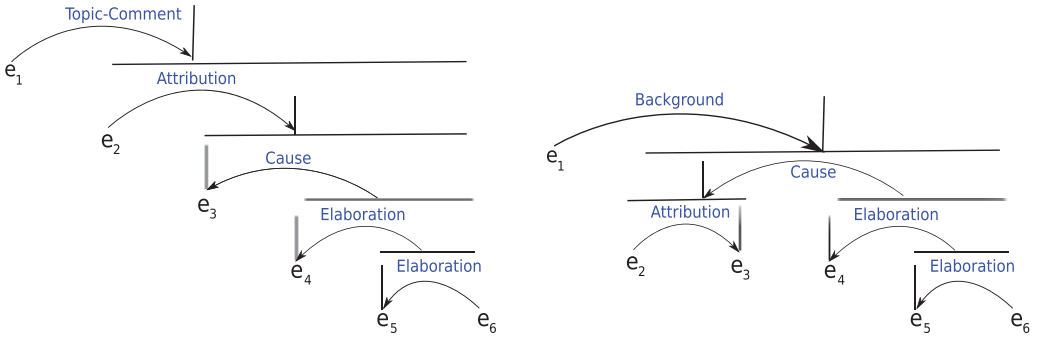
6.4.5 Analysis of Features. To analyze the relative importance of different features used in our parsing models, Table 9 presents the sentence- and document-level parsing results on a manually segmented RST–DT test set using different subsets of features. The feature subsets were defined in Section 4.1.4. In each parsing condition, the subsets of features are added incrementally, based on their availability and historical importance. The columns in Table 9 represent the inclusion order of the feature subsets.

Because SPADE (Soricut and Marcu 2003) achieved the previous best results on intra-sentential parsing using ***Dom**inance set* features, these are included as the initial set of features in our intra-sentential parsing model. In HILDA, Hernault et al. (2010) demonstrate the importance of ***Org**anizational* and ***N-gr**am* features for full text parsing. We add these two feature subsets one after another in our intra- and multi-sentential parsing models.[21] ***Con**textual* features require other features to be computed; thus they were added after those features. Because computation of ***Sub**-structural* features requires an initial parse tree (i.e., when the parser is applied), they are added at the very end.

Notice that inclusion of every new subset of features appears to improve the performance over the previous set. Specifically, for sentence-level parsing, when we add the *Organizational* features with the *Dominance set* features, we achieve about 2 percentage points absolute improvements in nuclearity and relations. With *N-gram* features, the gain is even higher: 6 percentage points in relations and 3.5 percentage points in nuclearity for sentence-level parsing, and 3.8 percentage points in relations and 3.1 percentage points in nuclearity for document-level parsing. This demonstrates the utility of the N-gram features, which is also consistent with the previous findings of duVerle and Prendinger (2009) and Schilder (2002).

The features extracted from *Lexical chains* (***L-ch***) have also proved to be useful for document-level parsing. They deliver absolute improvements of 2.7 percentage points, 2.9 percentage points, and 2.3 percentage points in span, nuclearity, and relations,

21 *Text structural* features are included in the *Organizational* features for multi-sentential parsing.



(a) A human-annotated discourse tree. (b) A system-generated discourse tree.

Figure 17
Discourse trees generated by human annotator and our system for the text *[what's more,]$_{e_1}$ [he believes]$_{e_2}$ [seasonal swings in the auto industry this year aren't occurring at the same time in the past,]$_{e_3}$ [because of production and pricing differences]$_{e_4}$ [that are curbing the accuracy of seasonal adjustments]$_{e_5}$] [built into the employment data.]$_{e_6}$*

respectively. Including the *Contextual* features further gives improvements of 3 percentage points in nuclearity and 2.2 percentage points in relation for sentence-level parsing, and 1.3 percentage points in nuclearity, and 1.2 percentage points in relation for document-level parsing. Notice that *Sub-structural* features are more beneficial for document-level parsing than they are for sentence-level parsing, that is, an improvement of 2.2 percentage points versus an improvement of 0.9 percentage points. This is not surprising because document-level DTs are generally much larger than sentence-level DTs, making the sub-structural features more effective for document-level parsing.

6.4.6 Error Analysis. We further analyze the errors made by our discourse parser. As described in previous sections, the parser could be wrong in finding the right structure as well as the right nuclearity and relation labels. Figure 17 presents an example where our parser makes mistakes in finding the right structure (notice the units connected by *Attribution* and *Cause* in the two example DTs) and the right relation label (*Topic-Comment* vs. *Background*). The comparison between intra- and multi-sentential parsing results presented in Sections 6.4.2 and 6.4.3 tells us that the errors in structure occur more frequently when the DT is large (e.g., at the document level) and the parsing model fails to capture the long-range structural dependencies between the DT constituents.

To further analyze the errors made by our parser on the hardest task of relation labeling, in Figure 18 we present the confusion matrix for our document-level parser TSP 1-1 on the RST–DT test set. In order to judge independently the ability of our parser to assign the correct relation labels, the confusion matrix is computed based on the constituents (see Table 2), where our parser found the right span (i.e., structure).[22] The relations in the matrix are ordered according to their frequency in the training set.

In general, the errors can be explained by two different causes acting together: (1) imbalanced distribution of the relations in the corpus, and (2) semantic similarity between the relations. The most frequent relation *Elaboration* tends to overshadow others, especially the ones that are semantically similar (e.g., *Explanation*, *Background*)

22 Therefore, the counts of the relations shown in the table may not match the ones in the test set.

	T-C	T-O	T-CM	M-M	CMP	EV	SU	CND	EN	CA	TE	EX	BA	CO	JO	S-U	AT	EL
T-C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
T-O	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T-CM	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	7
M-M	0	0	0	10	0	0	0	0	0	0	0	1	1	0	0	0	1	3
CMP	0	0	0	1	4	0	0	1	0	1	0	3	3	0	1	1	0	2
EV	0	0	0	0	0	0	0	0	0	0	0	2	0	0	2	0	2	11
SU	0	0	0	0	0	0	8	0	0	0	0	0	0	0	1	0	0	12
CND	0	0	0	0	0	0	0	22	0	0	0	0	1	3	0	0	3	2
EN	0	0	0	0	0	0	0	1	24	1	0	0	0	0	0	0	1	7
CA	0	0	0	0	0	0	0	0	2	3	0	4	2	2	7	0	3	11
TE	0	0	0	1	0	0	0	1	2	0	7	1	9	1	9	0	3	4
EX	0	0	0	1	0	0	0	0	1	5	0	12	0	1	3	0	3	12
BA	0	0	0	1	0	0	0	1	0	1	4	1	19	2	6	1	5	12
CO	0	0	0	1	2	0	0	2	0	1	3	2	2	33	7	0	0	9
JO	0	0	0	0	0	0	1	2	0	1	1	1	1	2	57	1	0	13
S-U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	85	1	0
AT	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	3	272	9
EL	0	1	0	0	0	0	0	0	14	6	1	8	1	0	8	2	2	359

Figure 18
Confusion matrix for relation labels on the RST–DT test set. The *y*-axis represents *true* and *x*-axis represents *predicted* relations. The relations are Topic-Change (T-C), Topic-Comment (T-CM), TextualOrganization (T-O), Manner-Means (M-M), Comparison (CMP), Evaluation (EV), Summary (SU), Condition (CND), Enablement (EN), Cause (CA), Temporal (TE), Explanation (EX), Background (BA), Contrast (CO), Joint (JO), Same–Unit (S-U), Attribution (AT), and Elaboration (EL).

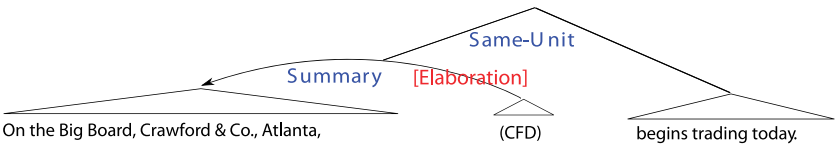


Figure 19
Our system mistakenly labels a *Summary* as *Elaboration*.

and less frequent (e.g., *Summary*, *Evaluation*). Furthermore, our models sometimes fail to distinguish relations that are semantically similar (e.g., *Temporal vs. Background*, *Cause vs. Explanation*).

Now, let us look more closely at a few of these errors. Figure 19 presents an example where our parser mistakenly labels a *Summary* as *Elaboration*. Clearly, in this example the text in parentheses (i.e., *(CFD)*) is an acronym or summary of the text to the left. However, parenthesized texts are also used to provide additional information (i.e., to elaborate), as exemplified in Figure 20 by two text snippets from the RST-DT. Notice that although the structure of the text *(widow of the ..)* in the first example is quite distinguishable from the structure of *(CFD)*, the text *(D., Maine)* in the second example is similar to *(CFD)* in structure, thus it confuses our model.[23]

Figure 21 presents two examples where our parser mistakenly labels *Background* and *Cause* as *Elaboration*. However, notice that the two discourse relations (i.e., *Background*

23 *D., Maine* in this example refers to *Democrat from state Maine*.

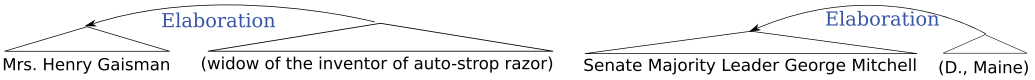


Figure 20
Two examples of *Elaboration* by texts in parentheses.

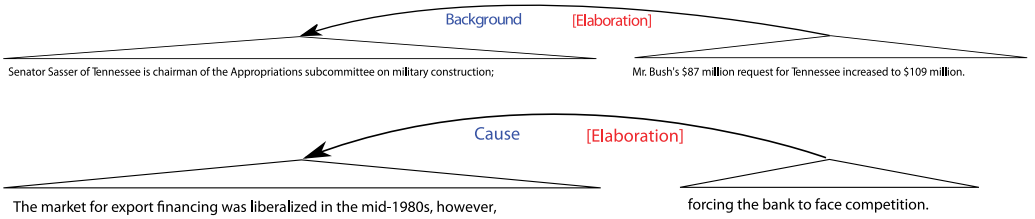


Figure 21
Confusion between *Background/Cause* and *Elaboration*.

vs. Elaboration and *Cause vs. Elaboration*) in these examples are semantically very close, and arguably both can be applicable.

Given these observations, we see two possible ways to improve our system. First, we would like to use a more robust method (e.g., *ensemble* methods with *bagging*) to deal with the imbalanced distribution of relations, along with taking advantage of richer semantic knowledge (e.g., compositional semantics) to cope with the errors caused by semantic similarity between the relations. Second, to capture long-range dependencies between DT constituents, we would like to explore the idea of k-best discriminative reranking using tree kernels (Dinarelli, Moschitti, and Riccardi 2011). Because our parser already produces k most probable DTs, developing a reranker based on discourse tree kernels is very much within our reach.

7. Conclusions and Future Directions

In this article we have presented CODRA, a complete probabilistic discriminative framework for performing rhetorical analysis in the RST framework. CODRA comprises components for performing both discourse segmentation and discourse parsing. The discourse segmenter is a binary classifier based on a maximum entropy model, and the discourse parser applies an optimal parsing algorithm to probabilities inferred from two CRF models: one for intra-sentential parsing and the other for multi-sentential parsing. The CRF models effectively represent the structure and the label of discourse tree constituents jointly. Furthermore, the DCRF model for intra-sentential parsing captures the sequential dependencies between the constituents. The two separate parsing models use their own informative feature sets and the distributional variations of the relation labels in their respective parsing conditions.

We have also presented two approaches to effectively combine the intra-sentential and the multi-sentential parsing modules, which can exploit the strong correlation observed between the text structure and the structure of the discourse tree. The first approach (1S–1S) builds a DT for every sentence using the intra-sentential parser, and then

runs the multi-sentential parser on the resulting sentence-level DTs. To deal with leaky boundaries, our second approach (sliding window) builds sentence-level discourse sub-trees by applying the intra-sentential parser on a sliding window, covering two adjacent sentences and then consolidating the results produced by overlapping windows. After that, the multi-sentential parser takes all these sentence-level sub-trees and builds a full rhetorical parse for the whole document.

Finally, we have extended the parsing algorithm to generate k most probable parse hypotheses for each input text, which could be used in a reranker to improve over the initial ranking using global features like long-range structural dependencies.

Empirical evaluations on two different genres demonstrate that our approach to discourse segmentation achieves state-of-the-art performance more efficiently using fewer features. A series of experiments on the discourse parsing task shows that both our intra- and multi-sentential parsers significantly outperform the state of the art, often by a wide margin. A comparison between our combination strategies reveals that the sliding window approach is more robust across domains. Furthermore, the oracle accuracy computed based on the k-best parse hypotheses generated by our parser demonstrates that a reranker could potentially improve the accuracy even further.

Our error analysis reveals that although the sliding window approach finds more correct tree structures, in some cases it induces noise for the relation labels from the neighboring sentences. With respect to the performance of our discourse parser on the relation labeling task we also found that the most frequent relations tend to mislead the identification of the less frequent ones, and the models sometimes fail to distinguish relations that are semantically similar.

The work presented in this article leads us to several interesting future directions. Our short-term goal is to develop a k-best discriminative reranking discourse parser using tree kernels applied to discourse trees. We also plan to investigate to what extent discourse segmentation and discourse parsing can be performed jointly.

We would also like to explore how our system performs on other genres like conversational (e.g., blogs, e-mails) and evaluative (e.g., customer reviews) texts. To address the problem of limited annotated data in various genres, we are planning to develop an interactive version of our system that will allow users to fix the output of the system with minimal effort and let the system learn from that feedback.

Another interesting future direction is to perform extrinsic evaluations of our system in downstream applications. One important application of rhetorical structure is text summarization, where a significant challenge is producing not only informative but also coherent summaries. A number of researchers have already investigated the utility of rhetorical structure for measuring text importance (i.e., informativeness) in summarization (Marcu 2000b; Daumé and Marcu 2002; Louis, Joshi, and Nenkova 2010). Recently, Christensen et al. (2013, 2014) propose to perform sentence selection and ordering at the same time, and use constraints on discourse structure to make the summaries coherent. However, they represent the discourse as an unweighted directed graph, which is shallow and not sufficiently informative in most cases. Furthermore, their approach does not allow compression at the sentence level, which is often beneficial in summarization. In the future, we would like to investigate the utility of our rhetorical structure for performing sentence compression, selection, and ordering in a joint summarization process.

Discourse structure can also play important roles in sentiment analysis. A key research problem in sentiment analysis is extracting fine-grained opinions about different aspects of a product. Several recent papers (Somasundaran 2010; Lazaridou, Titov, and

Sporleder 2013) exploited the rhetorical structure for this task. Another challenging problem is assessing the overall opinion expressed in a review because not all sentences in a review contribute equally to the overall sentiment. For example, some sentences are subjective, whereas others are objective (Pang and Lee 2004); some express the main claims, whereas others support them (Taboada et al. 2011); some express opinions about the main entity, whereas others are about the peripherals. Discourse structure could be useful to capture the relative weights of the discourse units towards the overall sentiment. For example, the nucleus and satellite distinction along with the rhetorical relations could be useful to infer the relative weights of the connecting discourse units.

Among other applications of discourse structure, Machine Translation (MT) and its evaluation have received a resurgence of interest recently. A workshop dedicated to discourse in machine translation was arranged recently at the ACL 2013 conference (Webber et al. 2013). Researchers believe that MT systems should consider discourse phenomena that go beyond the current sentence to ensure consistency in the choice of lexical items or referring expressions, and the fact that source-language coherence relations are also realized in the target language (i.e., translating at the document-level [Hardmeier, Nivre, and Tiedemann 2012]). Guzmán et al. (2014a, 2014b) and Joty et al. (2014) propose new discourse-aware automatic evaluation metrics for MT systems using our discourse analysis tool. They demonstrate that sentence-level discourse information is complementary to the state-of-the-art evaluation metrics, and by combining the discourse-based metrics with the metrics from the ASIYA MT evaluation toolkit (Giménez and Màrquez 2010), they won the WMT 2014 metrics shared task challenge (Macháček and Bojar 2014) both at the segment- and at the system-level. These results suggest that discourse structure helps to distinguish better translations from worse ones. Thus, it would be interesting to explore whether discourse information can be used to rerank alternative MT hypotheses as a post-processing step for the MT output.

A longer-term goal is to extend our framework to also work with graph structures of discourse, as recommended by several recent discourse theories (Wolf and Gibson 2005). Once we achieve similar performance on graph structures, we will perform extrinsic evaluations to determine their relative utility for various NLP tasks.

Finally, we hope that the online demo, the source code of CODRA, and the evaluation metrics that we made publicly available in this work will facilitate other researchers in extending our work and in applying discourse parsing to their NLP tasks.

Bibliographic Note

Portions of this work were previously published in two conference proceedings (Joty, Carenini, and Ng 2012; Joty et al. 2013). This article significantly extends our previous work in several ways, most notably: (1) we extend the parsing algorithm to generate k-most probable parse hypotheses for each input text (Section 4.2); (2) we show the oracle accuracies for k-best discourse parsing both at the sentence level and at the document level (Section 6.4.4); (3) to support our claim, we compare our best results with several variations of our approach (see CRF-NC in Section 6.4.2, and CRF-O and CRF-T in Section 6.4.3); (4) we analyze the relative importance of different features for intra- and multi-sentential discourse parsing (Section 6.4.5); and (5) we perform in-depth error analysis of our complete rhetorical analysis framework (Section 6.4.6).

Appendix A. Sample Output Generated by an Online Demo of CODRA

Rhetorical Analysis Demo

Enter your raw text here (currently only supports English):

Delwar Hossain and his wife were charged with homicide in December. The couple arrived at Dhaka magistrates court, and have now been jailed after their plea for bail was rejected.

Although arrests warrants had been issued in December, they had been living freely in Dhaka. It was not clear why they decided to give themselves up.

They face a maximum sentence of life in prison if convicted. The Tazreen fire was the country's deadliest garment factory fire, and brought attention to working conditions in the all-important garment industry.

Perform discourse segmentation only
Perform discourse parsing (includes segmentation)
Execute

Output in Textual format:

Root (span 1 11)

Nucleus (span 1 4) (rel2par span)

Nucleus (leaf 1) (rel2par span) | Text: Delwar Hossain and his wife were charged with homicide in December .

Satellite (span 2 4) (rel2par Elaboration)

Nucleus (leaf 2) (rel2par Joint) | Text: The couple arrived at Dhaka magistrates court ,

Nucleus (span 3 4) (rel2par Joint)

Nucleus (leaf 3) (rel2par span) | Text: and have now been jailed

Satellite (leaf 4) (rel2par Background) | Text: after their plea for bail was rejected .

Satellite (span 5 11) (rel2par Elaboration)

Nucleus (span 5 6) (rel2par span)

Satellite (leaf 5) (rel2par Contrast) | Text: Although arrests warrants had been issued in December ,

Nucleus (leaf 6) (rel2par span) | Text: they had been living freely in Dhaka .

Satellite (span 7 11) (rel2par Elaboration)

Nucleus (span 7 9) (rel2par span)

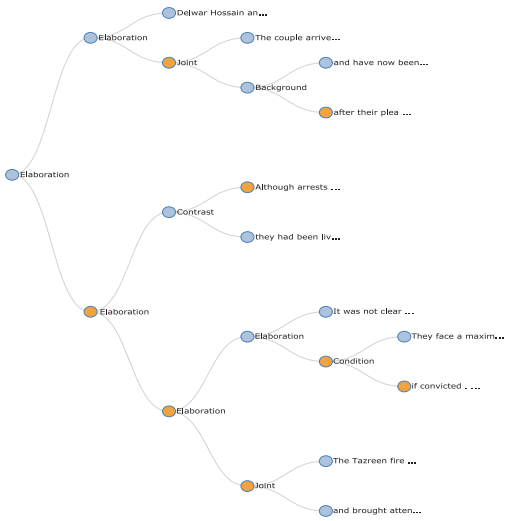
Nucleus (leaf 7) (rel2par span) | Text: It was not clear why they decided to give themselves up .

Satellite (span 8 9) (rel2par Elaboration)

Nucleus (leaf 8) (rel2par span) | Text: They face a maximum sentence of life in prison

Satellite (leaf 9) (rel2par Condition) | Text: if convicted .

Satellite (span 10 11) (rel2par Elaboration)



Output of discourse segmentation

[Delwar Hossain and his wife were charged with homicide in December .] [The couple arrived at Dhaka magistrates court ,][and have now been jailed][after their plea for bail was rejected .] [Although arrests warrants had been issued in December ,][they had been living freely in Dhaka .] [It was not clear why they decided to give themselves up .] [They face a maximum sentence of life in prison][if convicted .] [The Tazreen fire was the country 's deadliest garment factory fire ,][and brought attention to working conditions in the all-important garment industry .]

Acknowledgments

The authors acknowledge the funding support of NSERC Canada Graduate Scholarship (CGS-D). Many thanks to Bonnie Webber, Amanda Stent, Carolyn Rose, Lluis Marquez, Samantha Wray, and the anonymous reviewers for their insightful comments on an earlier version of this article.

References

Althaus, Ernst, Denys Duchier, Alexander Koller, Kurt Mehlhorn, Joachim Niehren, and Sven Thiel. 2003. An efficient graph algorithm for dominance constraints. *Journal of Algorithms*, 48(1):194–219.

Asher, Nicholas and Alex Lascarides, 2003. *Logics of Conversation*. Cambridge University Press.

Barzilay, Regina and Michael Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th European Chapter Meeting of the Association for Computational Linguistics, Workshop on Intelligent Scalable Test Summarization*, pages 10–17, Madrid.

Biran, Or and Owen Rambow. 2011. Identifying justifications in written dialogs by classifying text as argumentative. *International Journal of Semantic Computing*, 5(4):363–381.

Blair-Goldensohn, Sasha, Kathleen McKeown, and Owen Rambow. 2007. Building and refining rhetorical-semantic relation models. In *Proceedings of the Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL'07, pages 428–435. Rochester, NY.

Blitzer, John. 2008. *Domain Adaptation of Natural Language Processing Systems*. Ph.D. thesis, University of Pennsylvania.

Breiman, Leo. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.

Carlson, Lynn and Daniel Marcu. 2001. Discourse tagging reference manual. Technical Report ISI-TR-545, University of Southern California Information Sciences Institute.

Carlson, Lynn, Daniel Marcu, and Mary Ellen Okurowski. 2002. RST Discourse Treebank (RST–DT) LDC2002T07. Linguistic Data Consortium, Philadelphia.

Chali, Yllias and Shafiq Joty. 2007. Word sense disambiguation using lexical cohesion. In *Proceedings of SemEval-2007*, pages 476–479, Prague.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL'00, pages 132–139, Seattle, WA.

Charniak, Eugene and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL'05, pages 173–180, Ann Arbor, MI.

Christensen, Janara, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT'13, pages 1163–1173, Atlanta, GA.

Christensen, Janara, Stephen Soderland, Gagan Bansal, and Mausam. 2014. Hierarchical summarization: Scaling up multi-document summarization. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL'13, pages 902–912, Baltimore, MD.

Collins, Michael. 2003. Head-driven statistical models for natural language Parsing. *Computational Linguistics*, 29(4):589–637.

Collins, Michael and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1):25–70.

Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Cristea, Dan, Nancy Ide, and Laurent Romary. 1998. Veins theory: A model of global discourse cohesion and coherence. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and of the 17th International Conference on Computational Linguistics (COLING/ACL'98)*, pages 281–285. Montreal.

Danlos, Laurence. 2009. D-STAG: A discourse analysis formalism based on synchronous TAGs. *TAL*, 50(1):111–143.

Daumé, III, Hal. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, ACL'07, pages 256–263, Prague.

Daumé, III, Hal and Daniel Marcu. 2002. A noisy-channel model for document compression. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, ACL '02, pages 449–456, Philadelphia, PA.

Dinarelli, Marco, Alessandro Moschitti, and Giuseppe Riccardi. 2011. Discriminative reranking for spoken language understanding. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, 20:526–539.

duVerle, David and Helmut Prendinger. 2009. A novel discourse parser based on support vector machine classification. 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*, pages 665–673, Suntec.

Egg, Markus, Alexander Koller, and Joachim Niehren. 2001. The constraint language for lambda structures. *Journal of Logic, Language and Information*, 10(4):457–485.

Eisner, Jason. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, pages 340–345, Copenhagen.

Fellbaum, Christiane. 1998. *WordNet—An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Feng, Vanessa and Graeme Hirst. 2012. Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, ACL '12, pages 60–68, Jeju Island.

Feng, Vanessa and Graeme Hirst. 2014. A linear-time bottom-up discourse parser with constraints and post-editing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL '14, pages 511–521, Baltimore, MD.

Finkel, Jenny Rose, Alex Kleeman, and Christopher Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, ACL'08, pages 959–967, Columbus, OH.

Fisher, Seeger and Brian Roark. 2007. The utility of parse-derived features for automatic discourse segmentation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, ACL'07, pages 488–495, Prague.

Galley, Michel and Kathleen McKeown. 2003. Improving word sense disambiguation in lexical chaining. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, IJCAI'03, pages 1486–1488, Acapulco.

Galley, Michel, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics - Volume 1*, ACL '03, pages 562–569, Sapporo.

Ghosh, Sucheta, Richard Johansson, Giuseppe Riccardi, and Sara Tonelli. 2011. Shallow discourse parsing with conditional random fields. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, IJCNLP'11, pages 1071–1079, Chiang Mai.

Giménez, Jesús and Lluís Màrquez. 2010. Linguistic measures for automatic machine translation evaluation. *Machine Translation*, 24(3–4):77–86.

Guzmán, Francisco, Shafiq Joty, Lluís Màrquez, Alessandro Moschitti, Preslav Nakov, and Massimo Nicosia. 2014a. Learning to differentiate better from worse translations. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 214–220, Doha.

Guzmán, Francisco, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2014b. Using discourse structure improves machine translation evaluation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 687–698, Baltimore, MD.

Halliday, Michael and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.

Hardmeier, Christian, Joakim Nivre, and Jörg Tiedemann. 2012. Document-wide decoding for phrase-based statistical machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1179–1190, Jeju Island.

Hernault, Hugo, Helmut Prendinger, David duVerle, and Mitsuru Ishizuka. 2010. HILDA: A discourse parser using support vector machine classification. *Dialogue and Discourse*, 1(3):1–33.

Hirst, Graeme and David St-Onge. 1997. Lexical Chains as Representation of Context for the Detection and Correction of Malapropisms. In Christiane Fellbaum,

editor, *WordNet: An Electronic Lexical Database and Some of its Applications*. MIT press, pages 305–332.
Hobbs, Jerry. 1979. Coherence and coreference. *Cognitive Science*, 3:67–90.
Huang, Liang and David Chiang. 2005. Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing '05, pages 53–64, Stroudsburg, PA.
Ji, Yangfeng and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, MD.
Joty, Shafiq, Giuseppe Carenini, and Raymond T. Ng. 2012. A novel discriminative framework for sentence-level discourse analysis. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 904–915, Jeju Island.
Joty, Shafiq, Giuseppe Carenini, and Raymond T. Ng. 2013. Topic segmentation and labeling in asynchronous Conversations. *Journal of Artificial Intelligence Research (JAIR)*, 47:521–573.
Joty, Shafiq, Giuseppe Carenini, Raymond T. Ng, and Yashar Mehdad. 2013. Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 486–496, Sofia.
Joty, Shafiq, Francisco Guzmán, Lluís Màrquez, and Preslav Nakov. 2014. DiscoTK: Using discourse structure for machine translation evaluation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, WMT '14, pages 402–408, Baltimore, MD.
Jurafsky, Daniel and James Martin. 2008. Statistical parsing. In *Speech and Language Processing*, chapter 14. Prentice Hall.
Knight, Kevin and Jonathan Graehl. 2005. An overview of probabilistic tree transducers for natural language processing. In *Computational Linguistics and Intelligent Text Processing*, volume 3406 of *Lecture Notes in Computer Science*. Springer, Berlin Heidelberg, pages 1–24.
Knott, Alistair and Robert Dale. 1994. Using linguistic phenomena to motivate a set of coherence relations. *Discourse Processes*, 18:35–62.
Koller, Alexander, Michaela Regneri, and Stefan Thater. 2008. Regular tree grammars as a formalism for scope underspecification. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, pages 218–226, Columbus, OH.
Lafferty, John, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA.
Lazaridou, Angeliki, Ivan Titov, and Caroline Sporleder. 2013. A Bayesian model for joint unsupervised induction of sentiment, aspect and discourse representations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, Sofia.
Li, Jiwei, Rumeng Li, and Eduard Hovy. 2014. Recursive deep models for discourse parsing. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2061–2069, Doha.
Li, Sujian, Liang Wang, Ziqiang Cao, and Wenjie Li. 2014. Text-level discourse dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 25–35, Baltimore, MD.
Louis, Annie, Aravind Joshi, and Ani Nenkova. 2010. Discourse indicators for content selection in summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '10, pages 147–156, Tokyo.
Macháček, Matouš and Ondřej Bojar. 2014. Results of the WMT14 metrics shared task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, Baltimore, MD.
Magerman, David. 1995. Statistical decision-tree models for parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, ACL'95, pages 276–283, Cambridge, MA.
Mann, William and Sandra Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.
Marcu, Daniel. 1999. A decision-based approach to rhetorical parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on*

Computational Linguistics, ACL'99, pages 365–372, Morristown, NJ.
Marcu, Daniel. 2000a. The rhetorical parsing of unrestricted texts: A surface-based approach. *Computational Linguistics*, 26:395–448.
Marcu, Daniel. 2000b. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, MA.
Marcu, Daniel and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, ACL'02, pages 368–375. Philadelphia, PA.
Marcus, Mitchell, Mary Marcinkiewicz, and Beatrice Santorini. 1994. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
Martin, James, 1992. *English Text: System and Structure*. John Benjamins Publishing Company, Philadelphia/Amsterdam.
Maslennikov, Mstislav and Tat-Seng Chua. 2007. A multi-resolution framework for information extraction from free text. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 592–599, Prague.
McCallum, Andrew. 2002. MALLET: A machine learning for language toolkit. `http://mallet.cs.umass.edu`.
McCallum, Andrew, Dayne Freitag, and Fernando C. N. Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pages 591–598, San Francisco, CA.
McDonald, Ryan, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL '05, pages 91–98, Ann Arbor, MI.
McDonald, Ryan, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 523–530, Stroudsburg, PA.
Morris, Jane and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of structure of text. *Computational Linguistics*, 17(1):21–48.
Murphy, Kevin. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press. Cambridge, MA.
Pang, Bo and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, ACL '04, pages 271–278. Barcelona.
Pitler, Emily and Ani Nenkova. 2009. Using syntax to disambiguate explicit discourse connectives in text. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 13–16, Suntec.
Poole, David and Alan Mackworth, 2010. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press.
Prasad, Rashmi, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC)*, pages 2961–2968, Marrakech.
Prasad, Rashmi, Aravind Joshi, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, and Bonnie Webber. 2005. The Penn Discourse TreeBank as a resource for natural language generation. In *Proceedings of the Corpus Linguistics Workshop on Using Corpora for Natural Language Generation*, pages 25–32, Birmingham.
Regneri, Michaela, Markus Egg, and Alexander Koller. 2008. Efficient processing of underspecified discourse representations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 245–248, Columbus, OH.
Reyle, Uwe. 1993. Dealing with ambiguities by underspecification: Construction, representation and deduction. *Journal of Semantics*, 10(2):123–179.
Schapire, Robert E. and Yoram Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine Learning*, 39(2–3):135–168.
Schauer, Holger and Udo Hahn. 2001. Anaphoric cues for coherence relations. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*, RANLP '01, pages 228–234, Tzigov Chark.
Schilder, Frank. 2002. Robust discourse parsing via discourse markers, topicality

and position. *Natural Language Engineering*, 8(3):235–255.

Sha, Fei and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL-HLT'03, pages 134–141, Edmonton.

Silber, Gregory and Kathleen McCoy. 2002. Efficiently computed lexical chains as an intermediate representation for automatic text summarization. *Computational Linguistics*, 28(4):487–496.

Smith, Noah A. 2011. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool.

Socher, Richard, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013a. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia.

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, WA.

Somasundaran, S. 2010. *Discourse-Level Relations for Opinion Analysis*. Ph.D. thesis, University of Pittsburgh, PA.

Soricut, Radu and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL'03, pages 149–156, Edmonton.

Sporleder, Caroline. 2007. Manually vs. automatically labelled data in discourse relation classification. Effects of example and feature selection. *LDV Forum*, 22(1):1–20.

Sporleder, Caroline and Mirella Lapata. 2004. Automatic paragraph identification: A study across languages and domains. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, EMNLP '04, pages 72–79, Barcelona.

Sporleder, Caroline and Mirella Lapata. 2005. Discourse chunking and its application to sentence compression. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT-EMNLP'05, pages 257–264, Vancouver.

Sporleder, Caroline and Alex Lascarides. 2005. Exploiting linguistic cues to classify rhetorical relations. In *Proceedings of Recent Advances in Natural Language Processing (RANLP)*, pages 157–166, Bulgaria.

Sporleder, Caroline and Alex Lascarides. 2008. Using automatically labelled examples to classify rhetorical relations: An assessment. *Natural Language Engineering*, 14(3):369–416.

Stede, Manfred. 2004. The Potsdam Commentary Corpus. In *Proceedings of the ACL-04 Workshop on Discourse Annotation*, pages 96–102, Barcelona.

Stede, Manfred. 2011. *Discourse Processing*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool Publishers.

Subba, Rajen and Barbara Di-Eugenio. 2009. An effective discourse parser that uses rich linguistic information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT-NAACL'09, pages 566–574, Boulder, CO.

Sutton, Charles and Andrew McCallum. 2012. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373.

Sutton, Charles, Andrew McCallum, and Khashayar Rohanimanesh. 2007. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *Journal of Machine Learning Research (JMLR)*, 8:693–723.

Taboada, Maite. 2006. Discourse markers as signals (or not) of rhetorical relations. *Journal of Pragmatics*, 38(4):567–592.

Taboada, Maite, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307.

Taboada, Maite and William C. Mann. 2006. Rhetorical structure theory: Looking back and moving ahead. *Discourse Studies*, 8(3):423–459.

Teufel, Simone and Marc Moens. 2002. Summarizing scientific articles: Experiments with relevance and rhetorical

status. *Computational Linguistics*, 28(4):409–445.
Verberne, Suzan, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. 2007. Evaluating discourse-based answer extraction for why-question answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'07, pages 735–736, Amsterdam.
Vliet, Nynke and Gisela Redeker. 2011. Complex sentences as leaky units in discourse parsing. In *Proceedings of Constraints in Discourse*, pages 1–9, Agay–Saint Raphael.
Webber, B. 2004. D-LTAG: Extending lexicalized TAG to discourse. *Cognitive Science*, 28(5):751–779.
Webber, Bonnie, Andrei Popescu-Belis, Katja Markert, and Jörg Tiedemann, editors. 2013. *Proceedings of the Workshop on Discourse in Machine Translation*. ACL, Sofia.
Wick, Michael, Khashayar Rohanimanesh, Kedare Bellare, Aron Culotta, and Andrew McCallum. 2011. Samplerank: Training factor graphs with atomic gradients. In *Proceedings of the 28th International Conference on Machine Learning*, ICML'11, pages 777–784. Bellevue, WA.
Wolf, Florian and Edward Gibson. 2005. Representing discourse coherence: A corpus-based study. *Computational Linguistics*, 31:249–288.
Zhang, Yuan, Tao Lei, Regina Barzilay, Tommi Jaakkola, and Amir Globerson. 2014. Steps to excellence: Simple inference with refined scoring of dependency trees. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 197–207, Baltimore, MD.