Diploma Thesis in Computational Linguistics

Rhetorical Analysis with Rich-Feature Support Vector Models

David Reitter

February 2003

Professor Manfred Stede
Supervisor

Professor Deb Roy
Reader

University of Potsdam

Contents

1. Introduction

1.1. An attempt to motivate rhetorical analysis

Almost every author of a text has a mission: he or she wants to make an argument in support of an opinion or of information given, giving backgrounds, reasons, and discussing seemingly incompatible observations about the world. Hardly anyone tries to achieve this mission by just writing down a collection of singular statements. What readers would like to read, is a text.

Isn't a text merely a collection of statements? It is more than that! One of the central properties that distinguishes a *text* from a collection of random sentences is that one part of text builds on the other. A measure for this trait is *coherence*. The *Purdue University Online Writing Lab* gives an informal definition of coherence:

> When sentences, ideas, and details fit together clearly, readers can follow along easily, and the writing is coherent. The ideas tie together smoothly and clearly.

So, pieces of coherent text are connected. Can we define a typology of these connections? Yes, we can. Quite a few statements *elaborate* on the one they follow in a text. Other statements are in a *contrastive* relationship, yet others are connected in a time-related fashion, describing events that happened after each other. Some theories of rhetorical structure define even 300 different rhetorical relationships.

It's not only the case for single sentences that should form an argumentative chain in a well written text. Surprisingly, the same kind of rhetorical relations that can hold between small pieces of text, also occur between larger spans of text. Each span can be divided into two or more smaller spans, which are connected by rhetorical relations. In linguistic terms, the rhetorical analysis of a text document forms a tree.

In a nutshell, this thesis is about how to recognize the types of relations that connect small chunks of coherent text. The motivation behind this is that in almost all cases it is imperative to recognize the relationship between the parts in order to *understand* a text. Luckily, many people who speak or write have, in the early stages of their education, learned to structure their thoughts.[1] In order to facilitate communication, relations between the text spans are marked by means of phrases like *but* or *in order to*, and, in written text, by a colon. But it wouldn't be natural language if these *cue* phrases could easily be linked to a rhetorical relation. They are highly ambiguous. Nevertheless we can use a range of cues to make a good guess about the structure of rhetorical relations.

Natural language technology has been devoting an increasing amount of work to rhetorical analysis. There are good reasons for this interest. Rhetorical structure influences both natural language generation and analysis. Rhetorical parsing is an important precursor to

[1] Those who didn't get a job in legislation, as law makers produce one of the very few kinds of text that don't show rhetorical structure.

selecting and evaluating information, for example in summarization or semantic indexing. In speech synthesis, rhetorical information would be useful to determine prosody and pauses.

Linguistic accounts have the luxury to assume the availability of world knowledge and inference for rhetorical analysis. Automated systems resort, with quite some success, to a combination of little linguistic knowledge and a lot of shallow processing. How far can this carry us in rhetorical analysis? How can such an analyzer be built and optimized? Which rhetorical clues are hidden in text, especially apart from overtly formulated connectives? How much can the clues help to improve recognition performance?

A contribution to answer these questions will not only provide better document analysis techniques; it might also clarify our intuitions about style and the rhetorical tricks in natural language communication.

1.2. Accounts for rhetorical structure

Besides the fairly obvious macro-structure in the organization of texts in a work[2], and besides the well-understood syntactic and sub-syntactic structure, text generally shows rhetorical phenomena. *Discourse structure* mirrors coherence phenomena in texts, which describe how single semantic propositions or small spans of text are interrelated. Well structured text in this sense can make a concise argument.

One of the most prominent models for discourse is *Rhetorical Structure Theory* (Mann & Thompson, 1988, RST). This theory defines a set of relations that can hold between given spans of text. Common relations include ELABORATION(A, B), which is defined as a text span B which gives additional information regarding the facts presented in span A, and CONTRAST, which is defined as the contents of two or more text spans being knowingly presented as incompatible.

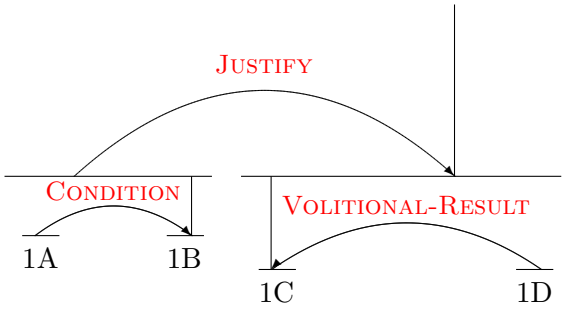
Sandra Thompson and Bill Mann chose their rhetorical relations with a functional goal in mind: relations should reflect coherence phenomena. Their definitions are not constrained to refer to properties from classical linguistic or computational processing levels. Semantic and world-knowledge based constraints interact with the presumed writer's or speaker's intentions.

Nuclei and Satellites: In RST, each text span takes on one of two *roles* in a relation: it may be a *nucleus* or a *satellite*. Nuclei are considered essential to the understanding of the text, satellites contribute additional information. The relations that hold between spans may be either *paratactic* or *hypotactic*. Paratactic relations connect two or more equally weighted spans of text and assign the same role to each of them. Hypotactic relations hold between one nucleus span and one satellite span (Figure 1.1).

To facilitate the distinction between the two roles of spans for human annotators, Carlson & Marcu (2001) give the following explanations:

> **Deletion test.** When a satellite of a relation is deleted, the segment that is left, i.e., the nucleus, can still perform the same function in the text, although

[2] An example for this macro-structure can be seen in this thesis, which consists of chapters, sections and subsections, footnotes and headlines.



[And if the truck driver's just don't want to stick to the speed limits,]^{1A} [noise and resentments are guaranteed.]^{1B} [It is therefore legitimate to ask for proper roads and speed checks.]^{1C} [And the city officials have signaled to support local citizens.]^{1D} (maz_{5007e})

Figure 1.1.: Example for an analysis with hypotactic relations in RST.

it may be somewhat weaker. When the nucleus is deleted, the segment that is left is much less coherent.

Replacement test. Unlike the nucleus, a satellite can be replaced with different information without altering the function of the segment.

These rhetorical relations can hold between adjacent text spans of any length. Spans connected by a relation are subject to a new relation, and so on. This way, the analysis of a coherent text is a tree structure. (For a more elaborate example, see Figure 4.1, page 24.)

Rhetorical relations can be signaled in a text by various means. The most striking type of signal is the *cue phrase* (or: discourse marker, connective). Among the common English cue phrases are *However, thus, so, but, on the one/other hand*. Usually, these cue phrases signal the rhetorical relation of the text span they occur in one of its adjacent text spans. *Punctuation* and text layout play a similar role. In this thesis, I examine other properties of text that can give hints at rhetorical structure. Virtually all of these clues are highly ambiguous: *as* may signal a temporal or a causal relation, *even though* may be used to make a CONCESSION or to discuss a CONTRAST. It should be noted that while these clues facilitate the understanding of a text, coherence phenomena ultimately depend on the semantics of the text and its referential structure.

1.3. Goals, methodology, and structure

The issues this thesis wishes to contribute are:

- demonstrating and evaluating in rhetorical analysis the use of a novel machine learning framework: Support vector machines, which often outperform other algorithms in difficult learning tasks.
- application of classifiers to the tasks of rhetorical relation assignment, nuclearity assignment and discussing a parsing algorithm based on these classifiers
- establishing a quantitative measure for several surface properties used in rhetorical analysis
- collecting a new rhetorical corpus

- formalizing under-specified rhetorical structure.

From a point of methodology, I will motivate the use of shallow features in rhetorical analysis informally, but linguistically. Features and algorithms are evaluated quantitatively; they were optimized using iterative evaluation. Human judgement of trained experts (annotators) constitutes the gold standard for evaluation. The methods used in the implementation ensure reusability and extensibility; they are shortly documented using the state-of-the-art methodologies established in industrial practice.

Evaluation does not merely rely on the self-collected data. Besides the German language corpus, we evaluate our analysis approach on a large English corpus of newspaper articles. Examples from both corpora appear throughout this thesis. Evaluation focuses on the detection of rhetorical relations. This is a scope-limiting decision that does not hinder us from a discussion of further analysis methods.

In the following, I will establish a background picture discussing related work in text linguistics. Other work that is pertinent to the aspects of the thesis are discussed in the appropriate chapters. Chapter 3 describes the collection of data, which establishes both a gold standard for evaluation and a training corpus for the machine-learning algorithms in the approach. With this data in mind, we can find a suitable and universal representation, URML, which is discussed in Chapter 4. This representation makes several important assumptions about rhetorical structure, but provides the flexibility needed to apply several annotation algorithms to the texts to be analyzed. The algorithms are discussed in Chapter 5, which also introduces the reader – assuming only little background in corpus linguistics and statistical theory – to the machine learning approach taken. The implementation of the different tools is sketched out in Chapter 6; it is used in a mostly quantitative evaluation and analysis of the central methods in Chapter 7.

Acknowledgements

I would like to thank my supervisor Prof Manfred Stede for his extremely valueable support and for providing the necessary funding for corpus collection; creating the corpus was a joint effort with him. I am grateful to Prof Deb Roy, whose comments durching the final phase of research and write-up. Carson Reynolds gave the valuable hints on coherence and syntax only a native speaker can contribute. Bianca Schön and Renee Hall did a great job proof-reading the draft. I would also extend my thanks to Cindy Ernst and Antje Sauermann for their tedious work annotatating the Potsdam Corpus, and to the editors of the *Märkische Allgemeine Zeitung* for giving their permission to download and re-publish their data.

Part of this work was funded by a fellowship from MIT's Media Lab Europe, Dublin and through the EU grant IST-2001-38685. Corpus collection funding was provided by the University of Potsdam.

2. Related Work, Important Issues

Our approach is informed by various fields, among them rhetorical analysis, parsing and machine learning. The analysis algorithm draws ideas from statistical parsing approaches using classifiers; we discuss an example for this class of algorithms (Ratnaparkhi, 1998). The classification models use support vector machines (Vapnik, 1995). The most notable work in rhetorical corpus collection effort as of this writing is Carlson et al. (2001). Furthermore, the work of Daniel Marcu has contributed significantly to the state of the art. He, among others, develops a formal system to describe rhetorical structure and implements a rhetorical parser based on cue phrases and some other cues. Simon Corston-Oliver uses an even broader variety of clues for relations. These and more works will be discussed in detail in the subsequent chapters. In the following, I will sum up the work surrounding rhetorical structure, which is the central aspect of the thesis.

2.1. Mann & Thompson's Rhetorical Structure Theory

The groundbreaking work of Sandra Thompson and Bill Mann *Rhetorical Structure Theory* (Mann & Thompson, 1988, RST) coined a now widely adopted theory of rhetorical structure during the 80's. RST has received much attention in the field of natural language generation. Here, rhetorical structure provides the basis for inserting connectives (Scott & Sieckenius de Souza, 1990; Rösner & Stede, 1992; Hovy, 1993). RST has been complemented by more dynamic views of text as Grosz & Sidner (1986); Grosz et al. (1995) which distinguish intentional and linguistic structure and, as also in Kamp & Reyle (1993), a state of saliency.

2.2. Grosz & Sidner's theory

Grosz & Sidner (1986) describe the relationships of three components of discourse structure. *Linguistic structure* models the segmentation of text, which consists of smaller units, utterances. *Intentional structure* consists of discourse segment purposes and defines relations between them. *Attentional state* accounts for salient objects, properties and relations at any given state in the discourse. The subsequently developed *Centering* model (Grosz et al., 1995) provides an account for the interaction between linguistic structure and attentional state. In this theory, Grosz, Joshi & Weinstein (1995) give a set of constraints on text. It is stated that the more a discourse follows the centering constaints, the higher its coherence will be. Thus, "inference load placed upon the hearer will decrease." In these constraints, the authors predict that the realization of referring expressions (e.g. pronouns) is governed by discourse factors, in particular the local center of discourse. Continuation of this center between discourse units is generally preferred over a shift of the center. Thus, continuation versus a shift of center reflects in lexicalization of referents. Thus, we can use referring expressions as clue to attachment decisions, i.e. to determine whether a text span is rhetor-

ically connected to its preceding or the following span. Also, some rhetorical relations from RST imply a shift of center and should thus depend on referring expressions.

2.3. Formalizing the semantics of RST

Past approaches to motivating relations are based on rich semantic information (Hobbs, 1979; Fukumoto & Tsujii, 1994). Such theories describe rhetorical relations by breaking them down to:

- a set of situational properties, referred to at two points of time in the discourse,
- atomic predicates that operate on the properties and serve as contraints.

Some of the situational properties would be:

- reader's attitude toward the semantic content of a segment, it's acceptance or dismissal
- reader's or writer's attitude toward the process of presentation of a segment
- reader's state of informedness

Such an analysis formalizes Rhetorical Structure Theory very much according to the original definitions of rhetorical relations. The situational properties give an algebraic account for some of the relations.

Rhetorical relations as defined by Mann & Thompson (1988) refer to levels of discourse. There are intentional, argumentative properties, e.g.:

- Evidence: a writer presents the satellite in order to increase the reader's belief presented in the nucleus,
- Antithesis: "Nucleus: ideas favored by the author. Satellite: ideas disfavored by the author",
- Concession: "Nucleus: situation affirmed by author. Satellite: situation which is apparently inconsistent but also affirmed by author."

Factual information, on the other hand, is referred to in the Cause relation: "a situation causes the other one". This relation depends on semantic relationships of the semantic content presented in each of the spans rather than their discoursive function. Similarly, Mann & Thompson (1988) define Volitional Result: "Nucleus: a situation, Satellite: another situation which is caused by that one, by someone's deliberate action".

This shows that Mann & Thompson (1988) account for more than mere "rhetorical" phenomena. The theory does not strictly distinguish among them; instead, it describes relationships on a semantic level. These interact with rhetorical structure. For instance, the four (Non-)Volitional Cause/Result relations always designate discoursive function with the nucleus being a center of discourse. Indeed, the authors suggest to classify relations as being primarily concerned with "subject matter" or as mainly "presentational." This would need to be included in any information-based formalization of rhetorical relations.

Formalization of this kind is practically applicable, as long as the text that is analyzed, could be formalized in the same framework. This, however, is at the present stage, rather unlikely. For this reason, I will follow a different methodology, which is based on shallow properties that can be automatically observed in the text.

2.4. Marcu (1999): Unifying RST and Grosz & Sidner's theory

As discussed before, Grosz et al. (1995) addresses the intentions of a writer or speaker in its *intentional structure*. Expressions on this level of analysis refer not only to semantic information contained in minimal discourse units. It is very often concerned with semantic propositions that result from rhetorical relations between spans of text (Marcu, 1999).

[John wanted to play squash with Janet,]^{2A} [but he also wanted to have dinner with Suzanne.]^{2B} [He went crazy.]^{2C} [1]

According to the Grosz, Joshi & Weinstein (1995) theory, the primary intention of the span [2A,2B] is: *Writer wants reader to believe that John wanted to do two incompatible things.* This directly results from a rhetorical relation from Thompson & Mann's theory. CONTRAST holds between 2A and 2B.

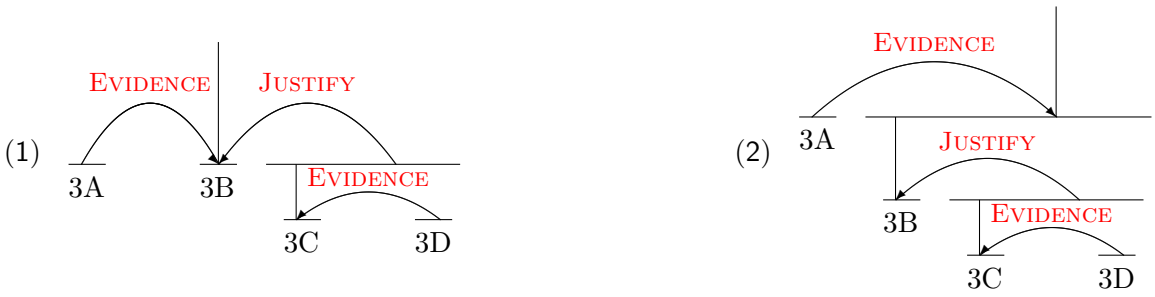
Marcu (1999) develops a formal framework that marries both theories. It operates with a data structure of four kinds of information on each span of text. The *status* in the predicate $S(l, h, status)$ describes its role in a rhetorical relation. It may be NUCLEUS, SATELLITE or NONE. By axiom, each span has exactly one piece of status information. The *relation name* in the predicate $T(l, h, relationname)$ describes the rhetorical relation that holds between the immediate descendants of the span. Its value is a member of the finite set of rhetorical roles. Each span has a set of *salient units*. These units, defined by the predicate $P(l, h, unitname)$ are promoted up in the tree. The last information describes the *primary intention* of the given spam in a predicate $I(l, h, intention)$. The intention is expressed in terms of a function f_I on a relation and a span. As axomatization describes, this intention is unique for each span.

As l, h are seen as index values, Marcu makes use of traditional operators for numbers and predicate logic. Thirteen axioms describe all constraints to operate on the system. Within this algebra, it is possible to deduct all allowed rhetorical-intentional tree analyses, provided the rhetorical relations that hold between spans are known and the oracle function giving the intention of the spans is defined. Marcu's approach is a constraint-based, symbolic system. No probabilistic functions are defined.

Formalization of discourse theories seems to be a major step in the automatic analysis of texts. In such a framework, empirical observations may be described and added to a constraint-based analysis system. The crucial step is to define a semantically oriented system for the intention function f_I and to find constraints that govern possible clues for the choice of relations.

The approach depicted in Chapter 5 uses a feature-rich classification algorithm to derive fuzzy sets of relations between single spans. A modified version of Marcu's framework might be able to integrate well into our approach and derive valid trees in the light of Centering theory.

[1] Example from Marcu (1999).



[Also, there can be no social boundaries to the seldom-publicized topic of a community-level obligation to social welfare.][3A] [The moral responsibility is ignored.][3B] [The question is, whether the disadvantaged group of the homeless is put away even more in their new quarters in Kyritz.][3C] [After all, their home is Wittstock.][3D] (maz14071e)

Figure 2.1.: Schemas vs. binary nodes. Text translated from original.

2.5. Issues in Rhetorical Structure Theory

2.5.1. Is a single tree adequate?

The tree-structured approach of RST suggests that there is – even though several analyses may be drawn from a text – only one structure that is faithful to the writer's intentions. It may however be questioned whether there is indeed such a "primary rhetorical intention" (Grosz & Sidner, 1986). Furthermore, when it comes to automatic rhetorical analysis, a program will have difficulties deriving the one and only correct structure. Our own approach (to appear) tries to derive a single interpretation that *most likely* matches the writer's intention. To meet the challenge of rhetorical structure, the algorithm can also output several likely interpretations, along with their probabilistic scores.

2.5.2. Binary trees?

The original layout of the underlying theory, described in Mann & Thompson (1988), forsees the existence of binary hypotactic relations that may share a common nucleus. Each nucleus and all its satellites form the instantiation of a *schema*. Subsequent applications of the theory, however, mainly made use of the idea of binary trees (Cristea & Webber, 1997; Marcu, 2000, and others). These contain a nucleus and exactly one satellite.

Translation from trees with multiple branches to binary ones is possible. In Figure 2.1, analysis (1) can be represented as shown in analysis (2). One reason lies in the fact that the nucleus contributes ("promotes") the essential meaning in comparison to its satellite. Consequently, if a relation holds between span A with several segments and span B, the same (or, a similar) relation holds between the nucleus of span A and span B (Marcu, 2000, compositionality criterion). Empirical evidence we gained in the collection of our newspaper corpus supports this view: in the intuition of the annotators, in no case was the non-nucleus-sharing, tree-structured representation representationally weaker than the nucleus-sharing view. The choice between the two possibilities for a two-satellite-one-nucleus schema was usually made from referential clues. Also, in the RST Discourse Treebank (Carlson et al.,

2001), we found that only in 9 out of 41856 nuclei and satellites (or 20529 relation schemas) there were satellites relating to a common nucleus with a different hypotactic relation.

As a consequence, we implement binary relations in our representation language URML, but no RST schemas.

2.5.3. How can relations be motivated?

The idea of relations to hold between text spans has become a paradigm in text linguistics. Also, rhetorical tree structures are a common observation in texts. Regarding the relations, Rhetorical Structure Theory never claimed that this set be closed. Consequently, existing relation definitions have been modified, new ones have been added since RST was first described. More extensive sets do not necessarily subsume the others. Hovy (1990) counts 350 relations and proposes a taxonomy of relations with three top-branching ones Elaboration, Enhancement, and Extension.

However, the semantic nuances of relations are often fine-grained, so that it may seem impossible to define a finite set of relations (Grosz & Sidner, 1986). Discoursive practice may provide reasons in favor or against a new relation; linguistically observable signals and tests – such as tests for cue phrases and for substitutability – might provide clues to psycholinguistic reality (Knott & Dale, 1994; Knott, 1996). Hereby, the notion of linguistic signals is not limited to cue words or syntactic phrases. Prosody plays a role, and gestures or body language (Cassell et al., 2001).

2.5.4. What is a terminal segment?

Analogous to the relation definitions, we could pursue a semantically driven, informed definition of terminal segments: "A *minimal discourse unit* is whatever span of text has a unique rhetorical relation to another one." It seems obvious that one can perform rhetorical analysis at different granularities. While the original version of RST builds on contiguous spans of text, many languages – in particular verb-final ones – will allow for the embedding of phrases in verbal arguments or adjuncts that have the semantics of a proposition and can act as rhetorical unit themselves. This leads us to discontiguous segments.

We can also define discourse units in terms of syntactic categories: every clause (Longacre, 1983) or complementizer phrase, or every sentence (Polanyi, 1988) could be a segment. Most convincingly, minimal discourse units are created intentionally and signaled in various ways (Grosz & Sidner, 1986). In discourse-semantic terms, they are "contextually indexed representation[s] of information conveyed by a semiotic gesture, asserting a single state of affairs or partial state of affairs in a discourse world" (Polanyi, 1996). Most of the semantically oriented definitions allow for segmentations on the sub-clausal level. However, as de Smedt et al. (1993) note, most connectives on this level are determined by syntactic constraints rather than merely rhetorical ones. Grote et al. (1997); Schauer & Hahn (2000) argue that adjuncts (mostly prepositional phrases) can, in certain cases, be analyzed as discourse units.

3. Collecting the Potsdam Corpus

3.1. Introduction

Annotated corpus data is a sought-after resource. This is not a surprising fact. Rhetorical annotation is an extremely labor-intensive process. Annotators need extensive training to master the sometimes subtle differences in relations. They need to gain a full understanding of each document, and even then their choices may be ambiguous.

The need for quantitative evaluation of research prototypes and the increasing popularity of machine learning hit rhetorical analysis like many other fields in computational linguistics. Rhetorical corpus data is rare. We therefore created a small German corpus. In the following I will explain the design decisions involved, as well as some of the experiences made.

3.2. Previous work

The RST annotated collection of 385 newspaper articles presented by Carlson et al. (2001) is, to the author's knowledge, the most extensive RST corpus. It contains almost 21,800 minimal discourse units in documents of varying sizes. The corpus was quality-controlled by tedious training of the annotators and partial blind, automated cross-validation. The texts are a subset of the Penn Treebank, so syntactic annotations are available.

The creators of the corpus decided to assume clauses as minimal discourse units. They were determined according to lexical and syntactic clues. Any phrase with a verb is to be a minimal discourse unit with the exemption of subjects, objects or any (other) complements of a main verb.

[**Although** Mr. Freeman is retiring,] [he will continue to work as a consultant for American Express on a project basis.] (wsj_{1317})

["The company's current management found itself **locked into this**," he said.][1] (wsj_{1103})

In addition to marking contiguous minimal discourse units, Carlson, Marcu & Okurowski (2001) define embedded discourse units. These are relative clauses, nominal postmodifiers and clauses that are surrounded by other discourse units.

[The Bush Administration,] [trying to blunt growing demands from Western Europe for a relaxation of controls on exports to the Soviet bloc,] [is questioning...] (wsj_{2326})[2]

[1] Examples taken from Carlson et al. (2001), emphasis in original.
[2] Bracket annotation simplified.

The texts were manually segmented by only two of the annotators, so that segmentation is fairly consistent throughout the corpus.

The inventory of relations consisted of 53 mononuclear and 25 multinuclear relations. However, during annotation, these relations were subsumed by 16 (more general) relation classes in order to facilitate the work of the annotators.

According to Carlson et al. (2001), professional language analysts were employed as annotators. They received extensive training. A written manual (Carlson & Marcu, 2001) ensures a common RST standard. After an initial phase of tagging 100 documents, annotation rules were refined and inter-annotator agreement was measured.

As an agreement measure, Kappa coefficients were used (Siegel & Castellan, 1988; Marcu et al., 1999). In the complete corpus, 53 documents were double-tagged. The Kappa values stated indicate an increase in agreement over time, reaching values that reflect "very high agreement" and "good agreement". However, the last values available were calculated on the basis of only 5 double-tagged documents.

3.3. Choosing the texts for the corpus

We chose a series of German language news commentaries, published recently in a local newspaper, the *Märkische Allgemeine Zeitung*. Most of the texts concern local issues, some address national and international issues. Average document length is 28.6 sentences. Several general design decisions provided a context for the choice of new commentaries. Texts should:

- expose a clear argumentative structure. Pre-existing knowledge about the writer's intentions should not be needed in understanding and analyzing the texts. The same applies to contextual factual knowledge. The text genre of editorials exposes an argumentative structure, while, as journalistic principle, the writer should not bear any intentions beyond those conclusions that result from the facts considered.

- be short enough to allow annotators to quickly gain a good understanding of their overall structure of a document and their argumentative goals. Yet, they should be long enough to expose the kinds of relations that possibly hold between larger spans of texts.[3]

- be stylistically well-written, in terms of a match between rhetorical signals and the intended argumentative structure,

- belong to the same genre of text. While contrastive studies of rhetorical signals or even single instances of relations are an interesting application of a corpus, we need enough texts (per genre) to evaluate machine-learning approaches,

[3] Readers of the *Märkische Allgemeine Zeitung* live in the small city of Potsdam and the surrounding rural areas. Following from the fact that Germany's most vibrant city Berlin is close-by, the paper does not openly compete with the influential daily newspapers published in Germany's captial. Rather, they address local issues for 'the common man', whom we would not expect to possess significant background-knowledge. This has shown to mark another item on our wish list as 'checked': The documents are easily and quickly understandable.

- be written in a language that complements efforts in other languages. The corpus is in German — to my knowledge, this is the first corpus collection effort in this language.

All of these criteria are fulfilled by the short German language newspaper editorials, which were written by professional journalists and selected for genre.

3.4. Text preparation and segmentation

All documents were extracted using a web crawler, which accessed the newspaper's online edition, stripped away layout-specific markup from the HTML code and conserved exact source information including some meta-data regarding authorship, date, newspaper section etc. .

We decided to automatically segment the documents into minimal discourse units, based on shallow syntactic features. Clause borders are recognized by punctuation and finite verbs. This segmentation is used as input for human annotators, who may not introduce new segment borders, but withdraw segmentations by using a JOINT relation. This methodology maintains a clear standard for segmentation and allows us to examine pure rhetorical relations and rhetorical structure.[4]

3.5. Training the annotators

Two advanced linguistics students were hired full-time to work through as many texts as possible. Prior to that, the annotators received training, discussing the original relation definitions as given by Mann & Thompson (1988). After 30 texts were annotated, annotations were reviewed and, after a new training session, re-annotated.

3.6. Annotation process

Annotation was carried out using RSTTool 3.1 (O'Donnell, 2000, see Figure 3.1), a software that provides a graphical and efficient user-interface to create RST-like structures.

We annotated the corpus with 19 relations. They are subsumed by 10 more general relations according to semantic properties in their definitions that they share. The following hypotactic (nucleus-satellite) were used in the corpus[5] :

[4] Marcu (2000) identifies segments using cue phrases. The automatic segment detection, however, weakens his system significantly: rhetorical relation recognition performance drops from 60 percent / 63 percent (recall/precision) to 17 percent / 36 percent, when the automatic segmenter is used. (Data from (Marcu, 2000, p. 170) for Wall Street Journal corpus as test set and a combined corpus as training set. Segment recognition performance is 25.1/79.6 percent.)

[5] See Mann (1999); Mann & Thompson (1988) for a definition of these relations.

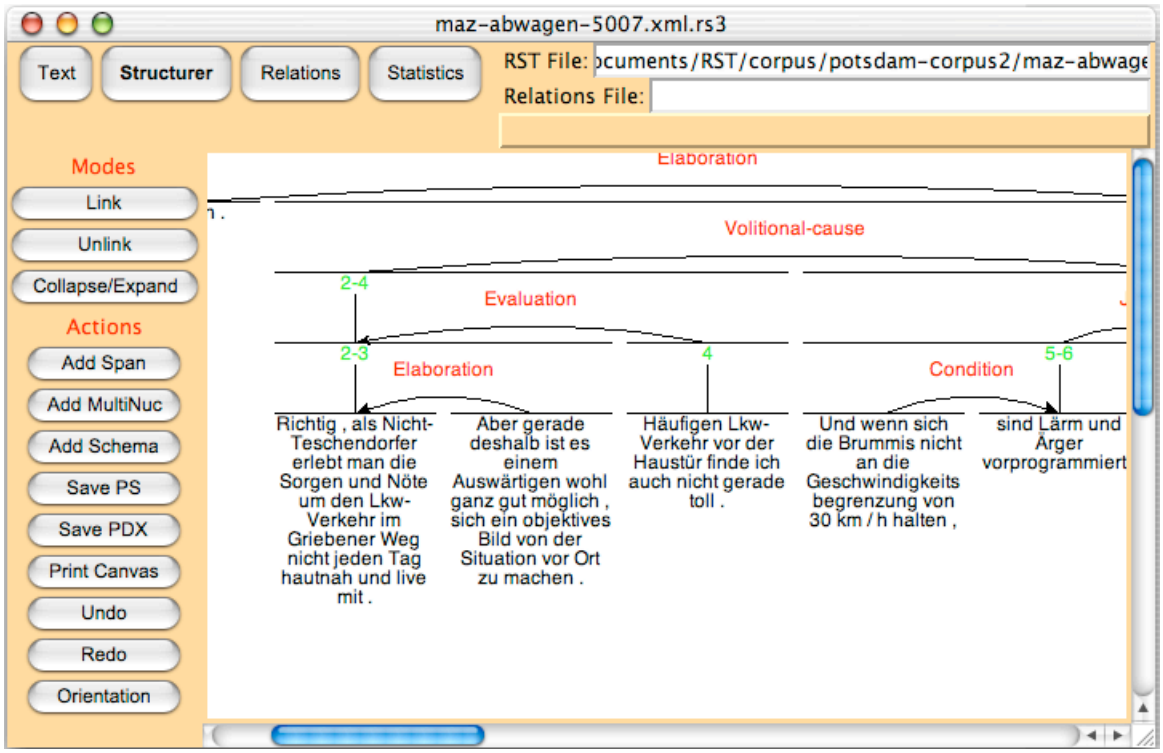


Figure 3.1.: Screenshot of RSTTool 3.0.

Hypotactic relations

Cause:	Justify, Nonvolitional-Cause, Volitional-Cause
Conclusive:	Evaluation, Solutionhood, Summary
Evidence	
Framework:	Background, Circumstance, Enablement
Preparation	
Result:	Nonvolitional-Result, Volitional-Result
Specification:	Elaboration, Restatement

Paratactic relations

Conjunction	
Contrastive:	Contrast
Sequential:	List, Sequence

Two of the relations offered to the annotators were not used. We assume that Summary is not frequent in short newspaper commentaries; remaining Summary relations were most probably marked as Restatement. Antithesis was most likely confused with its paratactic pendant, Contrast. Table 3.1 shows the frequency of the relations in the corpus.

The annotators completed 173 documents in about 14 person-days. Then, they switched datasets and cross-validated their work. We decided in favor of a collaborative annotation effort among the annotators and against a full (blind) cross-validation, where both annotators would annotate the same texts, and then, an automatic inter-annotator agreement

Table 3.1.: Relation distribution in the Potsdam Corpus

Relation	Frequency	Portion
Background	157	6.6%
Circumstance	35	1.5%
Concession	125	5.2%
Condition	47	2%
Conjunction	1	0%
Contrast	54	2.3%
Elaboration	738	30.9%
Enablement	2	0.1%
Evaluation	268	11.2%
Evidence	266	11.1%
Interpretation	3	0.1%
Joint	48	2%
Justify	19	0.8%
List	167	7%
Nonvolitional-cause	122	5.1%
Nonvolitional-result	54	2.3%
Preparation	161	6.7%
Restatement	4	0.2%
Sequence	27	1.1%
Solutionhood	6	0.3%
Summary	1	0%
Volitional-cause	48	2%
Volitional-result	34	1.4%

measurement would be applied. This was a trade-off decision between quality and quantity.

3.7. Conclusion and further work

Our decision to partially sacrifice automatic inter-annotator validation seems justified in the light of a first-time, small-scale corpus-collection effort. We see the corpus as basis for further annotation rather than as a fixed gold standard that will never be augmented. It also serves an intended purpose for corpus usage: automatic extraction of examples and of rhetorical signals, which are then manually examined (Berger et al., 2002). It has already been used in other efforts related to discourse analysis. Further work should include a thorough correction and cross-validation of the annotations.

4. URML: An Underspecified Representation of Rhetorical Analyses

4.1. Introduction

The demand for annotated linguistic corpora rises steadily. Recently, work has begun on providing data that is annotated not only on the sentence level but also on the discourse level. In particular, 'rhetorical annotation' has turned out to be important for applications such as automatic summarization. With the growing importance of machine learning and parsing algorithms that detect rhetorical structure and evaluate the results of manual or automatic annotation, it becomes clear that corpus data should be readable for both humans and machines.

In the following, we discuss design considerations for an XML-based rhetorical annotation format that is extensible and provides room for underspecification. This is a key feature for two reasons:

- Human annotators often find it difficult to make a clear decision on either a specific relation or the length of the spans (e.g., in the case of sentences starting with a conjunctive adverbial: *On the other hand, ...*). Thus it should be possible to leave such a matter open and represent it accordingly.
- For automatic rhetorical analysis, the problem above is much more pressing. Rather than enforcing a decision all the time, it is desirable for a parser to leave some aspects underspecified, represent this clearly, and possibly have additional components making a choice later on the basis of additional knowledge. More generally, underspecification allows for *incremental* rhetorical analysis based on sound representations.

We assume a theory of rhetorical description along the lines of Mann & Thompson (1988), which assigns relations between adjacent spans of text and recursively builds up a tree. We have applied our format to both manual and experimental automatic analyses of a new corpus of German newspaper texts. The format is open to extension and specialization, e.g. to enable multi-modal applications. In general, we think that the emergence of a standard for rhetorical annotation will be instrumental for comparing analyses, and obviously for training stochastic or machine learning algorithms.

Abridged version of this chapter, co-authored with Manfred Stede, to appear (Reitter & Stede, to appear 2003)

4.2. Previous work

Schilder (2002) formalizes a symbolic underspecification scheme for rhetorical structure. His relations connect Segmented Discourse Representation Structures; he explicitly states immediate dominance, general dominance, precedence and equivalence of text spans. Similar to our approach, a set of relations is given for the whole document, and for a specific pair of segments, the set of relations may be constrained during one of the analysis steps. His symbolic system is targeted at an analysis architecture based on cue phrases and a topicality measure that both constrain the rhetorical structure. In contrast, our architecture defines a serialized, XML based intermediate format that allows for the exchange of corpus data and for an *incremental* annotation of cues and rhetorical structure. Other representation efforts include Rehm (1998), who uses an SGML syntax to identify rhetorical cues. O'Donnell (2000) writes annotation data from single documents in several SGML based formats that include no underspecification. Other than this format, our representation integrates various layers of annotation, as there seems to be a consensus that there is no single type of rhetorical signal to be considered before a decision about rhetorical relations can be made (Marcu, 1997; Corston-Oliver, 1998b).

Other representations aimed at corpora (XCES: Ide et al. (2000), TUSNELDA: Kallmeyer & Wagner (2000), Text Encoding Initiative) define elaborate underspecified annotation schemes that focus on text-organization (e.g. dialog turns, paragraphs, semantic tagging and document-related meta data. As for the latter, the most notable standardization attempt is the Resource Description Framework (RDF), providing a general XML-based syntax to link arbitrary meta-data with well-defined scope. Dublin Core represents a taxonomy of meta information that can be used to instantiate RDF.

4.3. Rhetorical annotation in URML

4.3.1. Basics

Our representation format URML (Underspecified rhetorical markup language) allows for a free definition of dependencies among text spans. It clearly separates:

- Symbolic system and, in the document, an inventory of relation instances. A document annotation contains a set of relations that can be found in the data. Similar to a parse forest in syntactic parsing (Billot & Lang, 1989), it may state ambiguous relations. Some may even be incompatible according to the axioms that defined a well-formed rhetorical analysis.[1] This is defined by the format; all relation instances may be represented using URML.

- Axioms and, in the document, well-formed tree analyses of a text. It is up to the client software to define restrictions that hold for well-formed analyses. A well-formed

[1] In URML, only one relation may constitute a satellite or a nucleus of a higher-level relation. This is a result from the experiences made in corpus-collection and the examination of the English LDC corpus (see sections 4.4.4 and 2.5.2). If, in contrast to the aforementioned structural property, multiple relations are to be combined in a schema, we propose a paratactic relation of `type="joint"` An additional tag was not introduced in favor of simplicity.

analysis contains a subset of the relations defined in the relation set of the document. This well-formed analysis can also be represented and identified as such in URML.

We chose XML as underlying formalism in order to maximise re-usabilty. A "document-type definition" grammar was defined to describe the format of documents, rhetorical and morpho-syntactic annotations.[2]

In URML, all documents are contained in one single file. This facilitates automatic handling, since the filesystem is not involved in retrieving parts of the corpus. To make use of the XML-based data, standard XML libraries are available for all common implementation formalisms.

The set of relations (ELABORATION, CONCESSION) are declared once for all documents (Section 4.3.8, <reltypes>). Individual documents consist of minimal discourse units (<text>), followed by relation nodes (<analysis>) explained below.

When analysis information is added to the raw data, we want to preserve the original information wherever possible (especially in the light of the incremental rhetorical parsing we are developing; see below). Thus, all information used in the analysis process is stored in the corpus in a persistent fashion.

4.3.2. Meta-data

The possibility to trace partial analyses of documents is valuable, especially in large corpora. For example, when a manually annotated rhetorical relation is to be clarified later, there should be a way to identify the annotator. Similarly, documents may get corrupted due to bugs in an annotation tool. Obviously, a source identifier is needed.

We define a rather simple <info> tag, giving source information for documents and annotator information for analyses.

```
<analysis id="maz3379.a.1" scheme="interpretation">
 <info>
   <editor job="annotate" date="18.02.02">
           Antje Sauermann
   </editor>
   <editor job="revised" date="20.09.02">
           David Reitter
      <note>revised result/cause
                  nuclearity
      </note>
   </editor>
 </info>
...
```

4.3.3. Representing tree structure in XML

One crucial decision in syntax design was whether annotations should be coded as on-the-spot markup within the text or as relation nodes with referential indices. The first is easy to read (analogous to the one used in a LISP-style format in Carlson et al. (2001)). For example:

[2]The complete document type defintion (DTD) for URML is available from our web site, http://www.ling.uni-potsdam.de/cl/rst/

```
<concession>
  <satellite>Admittedly, ...</satellite>
  <nucleus>
    <contrast>
      <nucleus>However, some ...</nucleus>
      <nucleus>others ...</nucleus>
    </contrast>
  </nucleus>
</concession>
```

This format demands a fully specified, unambiguous single tree structure and encodes the underlying relation set in the document grammar (DTD). In XML, this may be desirable when generic editors are used, because they restrict annotators to comply with the DTD, thus with theoretic claims. All processing modules, however, are also determined to follow the fixed syntax. Changes in theoretic assumptions, such as the introduction of discontinuous constituents or several rhetorical analysis layers, inevitably lead to a chain of modifications in the system, even if the changes concern only concern one analysis layer.

Therefore, in our format every node of a discourse tree represents one relation: either a hypotactic one (one nucleus, one satellite) or a paratactic one (several nuclei). Nodes are indexed and reference each other to express references to the according text spans. In the following example, the IDs 1, 2, 3 refer to minimal discourse units (cf. Section 4.3.8).

```
<hypRelation type="concession" id="10">
  <satellite id="1" />
  <nucleus id="11" />
</hypRelation>
<parRelation type="contrast" id="11">
  <nucleus id="2" />
  <nucleus id="3" />
</parRelation>
```

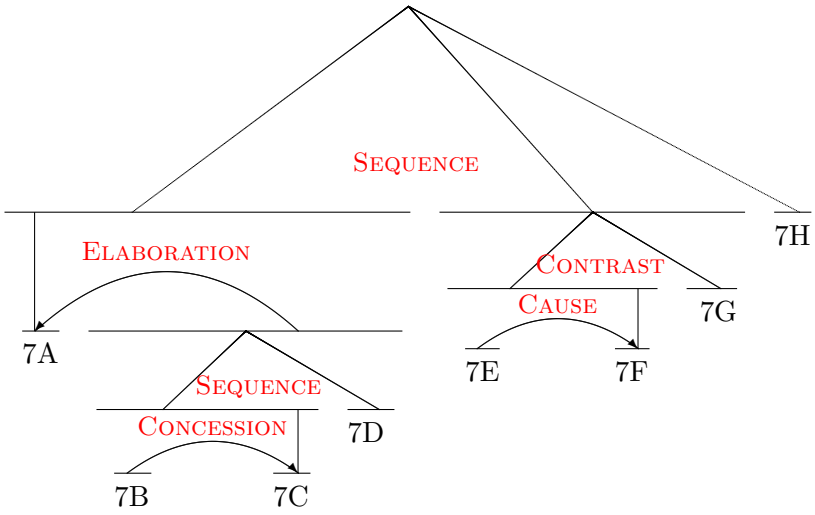
This referential markup syntax is flexible by design. It provides an elegant way to underspecify annotations by leaving out tree nodes or by stating possible disjunctive alternatives as sets of nodes. Also, *scores* may be mentioned for a node to indicate a preference or the result of some heuristics.

Paratactic and hypotactic relations are represented in the same way, and there are no additional span types. In contrast to the RSTTool format, our URML format is not aimed at presenting and manipulating RST *diagrams*. Rather, we wish to store rhetorically annotated data, potentially underspecified, in a manner that is independent from a particular application and readable for both humans and machines.

In contrast to semantics-based representations (Schilder, 2002), URML refers to textual data. It implicitly states linear precedence. Tree nodes in an analysis state immediate dominance; underspecified dominance situations have to be explicitly stated with concurrent tree nodes. This keeps annotations simple enough to work with them both manually and automatically.

4.3.4. Underspecification

Consider the sample text shown in Figure 4.1. What is its primary discourse intention, and what structure should be ascribed? Annotators may disagree. For example, 7D may be rightly characterized as being in temporal sequence with span [7B,7C], but it could also seen



[Yesterday, the delegates chose their new representative.]^{7A} [Even though Smith received only 24 votes,]^{7B} [he accepted the election with a short speech.]^{7C} [Then the assembly applauded for three minutes.]^{7D} [Due to the upcoming caucus meeting,]^{7E} [the subsequent discussion was very short.]^{7F} [Nonetheless the most pressing questions could be resolved.]^{7G} [The meeting was closed at 7pm.]^{7H}

Figure 4.1.: A text analysis within Rhetorical Structure Theory. It is one of the interpretations that can be derived from the underspecified URML representation in Section 4.3.8.

as in sequence with only 7C, and 7B is a concession to span [7C,7D]. Automated analysis tools might only give a partial answer here. Also, they might not be able to infer the Elaboration relation between 7A and the subsequent segments. In particular, automatic analysis will often encounter problems to locate the precise boundaries of larger segments: Where does the just-mentioned Elaboration end? Also, *nonetheless* at the beginning of 7G signals a Contrast or Concession, but based solely on surface cues, it is by no means clear how far to the left the first span stretches, i.e., what the exact scope of the *nonetheless* is.

We face two different kinds of ambiguities. Depending on the rhetorical bias of those annotating a corpus, they may decide that a single primary discourse intention cannot be established (cf. section 2.5.1). The other type of ambiguity may arise in a step-by-step annotation architecture, where certain decisions about a relation simply have not been taken yet.

Uncertainties related to the kind of relation can be represented by simply leaving out relation information — see Section 4.3.8. For instance, if the specific relation between two spans is unknown, the `relation` tag can omit the `type` attribute (node12). If, however, the class of a relation (hypotactic or paratactic) is known, a `hypRelation` or `parRelation` tag should be used. If a span is known to be an argument of a relation, but its role is unknown, it should be labeled `element` instead of `satellite` or `nucleus` (nodes 1E, 1F).

Each relation statement refers to its direct descendants in the tree via the identifier of

the relation. We could, alternatively, refer to spans by their left and right borders in the sequence of minimal discourse units. This way, ambiguous analysis would share common nodes high up in the tree. However, it would discard our assumption that the score assigned to those nodes relates to their descendants. In the example, the score given to the CAUSE relation might be lower because the connective *then* in segment 7D usually indicates a SEQUENCE relation.

The set of relations implements what is known in chart parsing systems as *subtree sharing*: We do not represent each tree derivation separately, but several analyses may share subtrees. This happens when two or more relations refer to the same relation as their nucleus or satellite (node10a and node10b). Another technique allows structure-sharing of nodes that are the same, but have different subtrees (*local ambiguity packing*). To do this, we introduce an attribute `group` which defines a common name for two or more relations. The `score` attribute also makes disjunction explicit, which would otherwise be implicit. Node11 summarizes three subtrees (with node10a, node10b, node10c). While this saves space, local ambiguity packing is limited to those cases where the unified node contains exactly the same data. This applies to the `score` given to node14 which might depend on the (ambiguous) structure of its content. If so, local ambiguity packing should not be used by the client application.[3]

4.3.5. Interpreting relation sets

Turning now to automatic tools for structural analysis, we encounter three paradigms to understand sets of relations, which correspond to different phases of analysis. The first is the *parse forest* scheme, where concurrent partial analyses are present. The parse forest holds relations already processed. They may be annotated with a score (Section 4.3.8, node10a and node10b). We indicate the scheme in the <`analysis`> element with a `status= "forest"` attribute. Different stages of discourse analysis will modify existing relation scores and add new relations. At this phase of analysis, a missing relation in the URML document indicates that this relation has not been considered yet. If it was considered, it should be included, possibly with a `score="0"` attribute. If the process is finished, pruning may occur and low-scoring relations may be removed from the relation set. This changes the semantics of the parse forest: relations that don't exist in the forest are assumed not to hold. To indicate this, we simply note it with a `status ="forest-complete"` attribute.

The third way to see a set of relations is the *interpretation* scheme. In this case, the analysis algorithm has singled out (and, possibly, scored) a whole, well-formed derivation for a document. We indicate this in the `status="interpretation"` attribute of the analysis. Several analyses may be given for a document. These reflect different *primary discourse intentions* as anticipated by an automated tool. The same mechanism is used by human annotators, as blind cross-validation asks for documents to be analyzed twice. Human annotators will most likely produce only the *interpretation* type of analyses.

As shown, the *forest* scheme assigns scores to each relation, while the second scheme assigns them to each analysis. This happens on grounds of the locality of classification decisions. In bottum-up style algorithms, the partial analyses are preselected only according to local constraints, i.e. constraints that refer to data covered by the local text spans. For

[3]We would like to thank Silvan Heintze for his helpful comments.

example, the rhetorical relation LIST might be proposed to hold between the segments B and C, because of a comma found at the right border of B and the connective *and* found at the left border of C. At a later stage in processing, the analysis algorithm might find that B, C elaborate on A, which ends with the words "is highly contradictory" and a colon. It may revise its earlier decision and find that a CONTRAST relation holds between B, C, because of the ELABORATION relation and the cue phrase to be found. These properties are *non-local*.

The three-scheme layout shown here does not restrict analysis algorithms to work decrementally, reducing search-space tool by tool. They can, alternatively, add anticipated relations. These tools would be expected to add their own <`analysis`> to the document, copying and extending the original <`info`> tag to preserve meta-data.

4.3.6. Specializing the DTD

The proposed document type definition (DTD) is open: We see it as a base class that may be extended. Therefore, unknown tags should be ignored by applications. A specialized DTD, derived from the base DTD proposed, should not alter original grammar productions, but it may add others and extend existing ones. It may also allow new optional tags within existing tags, and it may add optional attributes to existing tags.[4] This way, data can be exchanged between applications.

We have used a derived DTD which provides optional part-of-speech information for each token of a text and includes results from a stemming algorithm. Other extensions could, e.g., designate boundaries of topic chains or disambiguate discourse markers. While the following section shows an example URML document that conforms to the base DTD, an example document containing meta-data and part-of-speech tags can be found in Appendix B.

4.3.7. The URML document type defintion

```
<!-- DTD for Underspecified Rhethorical Structure Trees -->
<!-- Version 0.5 Jan05 2002 - Reitter - reitter@mle.media.mit.edu -->

<!ENTITY markup "((#PCDATA)*)">

<!ELEMENT urml (header, document*)>

<!ENTITY % reltype "(par|hyp)">

<!ELEMENT header (reltypes, postypes?)>
<!ELEMENT reltypes (rel)*>
<!ELEMENT rel EMPTY>
<!ATTLIST rel
name      CDATA #REQUIRED
type    %reltype;       #REQUIRED >

<!ELEMENT postypes (pos*)>
<!ELEMENT pos EMPTY>
```

[4] *Schema for Object-Oriented XML* is a proposal for an extension to XML. However, common XML/DOM libraries do not support this specification.

```
<!ATTLIST pos
        name    CDATA   #REQUIRED>


<!ELEMENT document (info?,text,analysis+)*>
<!ATTLIST document
        id      ID      #IMPLIED
        lang   CDATA   #IMPLIED>
<!-- language follows the iso639 stanard: en, de, fr, es etc. -->

<!ELEMENT info (source?, editor*, note?)>
<!ELEMENT source ANY> <!-- Source of the doc/analysis -->
<!ELEMENT editor (#PCDATA)>  <!-- Who inserted the doc, created the analysis
    ? -->
<!ATTLIST editor
        job     CDATA   #REQUIRED
        date    CDATA   #IMPLIED>
<!ELEMENT note ANY> <!-- Any additional notes -->


<!ELEMENT text (segment|unsegmented|ignore)*>  <!-- The text of a document
    -->

<!ELEMENT ignore (#PCDATA)> <!-- Data decided to be unimportant -->
<!ELEMENT segment (#PCDATA|p)*> <!-- A minimal unit of discourse -->
<!ATTLIST segment
        id      ID      #REQUIRED>
<!ELEMENT unsegmented (#PCDATA|p)*>  <!-- Text that has not been segmented
    yet -->


<!ELEMENT br EMPTY>
<!-- paragraph delimiter - refers to layout
        -->


<!-- a distinct analysis of the text. -->
<!ELEMENT analysis (info?, (hypRelation|parRelation|relation|span)*)>
<!ATTLIST analysis
status  CDATA   #IMPLIED
id      ID      #IMPLIED
score   CDATA   #IMPLIED
>

<!-- paratactic (multi-nuclear) relation -->
<!ELEMENT parRelation ((nucleus|element)*)>
<!ATTLIST parRelation
type    CDATA   #REQUIRED
id      ID      #REQUIRED
score   CDATA   #IMPLIED
>
<!-- if type is unspecified: unknown paratactic relation -->


<!-- hypotactic (nucleus-satellite) relation -->
<!ELEMENT hypRelation (((satellite|element),(nucleus|element))|((nucleus|
```

```
    element),(satellite|element))>
<!ATTLIST hypRelation
type     CDATA    #IMPLIED
id       ID       #REQUIRED
score    CDATA    #IMPLIED
>
<!-- if type is unspecified: unknown hypotactic relation -->

<!ELEMENT relation (((satellite|element),(nucleus|element))|((nucleus|
    element),(satellite|element))|((nucleus|element)*))>
<!ATTLIST relation
type     CDATA    #IMPLIED
id       ID       #REQUIRED
score    CDATA    #IMPLIED
>
<!-- if type is unspecified: unknown relation -->

<!-- satellite role -->
<!ELEMENT satellite EMPTY>
<!ATTLIST satellite
id       IDREF    #REQUIRED>              <!-- IDREF -->

<!-- nucleus role -->
<!ELEMENT nucleus EMPTY>
<!ATTLIST nucleus
id       IDREF    #REQUIRED>              <!-- IDREF -->

<!-- unknown element role -->
<!ELEMENT element EMPTY>
<!ATTLIST element
id       IDREF    #REQUIRED>              <!-- IDREF -->

<!-- If a satellite or nucleus consists of more than a single relation,
       the group of elements may be assigned an ID to be referenced using
       the group attribute. -->
```

4.3.8. Example URML document

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE urml SYSTEM "urml.dtd">
<urml>
 <header>
  <reltypes>
   <rel name="Cause" type="hyp"/>
   <rel name="Circumstance" type="hyp"/>
   <rel name="Concession" type="hyp"/>
   <rel name="Condition" type="hyp"/>
   <rel name="Contrast" type="par"/>
   <rel name="Elaboration" type="hyp"/>
   <rel name="Joint" type="par"/>
   <rel name="List" type="par"/>
   <rel name="Means" type="hyp"/>
   <rel name="Purpose" type="hyp"/>
   <rel name="Result" type="par"/>
   <rel name="Sequence" type="par"/>
  </reltypes>
```

```
 </header>
 <document id="sample001">
  <text>
   <segment id="7A">Yesterday, the delegates chose their new representative.</segment>
   <segment id="7B">Even though Smith received only 24 votes,</segment>
   <segment id="7C">he accepted the election with a short speech.</segment>
   <segment id="7D">Then the assembly applauded for three minutes.</segment>
   <segment id="7E">Due to the upcoming caucus meeting,</segment>
   <segment id="7F">the subsequent discussion was very short.</segment>
   <segment id="7G">Nonetheless the most pressing questions could be
       resolved.</segment>
   <segment id="7H">The meeting was closed at 7pm. <P /></segment>
 </text>
 <analysis status="forest-complete">
 <hypRelation id="node9a" type="Concession">
    <satellite id="7B"/>
    <nucleus id="7C"/>
 </hypRelation>
 <parRelation id="node9b" type="Sequence">
    <nucleus id="7C"/>
        <nucleus id="7D"/>
 </hypRelation>
 <hypRelation id="node10a" group="node10" type="Cause" score=".3">
    <satellite id="node9a"/>
        <nucleus id="7D"/>
 </hypRelation>
 <parRelation id="node10b" group="node10" type="Sequence" score=".6">
    <nucleus id="node9a"/>
        <nucleus id="7D"/>
 </hypRelation>
 <hypRelation id="node10c" group="node10" type="Concession" score=".1">
    <satellite id="7B"/>
    <nucleus id="node9b"/>
 </hypRelation>
 <hypRelation id="node11" type="Elaboration" score=".4">
    <nucleus id="7A"/>
    <satellite id="node10"/>
 </hypRelation>
 <relation id="node12">
    <element id="7E"/>
    <element id="7F"/>
 </relation>
 <parRelation id="node13" type="Contrast">
    <nucleus id="node12"/>
    <nucleus id="7G"/>r
 </parRelation>
 <parRelation id="node14" type="Sequence">
    <nucleus id="node11"/>
    <nucleus id="node13"/>
    <nucleus id="7H"/>
 </parRelation>
 </analysis>
</document></urml>
```

4.4. Applications

4.4.1. Building tools with XML/DOM

We use the data format to incrementally annotate data using various tools. The XML-based format serves as common interface between the tools. This interface standardization facilitates the implementation of a layer-based processing model. Each layer may be assigned to different machine or human annotators.

The Document Object Model (DOM) defines an application programmers interface to access a tree-structure of the document, created by an off-the-shelf XML parser. This allows fairly easy access to the data. In the development of annotation tools it has proven advantageous to use the DOM data in debugging. If corpus-related data is stored in the DOM in readable form, it may be visualized easily through a generic formating routine at any point during a system run. This is superior to state-of-the-art (C++) debuggers, which hardly ever display the content of complex data structures properly. As the XML data is always available in serialized form to be stored on hard-disk, turn-around times (compile&test) are reduced greatly, as only one module needs to be tested at a time. However, we recommend that simple tools that access only partial content, such as tokenizers and POS taggers, be implemented with simple methods. Ours operate on the raw XML file with regular expressions.

4.4.2. Annotating the Potsdam Corpus in URML

The format described has evolved from a practical application. We collected a corpus of newspaper texts and performed manual RST annotation (see Chapter 3).

Two annotators worked through 173 texts. Data was converted from the annotation application format to URML, part-of-speech-tagged and segmentized with Perl tools. These access external tokenization and tagging applications.

4.4.3. Editing and visualization

Thanks to the XML architecture, the URML-based corpus can be inspected and manually edited with one of numerous XML browsers and editors available. For the purpose of visualization, we provide a package for LaTeX(see Chapter C). Rhetorical analyses and their corresponding document text can be extracted from the URML data and converted to the appropriate format.

4.4.4. Conversion of the LDC discourse treebank

We also converted the LDC discourse treebank, a corpus of rhetorical annotations (Carlson et al., 2001) to our format. The RST annotated collection of 385 English language newspaper articles presented by Carlson et al. (2001) is the most extensive RST corpus. It contains almost 21,800 minimal discourse units in documents of varying sizes.

4.5. Conclusion

We have shown an underspecification method for rhetorical structure and introduced an XML-based corpus format. The format has already proven to be useful in corpus collection efforts, in a pipeline-based rhetorical parser, and in the implementation of a machine-learning analysis algorithm. We would like to see it as proposal towards a standardization of corpus representation and for tool building in rhetorical analysis.

5. Rhetorical Analysis

5.1. Introduction

In this chapter, bits and pieces described before are brought together. I describe how various aspects of rhetorical structure can be determined using a set of classifiers. These classifiers are trained using resources such as the one described in Chapter 3.

Rhetorical analysis is an incremental process: a chain of operations is applied to the data. In between these steps, we make heavy use of the underspecified representation discussed in Chapter 4.

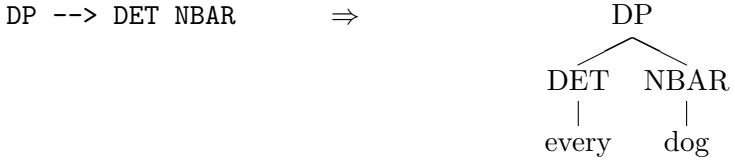
In the following section, I describe different approaches to rhetorical analysis, and, in detail, our novel data-intensive classification-based method. It consists of a series of transformations that allow the use of statistical classifiers in a chart-parsing algorithm (section 5.3). We use support vector machines as classifiers (section 5.4).

5.2. Previous work

The following overview of parsing approaches is not meant to be exhausting. Instead, we present a few approaches to statistical parsing of syntax and of rhetorical structure that we deemed relevant to our RST parsing approach. Also, I will discuss lexical chaining techniques, which analyze the topic structure of text. I will first give a comparison of syntactic and rhetorical parsing in general.

5.2.1. Syntactic and rhetorical parsing unified

Where do syntactic and rhetorical parsing intersect? In both instances of parsing, we try to construct a (possibly partially specified) tree structure from a sequence of terminal symbols. The tree structure is constrained by rich information attached to the terminal symbols; in syntactic parsing, it is morphologic and lexical semantic information. In our case, it is a set of feature-value pairs for rhetorical features on lexical level, including the occurrence of cue phrases or punctuation. In syntactic parsing, some form of grammar is used: in the case of phrase structure grammars, parsing often relies on a manually defined rule set. These rules *license* each node in the tree depending on its daughter nodes:



Phrase-structure rules may define *linear precedence* (in a DP, the DET may precede the NBAR), as shown in the example. In rhetorical parsing, the ordering is, as we may suspect, sometimes a useful restriction in rhetorical parsing. This is illustrated in Figure 5.2.1.

Figure 5.2.1 shows a marked analysis. The discourse marker *but* cannot enforce a CONCESSION relation from its relation-initial position. We can explain this with a strong ordering contraint for the phrase marker *but*, which forces the nucleus to follow the satellite.

What, if *but* was replaced by another discourse marker? We can then construct the inverse ordering in a single sentence (Figure 5.2.1).

Data from the Potsdam Corpus suggests an ordering preference, but no hard constraint. There are 125 CONCESSIONS in the corpus. In 91 of them, the satellite precedes the nucleus. In 34 cases, the nucleus is in initial position. A good explanation for this can probably be found on a more abstract level. The ordering preference could be formulated as a Theme<<Rheme constraint.

As a consequence, ordering reflects preference criteria. As such, it cannot be used as a hard constraint as in phrase structure rules, but as a feature influencing statistics.

Phrase-structure rules also license an *immediate dominance* relationship (a DP consists of a DET and an NBAR). In rhetorical structure, this would mean that, for example, texts are unlikely to elaborate (ELABORATION) on a fact presented with a CONCESSION relation.

Head-driven syntactic theories (Gazdar et al., 1985; Pollard & Sag, 1994) are based on the percolation of partial features in one designated daughter of a parent node (the head-daughter) to the parent node. Similarly, Marcu (1996) proposes a rule governing all rhetorical relations:

> A relation between two Spans A,B only holds if that relation holds between the nuclei of A and B respectively.

Thus, information of one designated daughter (the nucleus) percolates upwards in the tree. Syntactic parsers that perform unification could be applied to rhetorical parsing. However, even when unification is learned and not based on manually defined rules, common parsing algorithms are still based on compositionality for efficiency reasons.

> *Compositionality*: the meaning of a complex is wholly determined by its structure and the meanings of its parts. (Frege, 1892)

In our case, the rhetorical relation and some *salient information* that holds between a set of segments can solely and unambiguously be determined from these segments. Mann & Thompson (1988) do not refer to any conditions outside of the segments that are connected by the relation being defined. However, we need to define the properties of salient information. Marcu (1999) provides a mechanism to represent this text by adding a *promotion* property to each node. It contains the union of the promotion values of all salient children, on unit level, the unit itself.

Can compositionality be applied to rhetorical analysis? As already mentioned in 2.5.1, RST based manual analyses do not solely mirror this purely *rhetorical* perspective. Humans assume an *intentional* viewpoint of rhetorical relationships. As Moore & Pollack (1992) point out, the intentional and rhetorical analyses of a text are not necessarily isomorphic. They give the following example:

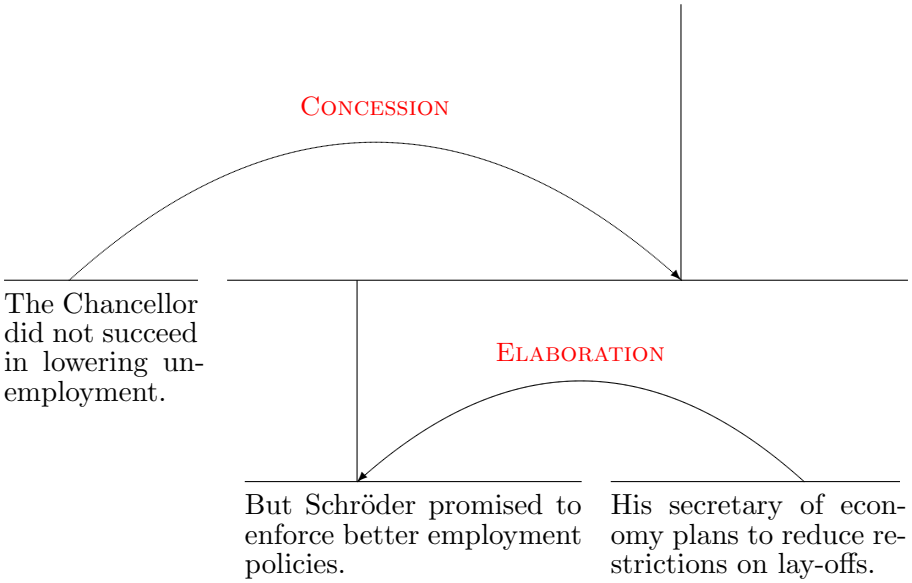


Figure 5.1.: Linear precedence in RST: acceptable analysis versus...

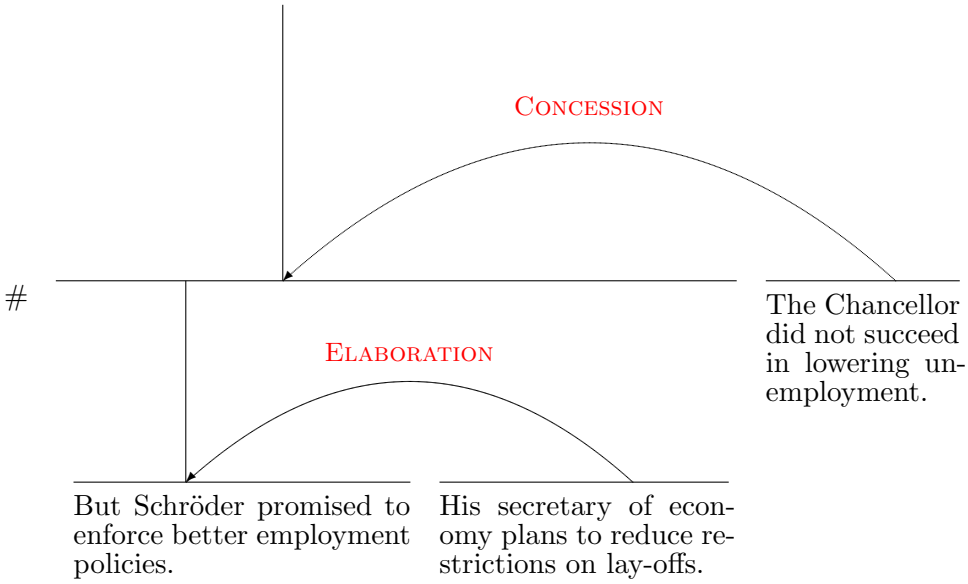


Figure 5.2.: ... an unacceptable RST analysis.

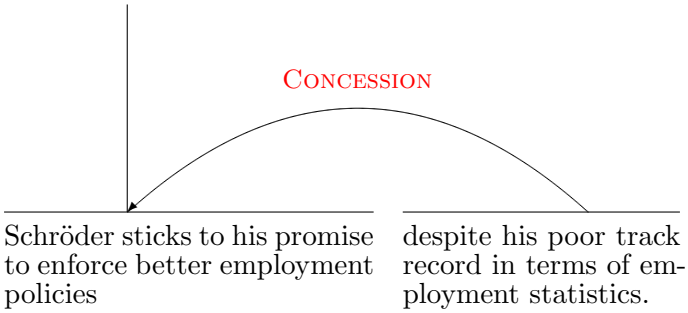
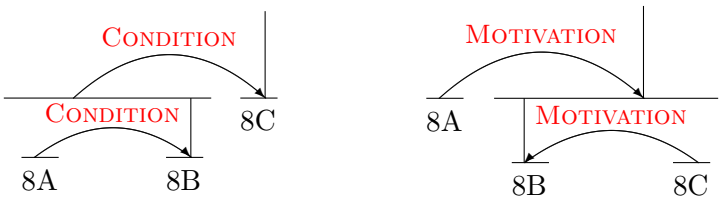


Figure 5.3.: Inverse ordering in a Concession

[Come home by 5:00.][8A] [Then we can go to the hardware store before it closes.][8B] [That way we can finish the bookshelves tonight.][8C]

Among the four different discourse trees possible (Marcu, 1999) are the following:



To disambiguate the set of analyses and pick the right one, knowledge (or clues) about intentions is needed. This is not necessarily apparent from the text connected by the relation, however, it is sometimes available in the context. This applies especially to meta-statements by the authors (*I am going to prove, that*, or *A summary will conclude the article.*). Since they are usually given before the statements are made that they refer to, this opens up a vage similiarity to syntactic parsing systems that look to the analysis done with the left context of a given symbol in order to disambiguate it. A counter-argument in favor of a compositional view would be to dismiss contextual rhetorical information as purely semantic data.

Another argument against the compositionality of rhetorical analysis trees is coreference: referential expressions in a child node can be bound by a concept introduced in tree leaves that are not sisters to the node. They are only positionally constrained by syntax and processing performance. Stylistically well-written texts have a high degree of cohesion (and thus, coreference).

5.2.2. Statistical parsing: a syntactic multi-stage parser

Adwait Ratnaparkhi demonstrates a method of syntactic parsing using Maximum Entropy models in his PhD thesis (Ratnaparkhi, 1998). The parsing system is trained using a set of structurally annotated sentences. There is no explicit grammar; instead, the grammar is encoded in the language models that result from training.

Ratnaparkhi sees parsing a sequence of classification decisions. Each decision has an effect on the current state of the analysis. The classifiers use local information from the current

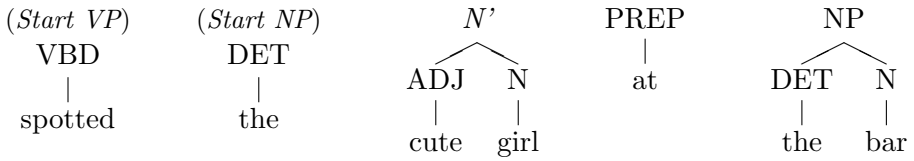
analysis to determine the next action. I describe his algorithm as an example for parsing with classifiers.

Parsing takes place in three stages. During the first parse, a statistical tagging algorithm assigns a part-of-speech tag such as *noun*, *finite verb* or *adjective* to each word of the input.

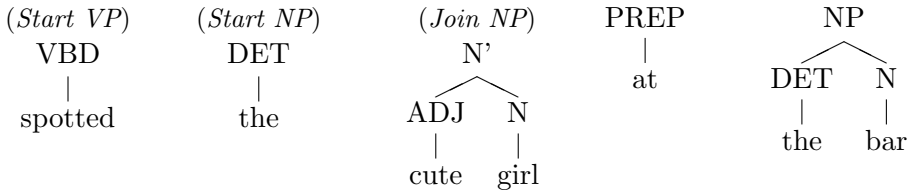
The second parse is meant to find phrases consisting solely of terminal nodes (i.e. words), which are called *chunks*. In this stage, the chunking classifier has three choices. To each word, it can assign on of three categories. "Start" would mark the left border of a chunk. All subsequent words of this junk will be tagged "Join". Words that do not belong to any category are marked "Other".

The third parse can be seen as the main part of syntactic parsing. In contrast to the first two steps, this stage always operates on the leftmost unfinished constituent and the following word in the analysis. In general, each existing constituent is annotated with either "Start X" (X being the label of a phrase, suche as VP oder NP), "Join X". Similar to step two, "Start X" marks the left border of a constituent, while "Join X" labels all other (direct) daughters of a constituent. These annotations are made by two processes, BUILD and CHECK.

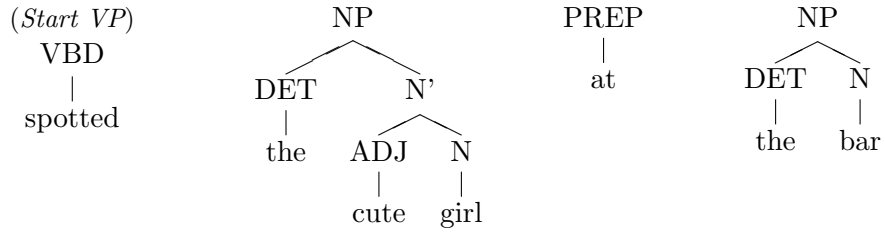
BUILD always examines the leftmost constituent that is not yet annotated. It decides, whether this constituent should be in adjoined position to the unfinished constituent to its left ("Join X") or begin a new constituent on its own ("Start X"). As an example, consider the following partial analysis.



At this point, chunking has already identified the two DET-N and ADJ-N phrases. Also, the first two words have been labeled "Start VP" and "Start NP". BUILD will now consider the leftmost constituent that is not annotated with "Start" or "Join", which is "N' ". It takes into account surrounding clues will be discuss later on. It decides to adjoin N' to the NP constituent by labeling it "Join NP"



After each BUILD decision, CHECK is invoked. This process determines, whether the sequence between the rightmost "Start X" label and the constituent just looked at by BUILD is complete and should be reduced. If it decides so, all of these subtrees are combined into an X node (here, X is NP) and our derivation looks like this:

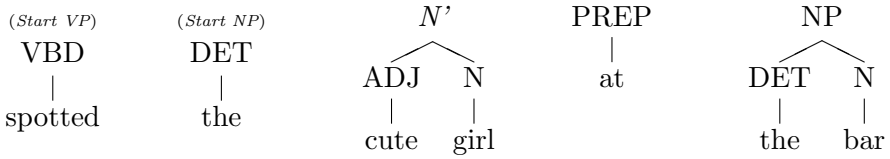


If CHECK decides not to reduce, the derivation is not changed. Next, BUILD is invoked again. As we recall, it operates on the rightmost unlabeled constituent, which is the nominal phrase (NP) "the cute girl". CHECK decides to adjoin it to the verb, building a VP. CHECK then decides whether to reduce or to go on, possibly shifting the PP.

Contextual features In order to make a decision, BUILD and CHECK use contextual information available at the time the decision is made. For each element in a 5-element window surrounding the constituent that is to be labeled, BUILD checks the head word, constituent label and, if any, "Start"/ "Join" annotation. It also checks for matching brackets, commmas or dots.

In order to decide whether a constituent is complete, CHECK looks at the direct descendants and checks their head word, the constituent labels. It also takes the label of the proposed constituent into account.

The information described is combined according to predefined templates. The BUILD and CHECK procedures will look at one, two or three constituents at once and combine the complex information derived from them to form several simple, binary features. Such a feature has a positive value, if and only if all the conditions are met in a given situation. Ratnaparkhi uses templates that are instantiated during training. One such template is *cons(0)*, which incorporates information about head word, label and annotation of the current constituent. Consider the following example:



BUILD(N') instantiates the above context template *cons*(0) as follows:

```
headword:  girl
label:  N
annotation:  (unknown)
```

This combination of values constitutes a distinct feature,which is a binary function. It evaluates to *true* if and only if presented with the above data. Extended forms of the template exist combining information from two or three constituents, which are selected using their relative position. *cons(0,-1,-2)* refers to the following data.

```
headword_1:  girl
```

```
label1:  N'
annotation1:  (unknown)
headword2:  the
label2:  DET
annotation2:  Start NP
headword3:  spotted
label3:  VBD
annotation3:  Start VP
```

Considered that this template contains open-class words in its *head* entry, it will generate a huge amount of irrelevant features. These are filtered (by frequency) in the next step. A decomposition of the features during filtering is not included. This could unify two distinct features with different headword_1 entries to form a single, more relevant feature.

Ratnaparkhi's *templates* are equivalent to our hard-coded *feature classes* (section 5.3.3).

Scoring parse trees Ratnaparkhi defines a score function for a parse tree in order to compare analyses. His score is the product of all actions at the different parsing stages (Tag, Chunk, Build, Check) involved in a particular derivation.

To find the best analysis, Ratnaparkhi applies a Top-K Breadth-first search heuristic. (For each action, the best K next steps are recorded at the end of queue. The queue is traversed from first to last, each action producing the next possible step.

5.2.3. Rhetorical parsing

Several works deal with the aspects of rhetorical parsing based on surface cues. Sanders & van Wijk (1996) discuss a general parsing architecture. It works incrementally and decides, for each new text segment, about attachment, relation type and relation (in this order).

The first-order formalization of RST developed in Marcu (1996) constrains possible tree analyses in most of Daniel Marcu's approaches. A more accurate choice of rhetorical relations is made by diambiguating potential cue phrases in their sentential vs. non-sentential use; Marcu (1997) uses regular expression pattern-matching to achieve this. Hereby, cue phrases are identified with 80.8 percent recall, 89.5 percent precision. The actual parsing then applies a heuristic that prefers right-branching nodes. As Corston-Oliver (1998c) notes, this algorithm suffers from combinatorial explosion.

Rehm (1998, 1999) collects ideas and approaches to summarize texts using rhetorical analyses. With a chain of tools, SGML markup is used to annotate texts. Schauer (2000a) assumes the availability of binding accessibility information for referents and presents an algorithm to integrate coreference and discourse structure.

Marcu (2000) discusses several other methods of parsing. He implements a learning approach that automatically derives rules to assign relations. Also, he shows a shift/reduce parser that uses the rules in a decision tree algorithm (trained with C4.5) to decide on its actions. Features relate, among others, to cue phrases and the previous parsing steps. The shift/reduce parser is computationally less expensive to the one shown below, but is not able to integrate non-local features.

Marcu (2000) also shows an implementation of a parser that uses manually derived rules. For each discourse marker, these rules state the distribution of nucleus and satellite (to the

left or to the right hand side segment), the relations that can be signaled, the size of textual unit. Relations are also hypothesized according to word cooccurrence. Rhetorical relations are recognized with .47 recall, .78 precision.

In the following, I will describe two other rhetorical parsing approaches in detail. RASTA uses knowledge about relations to reduce the search space. Marcu & Echihabi (2002) learn classification decisions from a automatically collected corpus.

Parsing with a discourse marker lexicon: RASTA

The *Rhetorical Structure Theory Analyzer* (Corston-Oliver, 1998c, RASTA) identifies rhetorical relations in text. Not unlike other approaches, it looks at various clues from syntactic and partial semantic analyses.

Deep syntactic analysis is used to eliminate ambiguities, where the discourse function of a phrase depends on its syntactic position. E.g., the phrase *as long as* is only interpreted as rhetorical clue, if a full sentence follows. RASTA distinguishes *necessary* and *contributing* criteria for a discourse relation. A selection of the necessary criteria for the SEQUENCE relation:[1]

- Clause_1 precedes Clause_2.
- Clause_1 is not syntacticly subordinate to Clause_2.
- Clause_2 is not syntacticly subordinate to Clause_1.
- The subject of Clause_1 is not a demonstrative pronoun, nor is it modified by a demonstrative.
- If either Clause_1 nor [sic] Clause_2 has negative polarity, then it must also have an explicit indication of time.
- Clause_2 must not be immediately governed by a CONTRAST conjunction.

RASTA calculates a heuristic score for a relation and a given pair of spans. This score depends on the individual scores for the contributing clues, which where set manually or learned during regression tests. This score is used to prioritize any following action: Highest ranking relation choices are applied first. Essentially, this follows the principle of local optimization used in a best-K algorithm.

It is not clear how the heuristic scores are exactly learned, and how they are combined. Since heuristics, which are analogous to the *features* of our approach, are highly interdependent, a non-linear classifier is needed, which is not the case. Corston-Oliver (1998a) notes that machine learning could not improve the performance of RASTA, judged from "preliminary statistical analysis". The reasoning behind this remains unclear. The view can be supported, however, with the argument that extensive manual tuning has been applied to the necessary/contributing criteria for relations.

RASTA detects a comparatively small set of only 13 relations, as a smaller set improves reliability (it makes the task easier). The relations are ASYMMETRICCONTRAST, CAUSE, CIRCUMSTANCE, CONCESSION, CONDITION, CONTRAST, ELABORATION, JOINT, LIST, MEANS, PURPOSE, RESULT and SEQUENCE.

[1] Taken from Corston-Oliver (1998c). Emphasis not in the original.

CONTRAST (3.8 Mio examples) [BOS ... EOS] [BOS But ... EOS] [BOS ...] [but ... EOS] [BOS ...] [although ... EOS] [BOS Although ... ,] [... EOS]
CAUSE-EXPLANATION-EVIDENCE (0.9 Mio examples) [BOS ...] [because ... EOS] [BOS Because ... ,] [... EOS] [BOS ... EOS] [BOS Thus, ... EOS]
CONDITION (1.2 Mio examples) [BOS If ... ,] [... EOS] [BOS If ...] [then ... EOS] [BOS ...] [if ... EOS]
ELABORATION (1.8 Mio examples) [BOS ... EOS] [BOS ... for example ... EOS] [BOS ...] [which ... ,]

Figure 5.4.: Marcu and Echihabi's text span extraction patterns, cited after Marcu & Echihabi (2002), numbers simplified.

Corston-Oliver, the thesis leaves us under-informed about the actual performance of RASTA. Quantitative evaluation would be needed, if we were to compare it to other analyzers.

Similar to the relation-specific rules RASTA works with, cue phrases can be disambiguated with a rule-like solution. They are syntacticly and semantically described with a set of attribute/value pairs (Knott & Dale, 1994; Stede & Umbach, 1998; Berger et al., 2002). The machine learning road we take is a different one: we detect cue phrases and their syntactic context independently as separate features. There are no hard constraints. All disambiguation takes place in the language model.

Marcu und Echihabi's unsupervised classification approach

Marcu & Echihabi (2002) investigate unsupervised learning of classification decisions. Large quantities of sentences are extracted automatically by applying search patterns to corpora. (See Figure 5.2.3 for a list.) Pairs of text spans containing cue phrases are extracted and labeled with one of six relations. With this method, a high number of annotated samples may be collected and used as learning material.

Marcu & Echihabi (2002) use a very straightforward learning method: check as features word pairs (w_i, w_j) and calculate a maximum likelihood probability $P(r|W_i, W_j)$. During classification, Bayesian statistics are used to find the most probable relation. Before training, the cue phrases that triggered the extraction of the training set sentences are removed to prevent the classifier from learning the cue phrases.

With their system, this approach achieves a major improvement of data separation. Out of 238 CONTRAST relations between adjacent spans of text in the RST-annotated corpus of Carlson et al. (2001), 177 are not signaled by a cue phrase. The classifier presented can tell 123 of these from being ELABORATIONS. Together with the signaled CONTRAST relations,

this results in an accuracy of .77. CONTRAST and CAUSE-EXPLANATION-EVIDENCE are separated with an accuracy of .87.

The learning curve for these classifiers does not level off within the range of sample quantities tested. For both corpora used (BLIPP and RAW), around 100.000 training samples are needed to achieve accuracy measures beyond .80 with the dichotomizer for ELABORATION vs. CAUSE-EXPLANATION-EVIDENCE.

The results show that statistical classification of a relation between two text spans is feasible. The learning algorithm based on Bayesian statistics is fairly simple. The major drawback of this approach seems to be that it can't be applied to rhetorical relations for which no unambiguous cue phrases are known. It is mainly for this reason that corpora are still needed, if a broad number of relations is to be recognized.

5.2.4. Lexical chaining

A lexical chain is a sequence of words or lexicalized phrases that bear a common sense or display some semantic similarity. Detecting these chains in a text means to identify regions of topics in a text, where *lexical cohesion* is high.

Morris & Hirst (1991) show that lexical chains may serve as excellent clue to the structure of topics and sub-topics in text.

The approaches to identify lexical chains are usually shallow and statistical. In most cases, a variation of the following steps is employed.

1. Text is tokenized and segmented. Segmentation can be achieved by collecting a fixed number of tokens (Hearst, 1994a).

2. A set of candidate words is selected from the tokens. To be eligible to take part in a lexical chain, the word must belong to an open class.

3. The set of candidates can be expanded semantically. Usually, a thesaurus (WordNet, EuroWordNet) is used. This step generates, for each token, synonyms of several senses.

4. Chains of words have contain semantically similar senses are built. Chain borders are globally optimized. Brunn et al. (2001) use the preference criterion

   $$wordrepetition >> synonym/antonym >> isa/include$$

   (The latter relation refers to inference steps in the WordNet database.)

Chain boundaries may be interpreted as boundaries of text segments. Hearst (1994a); Richmond et al. (1997) demonstrate this even without accessing a thesaurus. Litman & Passonneau (1995) combine different types of features to detect discourse boundaries in speech.

Why justifies the use of a lexical chaining measure in the analysis approach presented here? It is the hypothesis that the boundaries of lexical chains match the boundaries of the arguments of certain higher-level relations. For example, satellite and nucleus of a BACKGROUND or ENABLEMENT relation may contain different lexical chains, while an ELABORATION relation is likely to span over the same lexical chain.

5.3. A machine learning approach to rhetorical analysis

We see parsing as a series of classification decisions. In this section, we describe the basis for the parsing framework. In particular, this is the well-known data structure *chart* (section 5.3.1), a transformation process to prepare data for classification (sections 5.3.2, 5.3.3) and an algorithm that actually performs parsing.

The core part of the parsing is certainly the actual binary classification. There are many classification models that operate on the kind of data our transformation process produces. Among them are decision-tree learning (Quinlan, 1993, C4.5) or classifiers based on the maximum entropy principle. We will show one such method, support vector machines, in the subsequent section.

5.3.1. Incremental construction of a chart

Statistical-based rhetorical parsing borrows some ideas from rule-based chart parsing. I assume fundamental knowledge in parsing techniques. Generally, an incremental parser attempts to integrate segments of input into the analysis drawn so far. At each step, the parser needs to make a decision on what to do with a token that is read from the input. It may either start a new constituent, i.e. in our case, it may be the left argument to a new relation, or it may be integrated as new element in an existing relation. In the tradition of chart parsing, we call complete and partially detected relations *edges*, and the set of edges the *chart*. Formally:

$$
\begin{aligned}
&C \text{ is a Chart} \longleftrightarrow \text{for each } <i, r, \beta, n> \in C : \\
&r \in R, \\
&\beta =< \beta_1, \beta_2, ..., \beta_k >, \\
&hypotactic(r) \rightarrow 1 \leq n \leq k, \\
&\forall \beta_j \in \beta(\exists r' \exists \beta' \exists n'(< \beta_j, r', \beta', n' > \in C)))
\end{aligned}
$$

R is the set of known relations. Note that n assigns nuclearity for each edge and that β is a list of references to its children.

During learning, the chart is generated directly from the relation nodes of a document. Each edge corresponds to one relation node, terminal segments are inserted as edges without nuclearity and relation assignment and without children.

During parsing, the chart is built up incrementally. Each minimal discourse unit is inserted in the chart (as edge with $\beta = \{\}$ and used to start a new edge or to complete existing adjacent edges. The decision about what to do with an edge, which relation to assign and which daughter β_i to designate as nucleus n is taking by a classifier.

The chart described here corresponds closely with the serialized data format introduced in chapter 4. In fact, each edge as shown before translates to a <relation> or tag and its contents.

To build up the chart, we need a classifier to tell us when all nodes belonging to a relation are found (compare Ratnaparkhi's CHECK procedure), and whether to attach a new node to the constituent before or to start a new subtree (BUILD). How can such decisions be drawn? The classifiers look at various linguistic properties (features) that occur in the text spans that are possibly related. To be successful, they need to observe a pattern of features

in the text spans — this pattern (feature vector) needs to be structurally same in all cases. That means, the same features need to be observed in the texts.

We achieve this with an intermediate data structure, the *classification instance*. In the following, the transformation is defined.[2]

5.3.2. Edges split up in classification instances

Classification instances are structurally constant units that allows classifiers to observe a pattern of features. The challenge is that paratactic relations can connect any number of nodes. Also, the nucleus and satellite elements of hypotactic relations can appear in two different orders, but features such as cue words are usually tied to the nucleus or the satellite, not to the first or second element of a relation.

We define a transformation function μ that assigns, for each edge, a set of classification instances. A classification instance $c :=< r, \delta, g >$ holds a list of text spans δ and a single span g. The role of δ and g differs in hypotactic and paratactic relations. For hypotactic relations, which are assumed to be binary, δ contains the nucleus and g represents the satellite. [3] For all $r \in R, hypotactic(r)$:

$$\mu(< i, r, \beta, 1 >) = \{< r, < \beta_1 >, \beta_2 >\}$$
$$\mu(< i, r, \beta, 2 >) = \{< r, < \beta_2 >, \beta_1 >\}$$

For paratactic relations, our classifiers need to decide multiple times whether to attach a newly encountered segment or not. To make a decision, the classifier needs to see all the elements to the left, because a discourse signal could be contained anywhere in these elements. Therefore, for all $r \in R, paratactic(r)$:

$$\mu(< i, r, \beta, 0 >) = \{< r, < \beta_1, \beta_2, ..., \beta_n >, \beta_{n+1} > | 2 \leq n < |\beta|\}$$

As an example, we will consider the case: $< 1, \text{CONTRAST}, < 2, 3, 4, 5 >>$. This relation node results in the following set of classification instances: {<CONTRAST, $< 1 >, 2 >$, < CONTRAST,$< 1, 2 >, 3 >$, < CONTRAST,$< 1, 2, 3 >, 4 >$} .

If put in the right order, the classification instances describe a serialized view in a left-to-right parse process. Again, this is what we want to do with the classification instances: Whenever an attachment of a text span is structurally possible, the situation represents a classification instance. We use present it to the classification algorithm to decide, whether the subtree may be added to an existing subtree, and if so, which relation the subtree could take on.

In our parsing framework, we only allow binary hypotactic relations. This is for efficiency reasons. The formalism established can handle the schemas proposed in Mann & Thompson (1988), where several hypotactic relations share one nucleus. The μ function (and with it the parsing algorithm) can be extended tom cover hypotactic schemata with more than one satellite by defining μ for hypotactic r in a similar fashion as we did for paratactic r.

Currently, the classification instances contain references to the full subtrees in β. Following Marcu's compositionality criterion, we could restrict the view to only look at the

[2] For a definition of terms, refer to the reference Appendix A

[3] the linear precedence of the text spans will be encoded as feature

promotion sets of the elements in β, which is, in general, the content of their nuclei. The reason we don't take the approach is that in this case, we would require a very reliable identification of the promotion set. Otherwise, the error adds up. The algorithm shown does not suffer from the this problem.

5.3.3. Transforming classification instances into feature vectors

Automatic annotation algorithms exist for many of the surface clues we will present here. Some are very reliable, such as part-of-speech tagging, others depend on the availability of large knowledge bases, such as topic chaining systems, which use ontologies, or syntactic parsers, which rely on broad-coverage grammars and morpho-lexical information. The concept of contribution renders the system robust: Even though some observations may be misleading due to their high degree of ambiguity, the result may still be correct. Therefore, it seems sensible to examine possible heuristics that could serve us in finding clues at significantly lower cost than a full analysis would take. The system uses heuristics that will not (by nature) demonstrate perfect performance. They draw their advantages from an easy and fast implementation, low resource usage and high coverage. To evaluate and improve each heuristic, a randomly selected set of whole texts was manually annotated with the intended features. Then, the performance of the algorithm was measured.

We call the cues observed to make that decisions *features*. A feature is a function $F : C \rightarrow [-1, 1]$ (where C is the set of classification instances).

The features are determined before learning: General *feature classes* are hard-coded (as described below), such as: *There is a cue phrase in the left part of the relation.* The actual features are instantiations of these classes, which are collected from the corpus, such as: *The cue phrase 'However,' is found in the left part of the relation.* This creates a finite set of features. For each feature and each classification decision, we can observe a *feature value*. For the example given before, possible feature values would be $\{+1, -1\}$. However, non-discrete feature values are possible. The mapping from a feature to a feature value for a certain classification instance is called *feature vector*.

We distinguish two types of features:

- Local features: These hold only information that is available within a classification instance. These may comprise information about the presence of discourse markers, of words with selected part-of-speech categories (conjunctions, Nouns etc.), the length of the text segment. This feature class also includes the presence of certain child relations (direct descendents). For example, this structural feature vector can also contain information about the number of relations contained in the constituent(s) the text represents.

- Non-local features: These features refer to contextual knowledge, such as: the type of siblings the whole text span (both spans that the classification instance refers to), to depth of embedding (length of the shortest path from the top-node in the tree to the current relation) and concept-cooccurrence information from the lexical chaining algorithm.

In the selection of heuristics, we try to avoid too specific definitions of the features. No additional data sets were given to the learning algorithm. The learning algorithm was in

charge of detecting useful features and discarding others. On the one hand, this made the task harder, on the other hand, we do not risk forgetting features that show a hidden contribution to the classification decisions.

Features were instantiated from templates as first step of the learning phase. In evaluation, they were only acquired from the test set. I will now describe the feature templates used to find the set of features.

Cue words and pronouns

The strongest clues consist of *discourse markers*, or *connectives*. However, they suffer from their many-to-many relationship to rhetorical relations: they are highly ambiguous, and relations may be signaled in different ways. We define a class of features $\textsc{Cue}_{w,b,s}(c)$ that return a positional indicator for a cue word w within the first ($b = 0$) or second ($b = 1$) half of the $first(c)$ ($s = 0$) or $second(c)$ ($s = 1$) span. For $b = 2$, it gives a binary indication of whether the word occurs somewhere within the span, as some relations are marked using combinations of lexical material: *the city has issues* ***so*** *much supply recently, /* ***that*** *some people are getting a little concerned.*

Words that form cue phrases are automatically extracted from the Corpus according to their part-of-speech. Apart from lexical items that overtly indicate rhetorical structure, the choice of pronouns and articles (definite, indefinite, missing) is taken into consideration. Table 5.1 shows the part-of-speech categories used to select potential connectives. The part-of-speech annotation follows the Stuttgart-Tübingen Tagset (Schiller et al., 1995) (for German) and the Enriched Tagset of the Brown National Corpus.

As discourse cues tend to be found at the beginning of a text span ("However,", "Thus") or at its end (", too" and punctuation), the relative position of a cue within the segment considered is encoded in the feature value. For each cue, two features were added: one denoting the occurrence of the cue in the first three words of the text span, the other feature concerns the cue position in the last three words.

Why do we need to filter the words at all? Theoretically, we could simply add all known words as features, since additional features should not slow down the learning process significantly. However, taking all words that occur in the training corpus into account would impede the transformation of classification instances into (then large) feature vectors. The filters make sure, however, that recall is as high as possible. A good precision measure is less important, since the reliability of each feature is calculated in the training phase of the support vector model.

While this approach can certainly find cue words, its drawback is that it cannot find whole cue phrases (*on the one hand*). The learning algorithm may use the words independently and discover combinations on its own. As with all features detected, the decision about prefabricated lists of cue phrases involves a trade-off between linguistic knowledge encoded in feature detection and complexity of the learned part, which comes with a need for additional training data. We additionally included a list of known cue phrases. For the German corpus, a list of 685 cue phrases collected by Rehm (1998) is recognized; for English, we added 320 discourse markers collected by Knott (1996).

A further approach to detecting cue phrases would be an informed one: With the help of a discourse marker lexicon cues could be disambiguated and included as features in form of (fuzzy) sets of possible relations. Common approaches to discourse marker lexicons (Stede

Table 5.1.: Part-of-speech categories used to filter potential cue words.

ART	determiner	der, die, ein
PPER	personal pronoun	er, sie
ADV	adverb	darum, aber, erstens
PDAT	demonstrative pronoun	dieses, jene
PDS	demonstrative pronoun (noun phrase)	das
PRELAT	relative pronoun	deren, dessen
PRELS	relative pronoun (noun phrase)	welcher, deren, dessen
PWAT	interrogative pronoun	welche
PWS	interrogative pronoun (noun phrase)	was
PWAV	adverbial interrogative pronoun	wann, worüber
PAV	pronominal adverb	trotzdem, deswegen, außerdem
KOUI	subordinating conjunction (inf. verb)	um (zu), anstatt (zu)
KOUS	subordinating conjunction (sentence)	dass, weil, obwohl, damit
KON	co-ordinating conjunction	entweder...oder, denn, doch
ITJ	interjection	ach, mhm, tja
PTKANT	answer particle	ja, nein
AT–	article	a, an
D–	determiner	this, that, a, the, another
CC	Coordinating conjunction, general	and, or
CCB	Coordinating conjunction: but	but
CS–	Subordinating conjunction	if, when, while, because,than
JJR	General comparative adjective	better, older
PNQ–	Wh-pronoun	who
RR	General positive adverb	often, long, easily, however
RRQ	Wh- general adverb	how, when, where, why
RRQV	Wh-ever general adverb	however, whenever, wherever
RT	Nominal adverb of time	tomorrow
XX	negation particle	not

& Umbach, 1998; Berger et al., 2002) define sets of contraints (syntactic, semantic) which refer to deep linguistic analysis — data unavailable to our algorithm.

Introduction of concepts

The way noun phrases occur is specific to discourse boundaries. In journalistic texts, definite noun phrases with a common noun (NN, "the chancellor", "das Restaurant") tend to refer to a concept that was introduced before. Indefinite noun phrases do not refer in a strict semantic sense. The indefinite subject in 9B is used to begin an evaluation or a conclusion of 9A.

[Rund 15 grüne Bundestagsabgeordnete haben erhebliche Zweifel, ob sie der Bereitstellung von Bundeswehrsoldaten für den Anti-Terror-Kampf zustimmen können. (...)]^{9A} [Ein Kanzler, den in einer solch existenziellen Frage die eigene Koalition im Stich lässt, muss sich neue Partner suchen.]^{9B} (maz_{9725})

Proper nouns (NE) are usually introduced with an attributive common noun phrase: "Bundeskanzler Schröder", "World's largest Unix vendor Apple". Later, they are referred to using common nouns or variations of proper nouns, but not using the introductory attributed form.[4]

This distinction of introducting referents and referring noun phrases makes up for the fact that a thesaurus is hardly complete, thus, proper dereferencing of noun phrases cannot be achieved. The same argument supports the detection of determiners in noun phrases with common nouns.

The detection of phrases is imlemented as regular expression matching. We detect the sequence NN-NE (proper nouns with a noun attribute) toward the beginning of a sentence. We also detect DDET-(ADJA—VVPP)*-NN, DDET being a definite determiner (in German: {der, die, das}).

$\textsc{NounPhrase}_e(c)$ returns true if a regular expression e occurs within the first three words of $right(c)$.

Punctuation

In German and English, punctuation at the right segment border of the left segment of a classification instance is an excellent cue. $\text{PUNCT}_{p,b,s}(c)$ returns true if punctuation p occurs within the first ($b = 0$) or last ($b = 1$) ten characters of $left(c)$ ($s = 0$) or $right(c)$ ($s = 1$).

Part-Of-Speech

The part-of-speech categories of the words at the borders of segments may help disambiguate the type of relation (sentential). Hirschberg & Litman (1993) note that they contribute only very little to the classification result. Our feature $\text{POS}_{p,b,s}(c)$ returns true if a word tagged

[4] Journalistically trained writers will employ a new (possibly shorter) introductory form, if the concept is not as salient any more, i.e. if the noun phrase occurrs at greater distance from the last occurrence. This knowledge is useful to determine the preferred size of the tree constituent it combines with.

p occurs within the first ($b = 0$) or last ($b = 1$) three words of $left(c)$ ($s = 0$) or $right(c)$ ($s = 1$).

Lexical similarity

Topic chaining has long been a popular property in discourse parsing (Hearst, 1997; Morris & Hirst, 1991; Kurohashi & Nagao, 1994; Litman & Passonneau, 1995; Richmond et al., 1997). Semantic similarity is measured by counting coocurrences of nouns, verbs and adjectives. The system also counts small sequences containing up to k words. Hearst (1994b) defines similarity as

$$sim(b_1, b_2) = \frac{\sum_{t \in T} w_{t,b_1} w_{t,b_2}}{\sqrt{\sum_{t \in T} w_{t,b_1}^2 \sum_{t \in T} w_{t,b_2}^2}}$$

w_{t,b_1} is a weight associated with the frequency of word t in text span b_1. T is the set of all tokens in the text minus closed-class categories. TopicSimilarity(c) returns $sim(leftwords(c), rightwords(c))$.

Span lengths

Intuitively, some connectives have a argument-length-dependent distribution. A Contrast relation will rather have a short satellite, at least, related to the total length of the text. Authors are expected to focus on their main argument, not on counter-arguments. Thus, the ratio of the text span lengths (in words) is used as feature.

While Knott (1996) notes "Insensitivity to span size is a useful feature of coherence relations" (p. 13), Marcu (1997, 2000) uses the number of subconstituents as a feature. The analyzer shown here lets the learning algorithm decide whether span length is an important feature.

As paratactic relations may also show a preference for the number of elements contained, the number of nuclei is counted. The feature Spanlength(c) returns the number of nuclei to the left of c. This can be a local feature, as the parser follows left-to-right processing.

5.3.4. A parsing algorithm for rhetorical structure

All the techniques and analysis steps described before now come together in an attempt to derive valid rhetorical structure from a text.

The classifiers described are already able to make different kinds of useful decisions:

- Rhetorical relation: this decision comes directly out of running the multi-class classifier on a classification instance.
- Nuclearity (which text span is the nucleus?): For hypotactic relations, we run each polychotomizer twice, once for each possible distribution of roles (satellite/nucleus).
- Attachment preference for a text span during structure building: Again, different possible attachment decisions may be stored and scored using the polychotomizer.

Those decision types are all integrated when it comes to rhetorical parsing. Whenever a decision concerning two text spans is to be made, all structurally sound decisions[5] are converted into a set of classification instances, which are then scored using the polychotomizer. The best scoring are explored further. Note that in each attachment situation, the parser must also decide about the relation that holds between the two spans in question and, in the case of a hypotactic relation, which element is assigned the role of nucleus.

As for the non-local features used, we cannot (finally) classify a classification instance before we know the rest of the tree. Since classification decisions depend on each other, we need a two-pass decision-making strategy.

- PARSE: Beam-search based preselection of a chart containing all possible derivations: During first pass, not all information is available. Based on the local information that is available, search space is reduced. We note the best K possible decisions and expand them.
- GLOBALSCORE: Scoring each possible derivation with the global set of features and selecting the best one. Each analysis (containing no disjunctive interpretations) is scored using full context.

The selected discourse tree represents an unambiguous rhetorical analysis, structurally conforming to RST.

Formal description of the parsing algorithm Note that the functions LOCALCLASSIFY and GLOBALCLASSIFY call binary classifiers.

1. Beam-search based edge detection:
Global structures:
R : set of relations C : set of edges of the form $< i, r, \beta, n >$ (chart)

PARSE (L: list of terminal edges):
    for each $l \in L$,
        $C \leftarrow l$,
        COMPLETE(l)

COMPLETE(i_s: edge index):
    O : set of triples < score, classification instance, nuclearity >
    for all $i \in C$, with RIGHTBORDEROF(i_s) = LEFTBORDEROF(i):
        ci : classification instance
        for all $r \in R$
            if *paratactic*(r)
                $ci :=< r,$ CHILDRENOF$(i_s), i >$ (present children in old / new edge)

[5]The analysis must form a valid tree. In short: text spans described by rhetorical relations may not overlap and must be contiguous. There may be not more than one rhetorical relation in the final, unambiguous analysis of a text for two spans of text; nuclearity assignment must be unambiguous and each relation must be either hypotactic or paratactic. Marcu (2000) develops a formal framework.

$O \leftarrow <$ LocalClassify(ci) , ci , $NUC >$
else if *hypotactic*(r)
$ci :=< r,$ ChildrenOf(i_s)$, i >$ (new element is satellite)
$O \leftarrow <$ LocalClassify(ci) , ci , $SAT >$
$ci :=< r, < i >,$ Flatten(ChildrenOf(i_s))$ >$ (... is nucleus)
$O \leftarrow <$ LocalClassify(ci) , ci , $NUC >$

for the K highest-scoring classification instances $< r, \beta, e, a > \in O$,
$i_j :=$ a new index
if $(a = NUC)$
$C \leftarrow < i_j, r, \beta \bigoplus < i_s >, 2 >$ (second element is nucleus, or r is paratactic)
else
$C \leftarrow < i_j, r, \beta \bigoplus < i_s >, 1 >$ (second element is nucleus)
Complete(i_j)

LocalClassify ($ci =< r, \beta, \gamma >$: classification instance):
observe all local features in ci and apply classifier for relation r to
the resulting feature vector. Return its normalized score.

LeftBorderOf (i: edge index):
for the $e =< i, r, \beta, n > \in C$,
if $MinimalDiscourseUnit(\beta_1)$, return β_1.
else, return LeftBorderOf(β_1).

RightBorderOf (i: edge index):
for the $e =< i, r, \beta, n > \in C$, $k = |\beta|$,
if $MinimalDiscourseUnit(\beta_k)$, return β_k.
else, return RightBorderOf(β_k).

Flatten (γ: list of edge indexes with $|\gamma| = 1$):
return γ_1

2. Derivation scoring

GlobalScore:
$O =$: set of pairs <score, set of edges>
for each well-formed $I \subset C$
t:=0 : score
n:=0 : counter
for each $e \in I$
for each $ci \in \phi(e)$
$t := t +$ GlobalClassify(ci)
$n := n + 1$
$O \leftarrow < \frac{t}{n}, I >$

return: I with highest score s in $< s, I > \in O$

GlobalClassify ($ci =< r, \beta, \gamma >$: classification instance):
observe all features in ci and apply classifier for relation r to
the resulting feature vector. Return its normalized score.

A worked example As an example for how the algorithm determines tree structure, we discuss the analysis of a short text fragment from the Potsdam Corpus.[6] While the text is original, the data structures shown are not actual data for reasons of simplicity. We will use elements from URML to describe the state of the chart.

Parse The analyzer begins with inserting an edge with a terminal element (maz9612.1) in the chart. There are no adjacent edges to the left, so we shift the next edge as well:

```
<segment id="maz9612.1">
  The army expects no consequences for the regional developments
  in the tourist sector from its activities in the air.
</segment>
<segment id="maz9612.2">
  Years of experiences in other parts of the republic had shown that
  military facilities can indeed coexist with the interests of tourism.
</segment>
```

For the edge maz9612.2, there is one adjacent edge to the left. Each known relation is hypothesized to hold between those two edges, hypotactic relations are tested twice for each distribution of nucleus/satellite. We add the best two edges to the chart:

```
<hypRelation id="maz9612.1001a" type="evidence" score="0.4">
  <nucleus id="maz9612.1" />
  <satellite id="maz9612.2" />
</hypRelation>
<hypRelation id="maz9612.1001b" type="background" score="0.2">
  <nucleus id="maz9612.1" />
  <satellite id="maz9612.2" />
</hypRelation>
```

For the two terminal segments in our chart, other attachments may exist than the two relations shown. It may well be that maz9612.1 and maz9612.2 are not directly connected through a relation at all. These possibilities are explored in the further steps. Next, the third minimal discourse unit is shifted:

[6] The original document is in German. The translation shows that discourse markers are hard to translate, in this text *jedenfalls* (anyway, anyhow, by all means, here: after all). The original text, which is the basis for the rhetorical analysis given, reads

[Die Bundeswehr geht davon aus, dass ihre geplanten Aktivitäten aus der Luft keine relevanten Auswirkungen auf die Entwicklung der Region im touristischen Bereich haben werden.] [Jahrelange Erfahrungen in anderen Gebieten der Bundesrepublik hätten gezeigt, dass militärische Einrichtungen durchaus mit den Interessen des Tourismus in Einklang zu bringen seien.] [So jedenfalls steht es in der Erläuterung zum Luft/ Bodenschießplatz bei Wittstock.] [Dazu werden in der kommenden Woche die betroffenen Gemeinden angehört.]

```
<segment id="maz9612.3">
  This is, after all, written in the declaration regarding the air- and land
       firing
  range close to Wittstock.
</segment>
```

This edge as three adjacent nodes: maz9612.2, maz9612.1001a and maz9612.1001b. For all known relations and all distributions of nuclei involving an adjacent node and the new segment, new edges are hypothesized. If 15 hypotactic and 4 paratactic relations are used (as in the Potsdam Corpus), we need consider 102 possible attachment, relation and nuclearity choices. Again, we keep the 2 best choices:[7]

```
<hypRelation id="maz9612.1002a" type="justify" score="0.3">
  <nucleus id="maz9612.2" />
  <satellite id="maz9612.3" />
</hypRelation>
<hypRelation id="maz9612.1002b" type="summary" score="0.4">
  <nucleus id="maz9612.2" />
  <satellite id="maz9612.3" />
</hypRelation>
```

These new relations are again subject to attachment to another adjacent node. There is only one adjacent node (maz9612.1). Choices for relations and nuclearity are scored, resulting in two additional edges:

```
<hypRelation id="maz9612.1003a" type="evidence" score="0.7">
  <nucleus id="maz9612.1" />
  <satellite id="maz9612.1002a" />
</hypRelation>
<hypRelation id="maz9612.1003b" type="evidence" score="0.6">
  <nucleus id="maz9612.1" />
  <satellite id="maz9612.1002b" />
</hypRelation>
```

There are no possible attachments for these new edges, so a new terminal segment is shifted:

```
<segment id="maz9612.4">
  In these matters the affected communities will be heard next week.
</segment>
```

Edges to be considered for attachment are maz9612.3, maz9612.1002a, maz9612.1002b, maz9612.1003a, maz9612.1003b. We assume the following scores:

```
<hypRelation id="maz9612.1004a" type="justify" score="0.3">
  <nucleus id="maz9612.3" />
  <satellite id="maz9612.4" />
</hypRelation>
<hypRelation id="maz9612.1004b" type="sequence" score="0.6">
  <nucleus id="maz9612.1003a" />
  <nucleus id="maz9612.4" />
</hypRelation>
```

Node maz9612.1004a could be attached to maz9612.2, maz9612.1001a, maz9612.1001b. The classifier ensemble votes in favor of the following two edges:

[7] In a better form of the beam search algorithm, we might choose to keep more than two relations according to their scores and the number of relations considered.

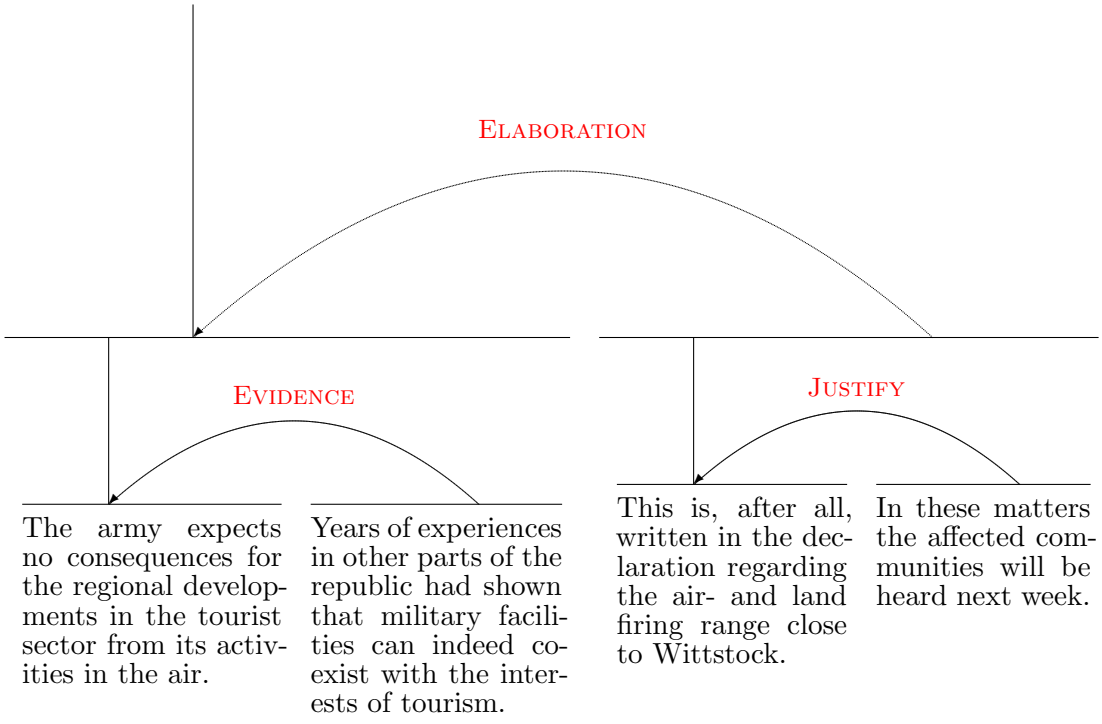


Figure 5.5.: Result of parsing process.

```
<hypRelation id="maz9612.1005a" type="elaboration" score="0.4">
  <nucleus id="maz9612.2" />
  <satellite id="maz9612.1004a" />
</hypRelation>
<hypRelation id="maz9612.1005b" type="elaboration" score="0.4">
  <nucleus id="maz9612.1001a" />
  <satellite id="maz9612.1004a" />
</hypRelation>
```

Node maz9612.1005a can be attached to the first discourse segment:

```
<hypRelation id="maz9612.1006a" type="preparation" score="0.1">
  <satellite id="maz9612.1" />
  <nucleus id="maz9612.1005a" />
</hypRelation>
<hypRelation id="maz9612.1006b" type="elaboration" score="0.2">
  <nucleus id="maz9612.1" />
  <satellite id="maz9612.1005a" />
</hypRelation>
```

All minimal discourse units have been processed and all possible attachments were scored. There are 16 edges in the chart. It's status in URML terminology is now *forest-complete*.

GlobalScore The second phase of parsing scores all trees contained in the chart. Other than during the Parse step, classification uses global and local features, as for each relation and each classification instance, context is available. We consider all top-level nodes that span all minimal discourse units. In our case, these are maz9612.1004b, maz9612.1005b, maz9612.1006a, maz9612.1006b. The best-scoring analysis is based on maz9612.1005b.

```
<analysis id="maz9612.analysis4" score="0.8" status="interpretation">
  <hypRelation id="maz9612.1001a" type="evidence">
    <nucleus id="maz9612.1" />
    <satellite id="maz9612.2" />
  </hypRelation>
  <hypRelation id="maz9612.1004a" type="justify">
    <nucleus id="maz9612.3" />
    <satellite id="maz9612.4" />
  </hypRelation>
  <hypRelation id="maz9612.1005b" type="elaboration">
    <nucleus id="maz9612.1001a" />
    <satellite id="maz9612.1004a" />
  </hypRelation>
</analysis>
```

This analysis, visualized in Figure 5.5, represents the final outcome of the analysis process.

5.4. Support Vector Machine learning

We use supervised machine learning to run the classifiers described in the previous section. This means: the classifiers base their decisions on knowledge that is automatically acquired from a set of sample documents. During training, the machine learning algorithm determines general characteristics of the samples that belong to each assigned category, in our case: the relations. The acquired knowledge, commonly called *language model*, may be used to recognize the characteristics found in new, not yet categorized data that is presented to the classifiers. In the following, the choice of support vector machines (SVM) for this learning task is discussed, and a brief introduction to SVMs in learning and classifying is given.[8]

5.4.1. Learning

To train the system, the documents of the training set are compiled into classification instances, which result, in turn, in training patterns (pairs of feature vectors and assigned classes). The decision about which training algorithm to chose depends on certain properties of the training data. There are several learning and classification algorithms, and usually, there is more than one training algorithm available for each classification method.

- Our classification problem uses features that are inter-related, and the exact qualitative and quantitative feature inter-dependence is not known. Bayes-based algorithms, a standard in statistical language processing (Manning & Schütze, 1999), can only approximate the ideal classification solution in this case. A training algorithm is needed that can solve highly non-linear problems.
- We also need an algorithm that handles a large number of features (> 1000) well.
- Compared to other machine learning tasks, we train on a small number of samples (< 10000).

Support Vector Machines (Vapnik, 1995) fulfill all these criteria with ease. They haven shown to perform extremely well on different kinds of classification tasks, usually with very

[8] I follow Schölkopf (1998) and Burges (1998) in the discussion of SVMs.

little a-priori knowledge involved. SVMs are based on three foundations: SVMs are large-margin classifiers and they use a kernel to transform the feature space. In the context of machine learning, SVMs can be recast as neural network (however, with a sophisticated training algorithm), and they can be used as binary classifiers or for regression estimation.

Large-margin hyperplane separation The task of training a linear classifier can be seen as finding a separation hyperplane in feature space. Our training samples (i.e. feature vectors) can be represented as data points in an N-dimensional space, where N is the number of features involved. Giving the coordinates of each sample, the feature vectors points into this space. We consider the binary classification case, where each sample belongs to one of two classes (+1, −1). A hyperplane divides the feature space, and it can be used to separate all or most of the data points. Once the hyperplane is determined, we have a classifier that and thus assigns a $+1/-1$ vote to each sample. The distance from the closest point of the hyperplane to a datum will later be interpreted as regression score.

The main idea of large-margin classifiers, the group of classifiers SVMs belong to, is the following: we aim at constructing the hyperplane (given by the equation $\vec{w} \cdot x + b = 0, \vec{w} \in \mathbb{R}^N, b \in \mathbb{R}$) so that the distance between the hyperplane and the nearest point(s) is maximal (Figure 5.6). This results in the following solvable quadratic optimization problem:

Find $\vec{w} \in \mathbb{R}^N, b \in \mathbb{R}$, and $\zeta_i, i = 1, 2, ..., n$, to minimize $\frac{(\sum_i \zeta_i)^q}{n} + \lambda \|\vec{w}\|^2$
under the constraints

$$\vec{w} \cdot \vec{x}_i + b \geq 1 - \zeta_i, \text{for } y_i = +1,$$
$$\vec{w} \cdot \vec{x}_i + b \leq -1 + \zeta_i, \text{for } y_i = -1,$$
$$\zeta_i \geq 0, i = 1, ..., n.$$

ζ_i allows for a 'soft' margin: it is 0 only in the separable case. Through ζ_i, support vectors (selected feature vectors that are closest to the separation hyperplane) are chosen; ζ_i then denotes their distance to the hyperplane, so that the hyperplane vector w will not be needed, as we will see.

The length of vector $\vec{w}$ is minimized, but through a λ coefficient, it can be weighted against the influence of noise in the training data.

In this optimization problem, time complexity depends on the number of samples n supplied, not on the number of features given, which makes SVMs suitable for rich-feature problems.

Figure 5.6 shows an example of large-margin linear separation. Note that the above formula – and also the actual classifier that is described lateron – only depends on the calculation of a dot product of vectors. This allows us to generalize to cases where data is not separable by means of a linear hyperplane.

The kernel trick Optimal separation is not necessarily linear: in the simplified two-dimensional case, we could separate data with a non-linear function better. SVMs make use of a *kernel* k to transform non-linear feature vectors into linear feature space F, before applying the training or classification algorithm:

$$\Phi : \mathbb{R}^N \rightarrow F$$

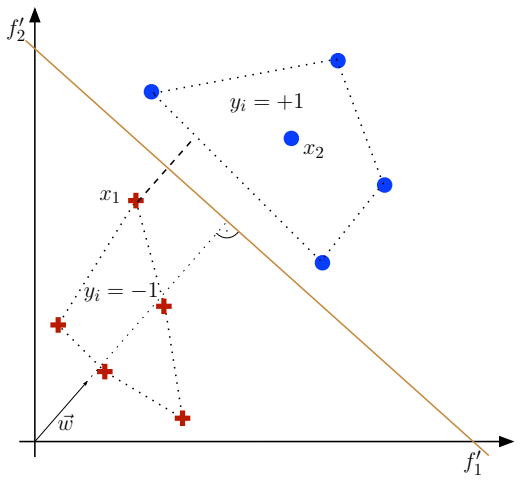


Figure 5.6.: Linear large-margin classification in the non-separable case in two-dimensional feature space. For two-dimensional space, a hyperplane is a line.

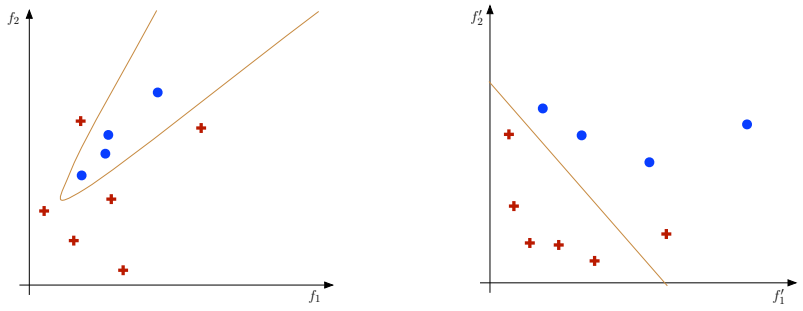


Figure 5.7.: A kernel can transform non-linear feature space (left) into linear feature space (right). (Here: fictious kernel.)

Figure 5.7 visualizes this scheme. The kernel integrates transformation and calculation of the dot product of two vectors:

$$k(\vec{x}, \vec{y}) := \Phi(\vec{x}) \cdot \Phi(\vec{y})$$

This allows for some optimization: the kernel is recast as a simple mapping function Φ that is unexpensive to compute. The kernel we use in our approach is a *Radial Basis Function* kernel:

$$k(\vec{x}, \vec{y}) = exp(-\|\vec{x} - \vec{y}\|^2/(2\sigma^2))$$

σ is a parameter which can be used to steer the generalization trade-off between precision and recall in the classifiers.

Training algorithms We have seen that a quadratic optimization problem is to be solved during the training of a support vector machine. While the number of features does not have exponential influence on training time, the number of training samples appears in the optimization problem as severe factor. Specialized training algorithms such as SVMLight (Joachims, 1998) and SVMTorch (Collobert & Bengio, 2001) allow for a fast, iterative selection of *support vectors* from the training data. SVMTorch gives a training time complexity of $O(n^2)$.

5.4.2. Solving

Solving is a computationally little expensive operation. We add up the support vectors selected during the training process via ζ_i. Solving time depends linearly on the number of support vectors:

$$f(\vec{x}) = \sum_i \zeta_i(\vec{x} \cdot \vec{x_i}) + b$$

For binary classification, $sign(f)$ gives the class. In our case, we use f as score.

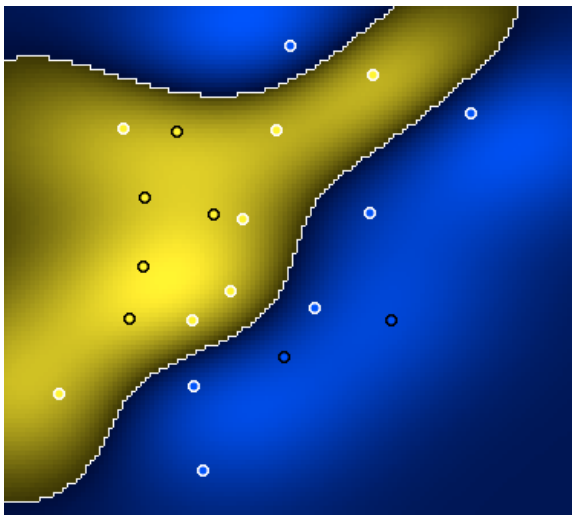


Figure 5.8.: A separable problem with 20 samples analyzed using a radial basis function kernel ($\sigma = 10$). 6 support vectors in class (-1) and 7 support vectors in (+1) were identified (white outline). (Generated with software courtesy of the AT&T Speech and Image Processing Services Research Lab.)

5.4.3. Parameter tuning

SVMs offer two parameters that should show significant effect on the classification performance. With a higher C coefficient, generalization capacity rises, and overfitting can be avoided. (Overfitting occurs when a classifier performs very well on the training data but poorly on the test set. In this case, the training algorithm does not generalize enough.) Sec-

ondly, the kernel – in our case, a gaussian kernel – can be configured. As mentioned before, a parameter σ defines the mapping from non-linear feature space. (Duan et al., 2001)

C and σ for our approach have been determined through iterative regression testing. For the evaluation of the approach, C is set to 25 and σ is set to 35. We found these values to work well.

5.4.4. Mapping multi-class problems to binary ones

SVMs are binary classifiers (dichotomizers), but we need to assign each classification instances a relation out of more than two possible relations. One very common solution to this challenge is the one-against-all strategy. This means training k binary classifiers if k relations are known. During solving, all k binary classifiers are run, and the best-scoring relation is chosen. We call the ensemble of k classifiers a polychotomizer or multi-class classifier. This technique is integrated in the parsing algorithm.

There are several alternatives to the one-against-all method. The pairwise classification strategy trains binary classifiers on all existing pairs of classes, which leaves us with $\frac{(k-1)k}{2}$ binary classifiers. Because each of the classifiers trains on only a subset of the data and the training time complexity is worse than linear, this scheme can actually work quicker. (However, with 60 relations, as in the English corpus after pruning, we would still have to train 1770 classifiers.) Another way of implementing the polychotomizer is *Error-correcting output coding* (Dietterich & Bakiri, 1995): we train several binary classifiers (M). For each of them, some of the original classes are selected and marked as +1, all others -1. This can be represented as vector $\{+1, -1\}^k$. This way, each classifier is trained on a different partition of the training set. The process yields a *decoding matrix* $D = \{+1, -1\}^{k \times \|M\|}$. During classification, each classifier is run, giving us a vector $\vec{v}$ of responses. In the ideal case (all classifiers return an error-free decision), $\vec{v}$ equals one of the rows in D. If one or more classifiers are in error, the closest match is chosen (smallest Hamming distance). The crucial algorithm here is the one that chooses the best M. For the case of margin-classifiers that return not just a binary classification decision, but a distance, Allwein et al. (2000) and Passerini et al. (2002) propose more elaborate methods.

Schölkopf & Smola (2002) note that none of the aforementioned methods generally outperforms the others. In the parsing algorithm, we apply the one-against-all strategy for reasons of simplicity, leaving error-correcting output coding open to be evaluated.

6. System Architecture

6.1. Introduction: tool chain concept

Data retrieval, conversion and interface tools are largely implemented in Perl. Almost all of them generate and augment URML data. The rhetorical analyzer uses a machine-learning framework (C++, C) which can interface several approaches to learning. Tool chain architectures have often been employed, often in conjunction with SGML or XML annotation in data-intensive linguistics (Christ, 1994; Cunningham et al., 1996; McKelvie et al., 1997).

6.2. Part-of-speech tagger

Part-of-speech (POS) tagging is accomplished using Brants (2000). In general, the tagger achieves an accuracy of > 97 percent for German data. It is trained on the NEGRA corpus (Skut et al., 1997). For the English corpus, the tagger is trained on the Susanne corpus (Sampson, 1993).

To designate POS for each token in URML, we use an additional XML tag. The resulting XML syntax specializes the basic URML; as a consequence, it is fully backwards-compatible.

6.3. Segmentizer

The segmentizer works on the POS data and on punctuation; it is implemented in Perl.

6.4. Rhetorical analyzer

6.4.1. Accessing and indexing the Document Object Model

The rhetorical analyzer makes heavy use of the Document Object Model (DOM) to work with the XML data. It is implemented in an object oriented fashion and provides reusable classes for accessing the XML corpus database. We use Xerces (`http://xml.apache.org`) as XML parser, which provides the DOM interface. However, Xerces does not provide indexing for fast access of the data. This is handled in a separate Docs class, through which the DOM is accessed. Besides the Hash-based indexing of all documents and document nodes, word indexes exist for each relation node of a document. This speeds up the checking of cue phrase based features and of topic chains.

6.4.2. Machine learning framework

Our machine learning library DRSVMlib used has been previously tested and employed (Reitter, 2002b) and allows interfacing two different SVM libraries, namely SVMTorch (Col-

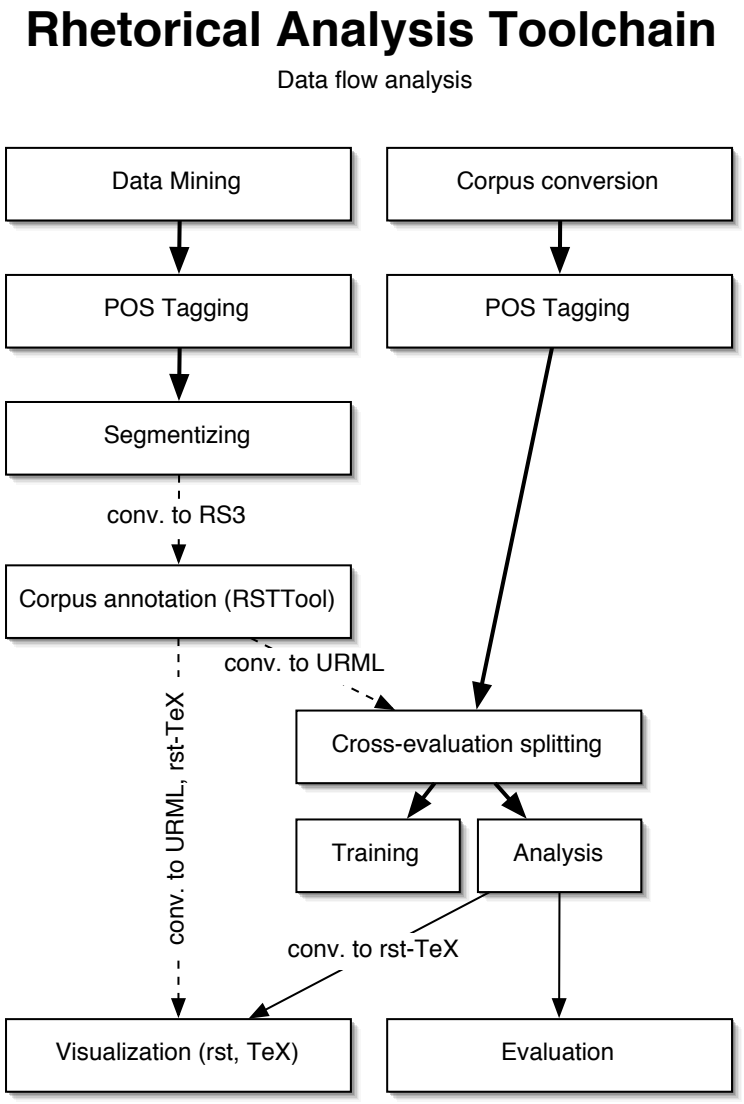


Figure 6.1.: Most tools work with the URML format that allows incremental annotation on various linguistic levels. The corpus annotation tool requires a special format. The visualization tool operates with a TeX conforming format.

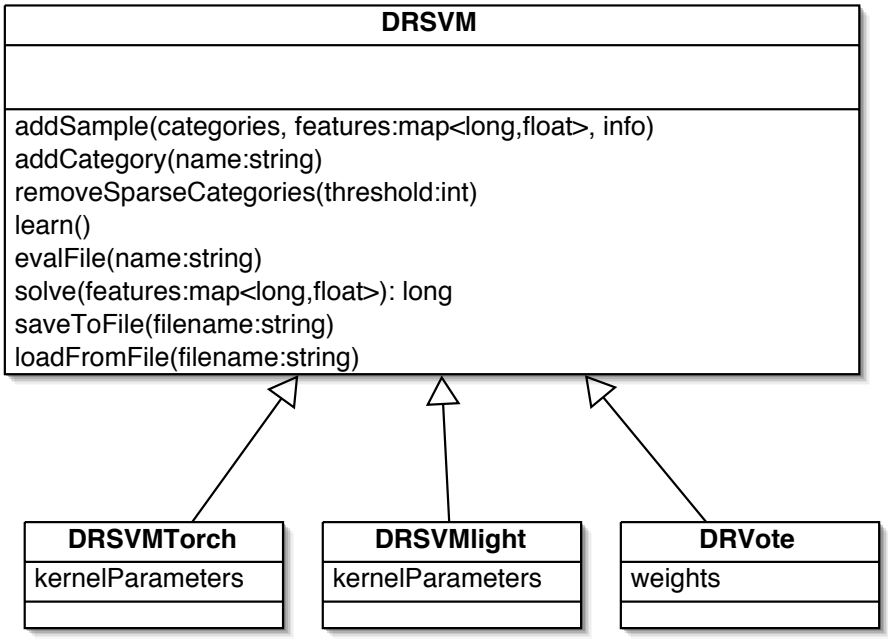


Figure 6.2.: Class diagram describing the architecture of the machine learning framework.

lobert & Bengio, 2001) and SVMlight (Joachims, 1998). DRSVMlib provides a set of C++ classes. The class Corpus encapsulates the collection of sample data, storing and retrieving the data. It is used in learning and evaluating and can run evaluation procedures on the data. DRSVM is an abstract class (see Figure 6.3) that represents the classifier. It is specialized as DRSVMTorch, DRSVMlight and DRSVMVote. The first two act as wrapper classes around the original implementations of SVMTorch and SVMlight. The latter is a simple classification algorithm that calculates maximum-likelihood probabilities for single feature-class combinations and lets these classifiers vote to achieve a classification result.

6.4.3. Parsing

The parsing algorithm keeps a chart (class Chart) to store active nodes during the first phase (beam-search based on local classification). In the second phase, only full well-formed trees are considered and scored. Therefore, the chart provides efficient mechanisms to select a subset of all edges. For an overview of the learning and parsing architecture, see Figure 6.3.

DOM nodes are created along with the edges in the chart, as the feature checking methods in the Corpus class always operate on the DOM. Also, serialization is possible at any point.

6.5. Visualization and conversion tools

The rst package for LaTeX is a visualization tool to enable comfortable typesetting of corpus data and rhetorical tree structures (Appendix C). It is implemented in TeX and recursively evaluates rhetorical descriptions to draw diagrams. An environment rhetoricaltext displays tagged text segments that can be referred to from other places in the document.

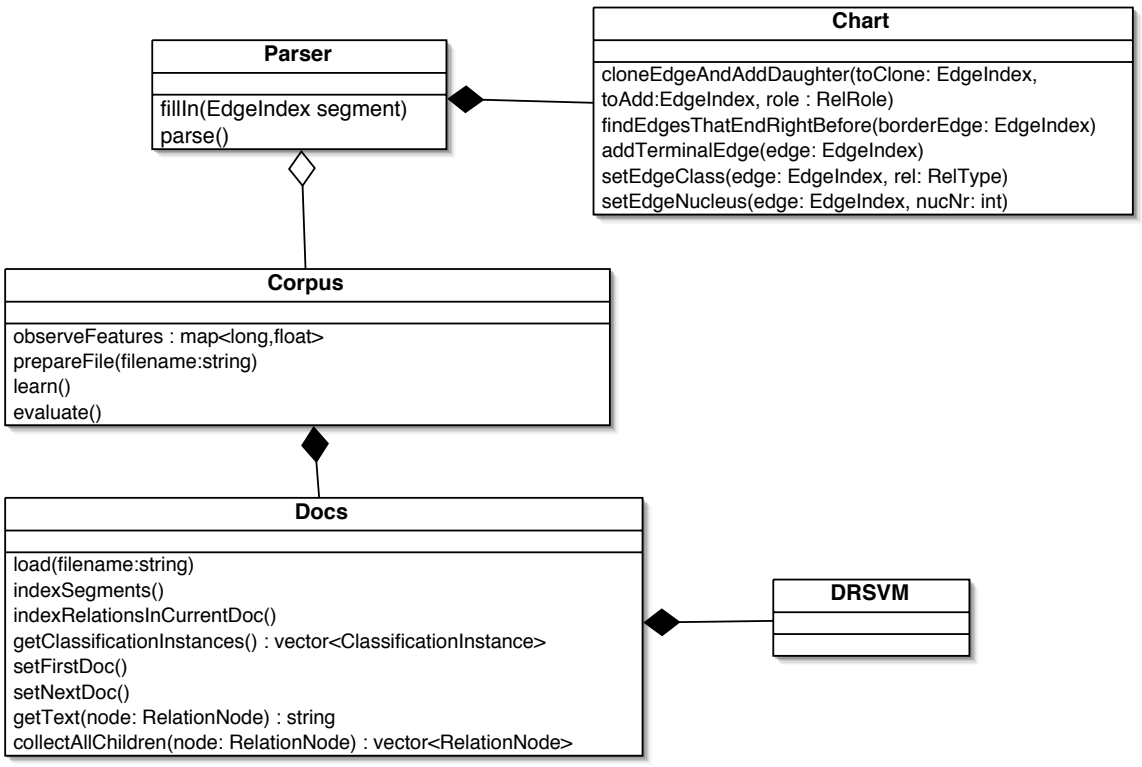


Figure 6.3.: Class diagram of the architecture used to implement the rhetorical parser.

We provide an application to convert sets of documents created with RSTTool from the RS3 format to URML. It was used during the corpus creation. Another program converts the english language corpus (Carlson et al., 2001) from its LISP-style format to URML. Both tools detect the use of RST schemata (see 2.5.2), where several relations share a nucleus in order to issue a warning.

7. Evaluation

7.1. Results

The rhetorical analysis algorithm is subject to extensive evaluation.

We focus on the central – and hardest – task the analysis system faces. We evaluate the assignment of a relation to a group of text spans. Each known relation node in the test corpus is subject to the multi-class classification through the ensemble of binary classifiers which works on all classification instances resulting from the relation. The feature set for these classifications is the global set. The best scoring decision is taken (as in the parsing algorithm).

For each relation x, we give *recall*, a measure that tells us, how many of the relation nodes from the corpus labeled with x are detected correctly by the binary classifier. *Precision* indicates how many of the classification instances labeled x by the analyzer are correctly labeled x.

The reason for this choice of evaluation mode is that we see the evaluation of the support vector classification approach as the foremost goal of the implementation.

Classifiers are trained on 240 documents and evaluated on 50 documents from the LDC corpus. The tests are 2-fold cross-validated. In this respect, the multi-class accuracy of our implementation is 61.0 percent (Table 7.2). Semi-automatic regression tests for parameter estimation of the SVM training algorithm and the SVM kernel were carried out on different test partitions. For the Potsdam Corpus, accuracy is significantly lower (39.1 percent). This can be explained by the fact that the training partition of the LDC corpus yielded an average of 7976 classification instances, while only 1943 training samples are derived from the Potsdam Corpus. With this amount of data, classification accuracy for the LDC corpus is 56.0 percent (see Figure 7.1).

The evaluation of single feature classes in terms of their contribution to relation assignment is of similarly important interest. We give data from a *leave-one-out* test (Table 7.1). Learning and classification steps are run on the partioned test and training set, however, during each run, one feature class is left out. The decrease in performance for each run indicates the specific contribution of a feature to the classification performance. The table shows the reduction in error rate (difference of accuracy divided by error rate with all features enabled). We evaluate the analyzer with 10022 samples out of the LDC corpus in two tries with 100 training documents and 90 test documents in each try. The Potsdam Corpus is evaluated with 935 samples in four tries with 138 training documents and 35 test documents. Due to its size, the data given is less significant.

Learning curve: Figure 7.1 shows how the performance of the system increases with the number of training samples, i.e. classification instances. For these evaluations, we used the following partitioning of the available data: 80 percent of all annotated documents for training and 20 percent for testing. From the partitions, subsets of different sizes are

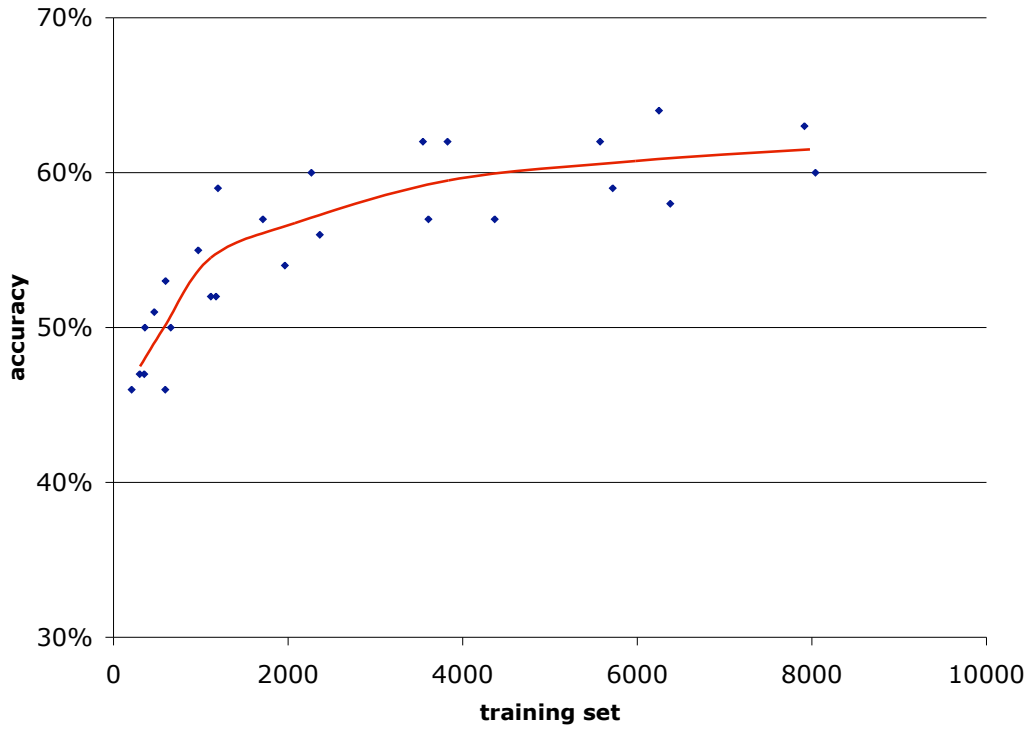


Figure 7.1.: Learning curve for LDC corpus showing gain in accuracy with number of training samples.

chosen to produce an accuracy rating. For cross-validation, the experiment is repeated with different randomly chosen data sets of the same size.

7.2. Analysis

We turn to a comparison of the results to a state-of-the-art approach. The relevant measure used by Marcu (2000) is *labeled precision & recall* for parsing results with manually selected, thus perfect segmentation: 57.9 percent precision, 56.3 percent recall. We automatically segment texts for both the gold standard (corpus) and automated analysis, so standards compare in this respect. However, our tests look at structurally correct relations only, so the relevant measure to multi-class accuracy is Marcu's precision.

To put the accuracy figures into perspective, we calculate a baseline value. The baseline is established by having the classifier ensemble always chose the most common relation, ELABORATION. It is 33.6 percent for the corpus used. As a theoretical upper bound for analysis performance, we can consider inter-annotator agreement in corpus collection efforts. Carlson et al. (2001) report kappa coefficients ranging from 0.62 to 0.80. Here, a kappa value of 0 indicates random agreement, 1 perfect agreement. The agreement was reached only after rigorous training and at the end of their corpus collection effort. Even

Table 7.1.: How important are the features? [percent]

Relation	Noun	POS	Punct	TopSim	Cue	Len$_{2nd}$	Len$_{1st}$
LDC Corpus (English)							
Attribution	0.6	0.2	15.9	0.5	11.6	15.1	
Background	0.7	-0.2	4	0.7	-3.1	1	
Comparison	-2.3	0	1.7	4	70.7	-7.5	
Condition	0	-1.4	-0.7	2.7	55.8	-2.7	
Contrast	2.1	0.2	2.3	2.3	29.7	-0.5	
Elaboration	-0.4	-0.2	2.6	-0.2	3.5	-2.3	
Enablement	0	0.6	0.9	1.5	10.6	-1.2	
Evaluation	-1.2	-0.2	-2.3	-2.1	8.5	-6.6	
Explanation	0	-1.2	2.2	-1	20.9	-5.1	
Joint	-0.1	-0.6	-2.4	-2.5	15	-6.5	
Reason	0	0	0	0	77.8	-22.2	
Summary	2.8	1.9	6.5	5.6	65.7	6.5	
Temporal	-1.9	1.9	1.1	1.1	23	-0.7	
Textualorganization	0	0	3.4	-2.3	4.5	9.1	
Topicchange	-9.9	4.2	7	-11.3	-4.2	1.4	
multiclass	-0.1	-0.1	2.8	-0.1	15.3	-0.7	
Potsdam Corpus (German)							
Cause	1.1	1.1	-5.6	-1.1	13.3	-6.7	-18.9
Concession	0	-1.7	0	1.7	36.2	-5.2	5.2
Conclusive	0	0	0	0	33.3	33.3	66.7
Condition	0	0	0	4.2	-8.3	-4.2	8.3
Conjunction	0	0	0	0	0	0	0
Contrastive	0	0	0	-3.8	19.2	-11.5	3.8
Evaluation	1.4	0	1.4	1.4	2.7	-0.7	6.1
Evidence	0	0	2.8	-2.8	2.1	-7.8	0.7
Framework	0	0	1.7	0	12.1	-5.2	-9.5
Interpretation	0	0	0	0	0	0	0
Joint	0	0	0	-8.3	16.7	-12.5	-12.5
Preparation	-4.1	0	2.7	-4.1	6.8	-6.8	-6.8
Result	0	0	2.1	4.2	14.6	4.2	20.8
Sequential	-0.9	0	0.9	0.9	-10.8	4.5	-1.8
Specification	-0.3	0	0.3	-4.1	-2	-7.1	-8.4
multiclass	-0.2	0	0.7	-1.4	5.4	-4.5	-2.9

Table 7.2.: Relation assignment performance [percent]

LDC Corpus (English)		
Relation	Precision	Recall
Attribution	64.3	79.4
Background	9.8	3.6
Comparison	60	6.4
Condition	48.6	45.9
Contrast	49.2	44.1
Elaboration	70.8	89.8
Evaluation	16.7	6.7
Explanation	33.3	24.8
Joint	46.3	52.7
Summary	45	26.5
Temporal	32.1	13.6
TextualOrg	33.3	20
TopicChange	28.6	36.4
Multi-class	Accuracy 61.8	

Potsdam Corpus (German)		
Relation	Precision	Recall
Cause	20.6	13.6
Concession	18.5	9.6
Condition	25	10.5
Contrastive	14.3	5.4
Evaluation	31.6	29.3
Evidence	28.8	24.1
Framework	27.3	25
Joint	16.7	5.1
Preparation	35.3	50.9
Result	10	2.5
Sequential	31.3	28.4
Specification	52.2	75.2
Multi-class	Accuracy 39.1	

Table 7.3.: Training time per classifier [sec]

Background	427
Cause-result	292
Comparison	94
Condition	94
Contrast	371
Elaboration	2456
Enablement	137
Evaluation	222
Explanation	386
Joint	714
Manner-means	84
Reason	4
Same-unit	172
Summary	62
Temporal	236
TextualOrg	25
Topicchange	47
Topic-comment	61
Average	323

then, as the kappa value indicates, there were several instances of disagreement among human annotators.

Our analyzer achieves its accuracy with a well-defined classification method based on large-margin separation and a sufficiently large data-set. Our English evaluation corpus is, though presumably similar, not the corpus used in Marcu's experiments. As the density of rhetorical signals and the distribution of rhetorical relations differ greatly between text genres and corpora, quantitative comparison seems difficult. Nevertheless, the accuracy rate indicates excellent performance.

Examining the classification results for single relations (Figure 7.2), we find that certain rhetorical relations are recognized with relatively high precision & recall, in particular: Elaboration/Specification and Attribution. Little surprisingly, these are the most-frequent relations in the LDC corpus. More important than that, cue phrases are unlikely to signal these relations. Attribution and Elaboration can be distinguished with information from the punctuation feature.

Indeed, an experiment leaving out single features (Figure 7.1) shows that punctuation is, after cue phrases, the feature that contributes most to overall classification accuracy. Some harder problems, such as detecting the less frequent and ambiguously signaled Cause-Result relation, are not solved as well by the ensemble of binary SVM classifiers. Here, further research is necessary to determine the usefulness of Error-Correcting Output Coding instead of a one-against-all training, as suggested by Allwein et al. (2000). Another approach to investigate is training specific classifiers to disambiguate single rhetorical classifiers.

For some features and specific relations, the tests show even a decrease in performance (e.g. 7.5 percent for the Comparison relation and the Len$_2$*nd* feature. This can be explained by noise introduced by irrelevant features. The noise/signal ratio is particularly high for certain relations, such as Reason, where only few examples are present in the corpus.

The learning curve converges with the maximum number of training samples used, the corpus seems to be large enough for this kind of training task. The Potsdam Corpus contains a lower number of training samples. The 18 binary SVM classifiers can be trained on 150 documents in reasonable time (216 min on 1 Mhz G4-CPU).

7.3. Where do we go from here?

Looking at the evaluation data, we can define certain areas of interest for further work. It seems obvious that a fully automatic structural analysis with relation assignment is unlikely to reach usability without major improvement of the classifiers. As a consequence, the optimization of the core classification system seems to be the first goal.

Improving the quality of features. We have shown that many of the feature heuristics do not contribute significantly to the results. For the topic similarity measure, this fact might be due to missing integration of a semantic network which would supply the algorithm with a list of hyponyms, hypernyms and synonyms. Also, a pronoun binding algorithm might add

Feature pre-selection While the training algorithm discards irrelevant features, it cannot overcome the sparse data problem as described before. This leads to situations where features lead to a decrease in performance. A hard-coded, relation-specific feature filter based on linguistic knowledge may improve performance.

Using SVM classifiers to disambiguate discourse markers. Training a classifier for each discourse marker is an approach we have not investigated in this thesis.

Improving the Potsdam Corpus. The classifiers showed a significantly worse performance on the German corpus. Further validation and extension of the corpus may provide for a credible comparison between languages.

8. Contributions and Conclusion

The work presented contributes to the research in rhetorical analysis in various ways: The Potsdam Corpus contains 173 newspaper opinion texts, manually annotated and validated with a standardized set of rhetorical relations. This is, to our knowledge, the first rhetorical corpus in German language.

Underspecified Rhetorical Markup Language (URML) is a novel flexible, XML based format that provides interfacing between tools in rhetorical analysis and corpus annotation. As a stand-alone data format, URML is applied in the collection of a novel corpus of news commentaries.

Rhetorical data can be visualized through the rst package for LaTeX(Appendix C). It enables users of LaTeXto create, edit and typeset tree-like rhetorical structure diagrams and corpus excerpts conveniently. It is the first and only package to do this.

On the implementation side, a C++ wrapper library provides a reusable common interface to C-based classification algorithms and a voting algorithm.

All these elements work constitute a framework for the rhetorical analysis of texts. Its instantiation is able to integrate a potentially high number of features using Support Vector Machines. We show that it can assign rhetorical relations with an excellent accuracy.

Furthermore, we demonstrate a two-stage parsing algorithm that makes use of statistical classifiers and the underspecified representation shown.

The analysis of features shows that only a few shallow feature types contribute to overall recognition performance. The analysis of strengths and weaknesses in rhetorical analysis suggests that further investigations should be based on both, corpora-driven learning and evaluation and an abstract, dynamic model of rhetorical relationships that explains coherence phenomena from a supra-semantic perspective.

A. Glossary: Parsing Framework

Classification Instance A classification instance is a data structure that consists of a *first span*, which is a list of relation and segment nodes, and a *second span*, which is a single relation or segment node. This data structure is used to create a binary view onto the rhetorical structure, which can have non-binary (multi-nuclear) nodes.

Classifier A classifier is a function $f : \mathbb{R}^N \rightarrow [-1, +1]$ which assigns class estimation to a given feature vector. In a machine learning framework, it is trained using training data. This data is a set of feature vectors that are annotated with a class:
$(x_1, y_1), ..., (x_n, y_n) \in \mathbb{R}^N \times \{-1, +1\}$

Feature A local feature is a function $C \rightarrow [-1, 1]$. It observes a linguistic property in a given classification instance and returns a scalar (or binary) value. We often use the term in the sense of "property" rather than function. Some features are automatically determined from the training corpus, for example potential cue phrases. Other *features* are hard-coded. These include "Size of second text span", "Lexical similarity of first and second spans". A global feature takes the context of a classification instance in a rhetorical analysis into account.

Feature vector A feature vector is an ordered list of feature values, $x \in \mathbb{R}^N$ (with N being the number of existing features). It is the result of the observation of all existing features in a classification instance.

Nucleus Text span referenced from a relation. The nucleus usually carries the more significant utterance, while satellites can be removed from the text without disguising the central intent of the author. (In URML: <nucleus id="15"/>.)

Relation One of a set of possible rhetorical relations that describe the relationship of two or more text spans. Examples are ELABORATION, CONCESSION, VOLUNTARY-RESULT.

Relation node A relation node is a data structure that consists of references to two other relation nodes (nucleus and satellite) in the case of a *hypotactic relation node*, or a list of references to at least two relation nodes in the case of a *paratactic relation node*. Furthermore, the data structure contains information about the rhetorical relation (CAUSE, CONTRAST, etc.) that holds between the nodes that are references. (In URML: <hypRelation> or <parRelation>.)

Relation type There are two types of relations: *Hypotactic* relations, which connect a nucleus and a satellite, and *paratactic* relations, which connect several nuclei.

Satellite Text span that is referenced from a relation.
(In URML: <satellite id="15"/>.)

Segment A segment is a terminal node in the tree that represents a rhetorical analysis. It contains a minimal discourse unit. (In URML: ... .)

B. An Extended URML Document

In the following, a sample URML document is shown. This document is taken from the Potsdam Corpus and contains part-of-speech annotations (in the first discourse unit) and meta-data.

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE urml SYSTEM "urml-extended.dtd">
<urml>
<header>
<reltypes>
<rel name="cause" type="hyp" />
<rel name="concession" type="hyp" />
<rel name="conclusive" type="hyp" />
<rel name="condition" type="hyp" />
<rel name="conjunction" type="par" />
<rel name="contrastive" type="par" />
<rel name="disjunction" type="par" />
<rel name="evaluation" type="hyp" />
<rel name="evidence" type="hyp" />
<rel name="framework" type="hyp" />
<rel name="interpretation" type="hyp" />
<rel name="joint" type="par" />
<rel name="means" type="hyp" />
<rel name="motivation" type="hyp" />
<rel name="otherwise" type="hyp" />
<rel name="preparation" type="hyp" />
<rel name="purpose" type="hyp" />
<rel name="result" type="hyp" />
<rel name="sequential" type="par" />
<rel name="specification" type="hyp" />
<rel name="specification-mn" type="par" />
<rel name="unconditional" type="hyp" />
<rel name="unless" type="hyp" />
<rel name="unstated-relation" type="hyp" />
</reltypes>
<postypes>
<pos name="\$(" />
<pos name="\$," />
<pos name="\$." />
<pos name="ADJA" />
<pos name="ADJD" />
<pos name="ADV" />
<pos name="APPO" />
<pos name="APPR" />
<pos name="APPRART" />
<pos name="APZR" />
<pos name="ART" />
<pos name="CARD" />
<pos name="FM" />
```

```
<pos name="KOKOM" />
<pos name="KON" />
<pos name="KOUI" />
<pos name="KOUS" />
<pos name="NE" />
<pos name="NN" />
<pos name="PDAT" />
<pos name="PDS" />
<pos name="PIAT" />
<pos name="PIDAT" />
<pos name="PIS" />
<pos name="PPER" />
<pos name="PPOSAT" />
<pos name="PRELAT" />
<pos name="PRELS" />
<pos name="PRF" />
<pos name="PROAV" />
<pos name="PTKA" />
<pos name="PTKANT" />
<pos name="PTKNEG" />
<pos name="PTKVZ" />
<pos name="PTKZU" />
<pos name="PWAT" />
<pos name="PWAV" />
<pos name="PWS" />
<pos name="TRUNC" />
<pos name="VAFIN" />
<pos name="VAINF" />
<pos name="VAPP" />
<pos name="VMFIN" />
<pos name="VMINF" />
<pos name="VMPP" />
<pos name="VVFIN" />
<pos name="VVIMP" />
<pos name="VVINF" />
<pos name="VVIZU" />
<pos name="VVPP" />
<pos name="XY" />
</postypes>
</header>
<document id="maz3377">
<info><source>Maerkische Allgemeine Zeitung 3377 09.10.2001</source>
<section> 09.10.2001  VOR ORT : HAVELLAND : KOMMENTAR
</section ><suptitle> ANDRE WIRSING</suptitle>
<title> Absicht </title >
</source></info>
<text>
<segment id="maz3377.0">
<sign pos="NN">Absicht</sign> <sign pos="\$.">.</sign> </segment>
<segment id="maz3377.1">Erst rührt sich niemand unter den Dallgower
    Kommunalpolitikern , </segment>
<segment id="maz3377.2">nun überschlagen sich alle mit Anträgen zur
    Gemeindereform und den vorausgehenden Verhandlungen mit den südlichen
    Nachbarn . </segment>
<segment id="maz3377.3">Nicht klar wird zum jetzigen Zeitpunkt , </segment>
```

```
<segment id="maz3377.4">welche Absichten hinter dem plötzlichen Aktionismus
    stehen : </segment>
<segment id="maz3377.5">Will die SPD-Fraktion - bisher völlig uneins über
    Dallgows Zukunft - wirklich faire Verhandlungen anstreben ? </segment>
<segment id="maz3377.6">Warum stellt sie dann schier unannehmbare
    Bedingungen , indem sie den Bürgermeister mit einem solch eng geschnürten
     Verhandlungsmandat ausstattet , dass dieser nur noch das Wie der
    Eingemeindung klären darf . </segment>
<segment id="maz3377.7">Warum beharrt die CDU auf einer neuerlichen
    Einwohnerversammlung ? </segment>
<segment id="maz3377.8">Warten die Christdemokraten nur darauf , genügend
    Claqueure zusammenzubekommen , die sie in ihrer Ablehnung zu jeglichen
    Fusionen bestätigen ? </segment>
<segment id="maz3377.9">Allein eine Fraktion , die der Freien
    Wählergemeinschaft , ist bisher mit einer einheitlichen und
    nachvollziehbaren Argumentationslinie aufgetreten , </segment>
<segment id="maz3377.10">bekennt sich zum Dreierbund und lässt Spielraum für
     die Art des Zustandekommens . </segment>
<segment id="maz3377.11">Die anderen werden am nächsten Mittwoch beweisen
    müssen , dass sie es ehrlich meinen . </segment>
</text>
<analysis status="interpretation">
<info><editor job="annotate" date="05.02.2002">Reitter</note></info>
<parRelation id="maz3377.1000" type="sequential">
  <nucleus id="maz3377.1" />
  <nucleus id="maz3377.2" />
</parRelation>
<parRelation id="maz3377.1001" type="conjunction">
  <nucleus id="maz3377.10" />
  <nucleus id="maz3377.9" />
</parRelation>
<hypRelation id="maz3377.1002" type="cause">
  <nucleus id="maz3377.11" />
  <satellite id="maz3377.1003" />
</hypRelation>
<parRelation id="maz3377.1003" type="contrastive">
  <nucleus id="maz3377.1000" />
  <nucleus id="maz3377.1004" />
</parRelation>
<parRelation id="maz3377.1004" type="contrastive">
  <nucleus id="maz3377.1001" />
  <nucleus id="maz3377.1006" />
</parRelation>
<parRelation id="maz3377.1005" type="sequential">
  <nucleus id="maz3377.1008" />
  <nucleus id="maz3377.1009" />
</parRelation>
<hypRelation id="maz3377.1006" type="specification">
  <nucleus id="maz3377.1007" />
  <satellite id="maz3377.1005" />
</hypRelation>
<parRelation id="maz3377.1007" type="joint">
  <nucleus id="maz3377.3" />
  <nucleus id="maz3377.4" />
</parRelation>
<parRelation id="maz3377.1008" type="contrastive">
```

```
  <nucleus id="maz3377.5" />
  <nucleus id="maz3377.6" />
</parRelation>
<parRelation id="maz3377.1009" type="contrastive">
  <nucleus id="maz3377.7" />
  <nucleus id="maz3377.8" />
</parRelation>
</analysis>
</document>
</urml>
```

C. Rhetorical Theory in LaTeX with the rst Package

C.1. Motivation

Drawing rhetorical analyses is no fun when you need to change and update diagrams as you refine your work, or, more importantly, if a lot of analyses are to be drawn. Previous solutions include O'Donnell's RSTTool application (O'Donnell, 2000), which can output (encapsulated) postscript graphics. However, these drawings are more suitable for reference than for use in publications. Besides, pdfTeX has a a notorious problem with postscript-items in source files.

Voilà, there we go: This package enables us to typeset beautiful diagrams with no hassle. It is oriented towards the style of the diagrams shown in Mann & Thompson (1988) and subsequent works.

Further developments may include a conversion tool to generate the rst format automatically from RSTTool files or from the LISP-based format used in Carlson et al. (2001).

Please refer to the last section for copyright and usage information.

C.2. Installation

The file rst.sty is almost all you need. The package uses three additional packages, color, which are usually installed in a well-equipped TeXsystem. If not, download and install them:

- color[1]
- ifthen
- calc

Install rst.sty with the usual mechanisms of your TeXdistribution or simply put it in the same directory where you keep the document that makes use of the rst package.

C.3. The rhetoricaltext environment

To give examples from a corpus, you can use the rhetoricaltext environment. Every environment is assigned its own identifier, counted as roman numeral.

[1] You can probably eliminate the need for the color package by removing the usepackage{color} command and uncommenting four \providecommand statements in the rst.sty file.

In the environment, single *minimal discourse units* are specified with the \unit command. It takes two arguments: the first one is optional and specifies a label for this unit. This label may be used to refer to the unit later on. The second argument to \unit contains the text.

An additional command is the \source command, which takes one argument. This usually identifies the exact source of the sample, e.g. an identification number in the corpus used.

You may refer to any given unit from any part of your document using the \refr command. It takes one argument, which should be the label of a unit defined somewhere else in the text. In case you make forward references, i.e. you refer to a unit that is defined later in the document, you will need to compile the document twice to get rid of the ?? appearing in the output. This is the same mechanism as with other kinds of references in LaTeX.

Here's an example:

```
In the following, \refr{bush2} is an embedded discourse unit.

\begin{rhetoricaltext}
\unit[bush1]{The Bush Administration,}
\unit[bush2]{trying to blunt growing demands from
Western Europe for a relaxation of controls on exports to the
Soviet bloc,}
\unit[bush3]{is questioning...}
\source{wsj$_{2326}$}
\footnote{Taken from the corpus described in \cite{carlson-corpus}.
Bracket annotation simplified.[...]}
\end{rhetoricaltext}

How do we treat the noncontiguous discourse unit
formed of \refr{bush1}, \refr{bush3}?
```

In the following, is an embedded discourse unit.

[The Bush Administration,]^{11A} [trying to blunt growing demands from Western Europe for a relaxation of controls on exports to the Soviet bloc,]^{11B} [is questioning...]^{11C} (wsj$_{2326}$) [2]

How do we treat the noncontiguous discourse unit formed of 11A, 11C?

C.4. Rhetorical structure diagrams

The rst package provides three types of diagram elements via the commands \dirrel, \multirel and \rstsegment.

[2] Taken from the corpus described in Carlson et al. (2001). Bracket annotation simplified. The other examples are made up. They are not intended to demonstrate any linguistic/rhetorical properties.

C.4.1. Terminal elements

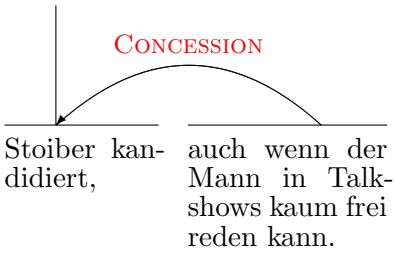
rstsegment defines a terminal-level *minimal discourse unit*: it is meant to contain normal text. This is what it expects to be given as argument. While omitting this command and simply typing the text in an \dirrel argument might work sometimes, it could give unexpected (and not-so-nice) results.

C.4.2. Directed relations

\dirrel draws a schema diagram for one or more directed (nucleus-satellite) relations. There is always one designated nucleus per \dirrel schema.

This command always takes $2n$ arguments. They should be thought of as pairs: The first argument always states a relation name, the second one contains the content of the span. Every argument pair given draws a span and a relation arrow to the nucleus argument of the schema. The nucleus is identified by leaving the relation name empty. Alternatively, you may specify \nuc if you think that's better to read. There may be up to four argument pairs.

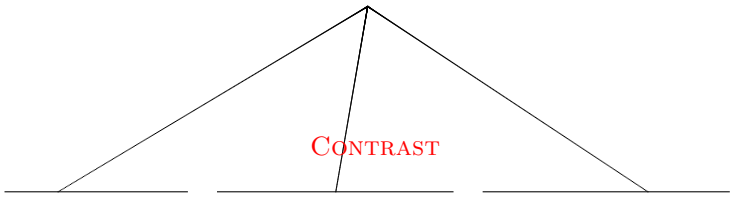
```
\dirrel{}
{\rstsegment{Stoiber kandidiert,}}{Concession}
{\rstsegment{auch wenn der Mann in Talkshows kaum frei reden kann.}}
```



C.4.3. Multinuclear relations

multirel draws a diagram for a multi-nuclear relation. It takes at least two arguments. The first argument is the relation name. All other arguments contain the contents of the spans that are connected by the relation.

```
\multirel{Contrast}
{\rstsegment{Schr\"{o}der wolle nicht wiedergew\"{a}hlt werden,
sagte er, falls er die Arbeitslosigkeit nicht reduzieren k\"{o}nne.}}
{\rstsegment{Trotzdem tritt der Kanzler wieder zur Wahl an.}}
{\rstsegment{Die Aussichten sind nicht schlecht.}}
```



Schröder wolle nicht wiedergewählt werden, sagte er, falls er die Arbeitslosigkeit nicht reduzieren könne.

Trotzdem tritt der Kanzler wieder zur Wahl an.

Die Aussichten sind nicht schlecht.

C.4.4. Configuration

The rst package knows a few configuration variables you may use to alter the appearance of the diagrams.

Width control

The \compressionWidth value is the maximum width that a single span may have in any diagram. Set it to 0pt to let not limit the width. Used on a per-diagram basis, it makes an excellent way of compression a diagram horizontally without making it look ugly. See the last example in section C.5 to get an impression of what this parameter does.

```
\setlength{\compressionWidth}{0pt}
```

The \terminalWidth value is the maximum width that a single span may have in any diagram. Set it to 0pt to let not limit the width.

```
\setlength{\terminalWidth}{100pt}
```

You may also alter the margins at the left and the right hand side of the tree with the \rstmargin value.

```
\setlength{\rstmargin}{3pt}
```

Finally, the following command sets the minimum space that's between spans in the \rstmiddleskip value.

```
\setlength{\rstmiddleskip}{1em}
```

Colors

\relnamebgcolor contains the name of the background color of relation names. Depending on other packages used, white could avoid transparency. (This functionality is disabled in the rst.sty package by default to enable the use of \raisebox commands in relation names.) \relnamecolor specifies the name of the text color of relation names.

```
\renewcommand{\relnamebgcolor}{white}%       background color
\renewcommand{\relnamecolor}{red}%          relation name color
```

Please note that these settings may or may not work, depending on your output format (Postscript or PDF). In my experiments, PDF worked generally better.

C.5. Complex structures - more examples

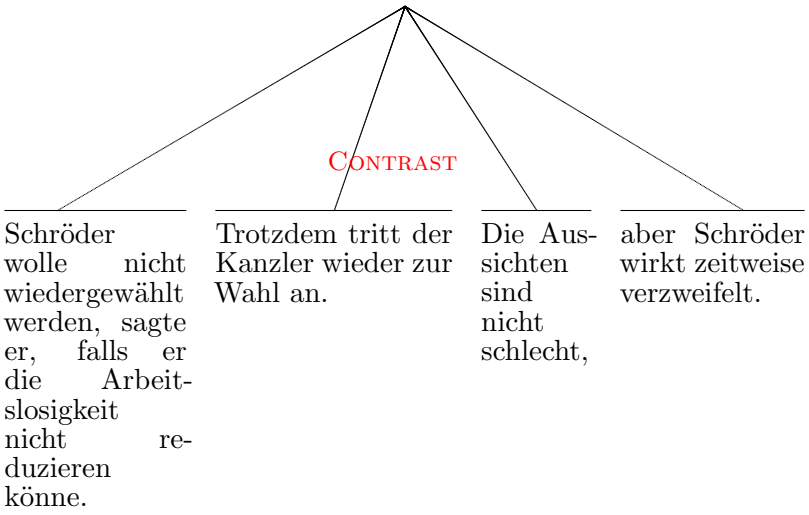
As you can see from the following examples, trees may be arbitrarily complex. In case the trees get to wide, reference numbers may be used in conjunction with the rhetoricaltext environment. Or, alternatively, you might want to consider choosing a more handy example, as readers tend to prefer smaller diagrams over unnecessarily complex ones...

Note: You may raise or lower the relation names in the diagrams with the \raisebox command in the relation name argument of directed relations. (See example below.)

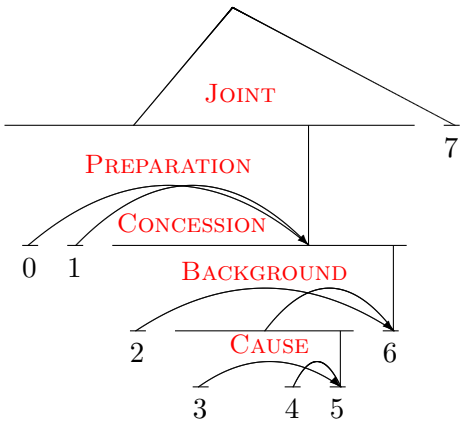
Note: Position relation names vertically by adding space at the left or right hand side. Use either the \ backslash syntax or an ordinary \hspace command. (See example below.)

Note: If you do not want to give a relation name, just specify one space with a \backslash (\ ) as shown below.

```
{\setlength{\terminalWidth}{90pt}
 \multirel{Contrast\ \ \ \ }
   {\rstsegment{Schr\"{o}der wolle nicht wiedergew\"{a}hlt werden,
                sagte er, falls er die Arbeitslosigkeit
                nicht reduzieren k\"{o}nne.}}
   {\rstsegment{Trotzdem tritt der Kanzler wieder zur Wahl an.}}
   {\rstsegment{Die Aussichten sind nicht schlecht.}}
   {\rstsegment{aber Schr\"{o}der wirkt zeitweise verzweifelt.}}
}
```

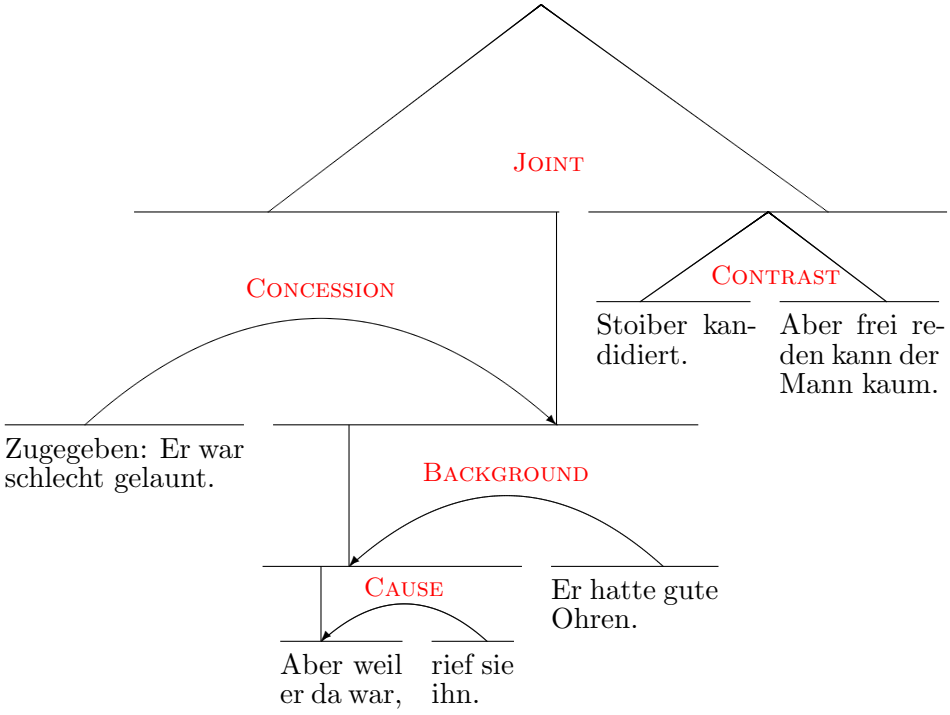



```
\multirel{Joint}{
\dirrel{Preparation}{\rstsegment{0}}
        {\raisebox{-2em}{Concession}}{\rstsegment{1}}
        {}{\dirrel{Background}{\rstsegment{2}}
                      {\ }{\dirrel{Cause}{\rstsegment{3}}
                                  {\ }{\rstsegment{4}}
                                  {}{\rstsegment{5}}}
                      {}{\rstsegment{6}}}}
        {\rstsegment{7}}}
```




```
{\hspace{30pt}\setlength{\compressionWidth}{160pt}
\multirel{Joint}
          {\dirrel{Concession}
                {\rstsegment{Zugegeben: Er war schlecht gelaunt.}}
                {\dirrel
```

```
                {Background}{\dirrel{Cause}
                    {\rstsegment{Aber weil er da war,}}
                    {}{\rstsegment{rief sie ihn.}}}
                {}{\rstsegment{Er hatte gute Ohren.}}}}
        {\multirel{Contrast}
              {\rstsegment{Stoiber kandidiert.}}
              {\rstsegment{Aber frei reden kann der Mann kaum.}}}
}
```



C.6. Known bugs

Drawing the diagrams is, unfortunately, slow. A solution to overcome this would be to use a faster machine.

The package can only draw up to four spans in one constituent, which is due to the limited number of arguments in TeX/ LaTeX.

The vertical line in compressed diagrams can be too far to the right, if the compression value is set too low. This will be improved in a later version. I recommend setting the compression value on a per-diagram basis.

The relation name drawn in multi-nuclear relations is often drawn over some lines. In a later version, I might find a way to put a white box behind the relation name. Sometimes, the Bezier curves of nucleus-satellite relations may also overlap, which is less favorable. This would be hard to change.

If the \compressionWidth length is set, the width of the diagram's bounding box may

be to small, resulting in a diagram that appears too far left. This has proven to be too complicated to be fixed. Please accomodate manually using \hspace.

Please notify me in case you encounter any other problems.

C.7. Copyright and Acknowledgements

I am grateful to Stephan Lehmke for providing his naturalparbox.sty from the TeXPower package (Lehmke, 2002).

Bibliography

Allwein, E. L., Schapire, R. E. & Singer, Y. (2000). Reducing multiclass to binary: A unifying approach for margin classifiers, *Journal of Machine Learning Research* **1**: 113–141.

Asher, N. (1993). *Reference to Abstract Objects in Discourse*, Dordrecht: Kluwer.

Berger, D., Reitter, D. & Stede, M. (2002). XML/XSL in the dictionary: The case of discourse markers, *Proceedings of the 2nd Workshop on NLP and XML (NLPXMP-2002), in: Proceedings of COLING 2002*, Taipei, Taiwan.

Billot, S. & Lang, B. (1989). The structure of shared forests in ambiguous parsing, *Proceedings of the 27th Meeting of the Association for Computational Linguistics*, Vancouver, pp. 143–151.

Brants, T. (2000). TnT – a statistical part-of-speech tagger, *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle.

Brunn, M., Chali, Y. & Pinchak, C. (2001). Text summarization using lexical chains, *Proceedings of the Document Understanding Conference*, New Orleans, pp. 135–140.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery* **2**(2): 121–167.

Carlson, L. & Marcu, D. (2001). Discourse tagging manual, *Technical Report ISI-TR-545*, Information Science Institute.

Carlson, L., Marcu, D. & Okurowski, M. E. (2001). Building a discourse-tagged corpus in the framework of rhetorical structure theory, *Proceedings of the 2nd SIGDIAL workshop on discourse and dialogue, Eurospeech*, Aalborg, Denmark.

Cassell, J., Nakano, Y. I., Bickmore, T. W., Sidner, C. L. & Rich, C. (2001). Non-verbal cues for discourse structure, *Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics*, Toulouse, France, pp. 106–115.

Christ, O. (1994). A modular and flexible architecture for an integrated corpus query system, *Proceedings of COMPLEX '94: 3rd Conference on Computational Lexicography and Text Research*, Budapest.

Collobert, R. & Bengio, S. (2001). SVMTorch: Support vector machines for large-scale regression problems, *Journal of Machine Learning Research* **1**: 143–160.

Corston-Oliver, S. (1998a). Beyond string matching and cue phrases: Improving efficiency and coverage in discourse analysis, *Proceedings of the AAAI 1998 Spring Symposium Series, Intelligent Text Summarization*, Palo Alto.

Corston-Oliver, S. H. (1998b). *Computing Representations of Discourse Structure*, PhD thesis, University of California, Santa Barbara, CA.

Corston-Oliver, S. H. (1998c). Identifying the linguistic correlates of rhetorical relations, *in* M. Stede, L. Wanner & E. Hovy (eds), *Discourse Relations and Discourse Markers: Proceedings of the workshop*, Association for Computational Linguistics, Montréal.

Cristea, D. & Webber, B. (1997). Expectations in incremental discourse processing, *Proceedings of ACL-EACL97*, Madrid, Spain, pp. 88–95.

Cunningham, H., Wilks, Y. & Gaizauskas, R. (1996). GATE – a general architecture for text engineering, *Proceedings of the 16th Conference on Computational Linguistics*, Copenhagen.

de Smedt, K., Horacek, H. & Zock, M. (1993). Architectures for natural language generation: Problems and perspectives, *in* H. Horacek & M. Zock (eds), *New Concepts in Natural Language Generation. Planning, Realization and Systems*, Pinter, London.

Dietterich, T. G. & Bakiri, G. (1995). Solving multiclass learning problems via error-correcting output codes, *Journal of Artificial Intelligence Research* **2**: 263–286.

Duan, K., Keerthi, S. S. & Poo, A. N. (2001). Evaluation of simple performance measures for tuning svm hyperparameters, *Technical Report CD-01-11*, Dept. of Mechanical Engineering, National University of Singapore.

Frege, G. (1892). Über Sinn und Bedeutung, *Zeitschrift für Philosophie und philosophische Kritik* **100**: 25–50.

Fukumoto, J. & Tsujii, J. (1994). Breaking down rhetorical relations for the purpose of analyzing discourse structures, *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, Kyoto, pp. 1177–1183.

Gazdar, G., Klein, E., Pullum, G. K. & Sag, I. A. (1985). *Generalized Phrase Structure Grammar*, Cambridge, MA.: Harvard University Press.

Grosz, B. J. & Sidner, C. L. (1986). Attention, intentions, and the structure of discourse, *Computational Linguistics* **12**: 175–204.

Grosz, B. J., Joshi, A. K. & Weinstein, S. (1995). Centering: a framework for modelling the local coherence of discourse, *Computational Linguistics* **21**(2): 203–226.

Grote, B. & Stede, M. (1998). Discourse marker choice in sentence planning, *in* E. Hovy (ed.), *Proceedings of the Ninth International Workshop on Natural Language Generation*, Association for Computational Linguistics, New Brunswick, New Jersey, pp. 128–137.

Grote, B., Lenke, N. & Stede, M. (1997). Ma(r)king concessions in english and german., *Discourse Processes* **24**(1): 87–118.

Hearst, M. (1994a). Multi-paragraph segmentation of expository text, *32nd. Annual Meeting of the Association for Computational Linguistics*, New Mexico State University, Las Cruces, New Mexico, pp. 9 – 16.

Hearst, M. A. (1994b). *Context and Structure in Automated Full-text Information Access*, PhD thesis, UC Berkeley.

Hearst, M. A. (1997). TextTiling: Segmenting text into multi-paragraph subtopic passages, *Computational Linguistics* **23**(1): 33–64.

Hirschberg, J. & Litman, D. J. (1993). Empirical studies on the disambiguation of cue phrases, *Computational Linguistics* **19**(3): 501–530.

Hobbs, J. R. (1979). Coherence and coreference, *Cognitive Science* **3**: 67–90.

Hovy, E. H. (1990). Parsimonious and profligate approaches to the question of discourse structure relations, *Proceedings of the Fifth International Workshop on Natural Language Generation*, Dawson, PA, pp. 128–136.

Hovy, E. H. (1993). Automated discourse generation using discourse structure relations, *Artificial Intelligence* **63**(1-2): 341–385.

Ide, N., Bonhomme, P. & Romary, L. (2000). Xces: An xml-based standard for linguistic corpora, *Proceedings of the Second Language Resources and Evaluation Conference (LREC)*, Athens, pp. 825–830.

Joachims, T. (1998). Making large-scale support vector machine learning practical, *in* A. S. B. Schölkopf, C. Burges (ed.), *Advances in Kernel Methods: Support Vector Machines*, MIT Press, Cambridge, MA.

Kallmeyer, L. & Wagner, A. (2000). The TUSNELDA annotation standard: An XML encoding standard for multilingual corpora supporting various aspects of linguistic research, *Proceedings of the conference Digital Resources for the Humanities DRH 2000*, Sheffield, United Kingdom.

Kamp, H. & Reyle, U. (1993). *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, Dordrecht: Kluwer.

Knott, A. (1996). *A Data-Driven Methodology for Motivating a Set of Coherence Relations*, PhD thesis, Department of Artificial Intelligence, University of Edinburgh.

Knott, A. & Dale, R. (1994). Using linguistic phenomena to motivate a set of rhetorical relations, *Discourse Processes* **18**(1): 35–62.

Kurohashi, S. & Nagao, M. (1994). Automatic detection of discourse structure by checking surface information in sentences, *Proceedings of the 15th International Conference on Computational Linguistics (COLING-1994)*, Kyoto, pp. 1123–1127.

Lehmke, S. (2002). TeXPower – Dynamic Online Presentations with LaTeX, `http://texpower.sourceforge.net/` as of 11/2002.

Litman, D. J. & Passonneau, R. J. (1995). Combining multiple knowledge sources for discourse segmentation, *Meeting of the Association for Computational Linguistics*, pp. 108–115.

Longacre, R. (1983). *The Grammar of Discourse*, Topics in Language and Linguistics, New York: Plenum Press.

Mann, W. (1999). An introduction to Rhetorical Structure Theory (RST). `http://www.sil.org/~mannb/rst/rintro99.htm` as of 11/2002.

Mann, W. C. & Thompson, S. A. (1988). Rhetorical Structure Theory: Towards a functional theory of text organization, *Text* **8**(3): 243–281.

Manning, C. D. & Schütze, H. (1999). *Foundations of statistical natural language processing*, Cambridge, MA: MIT Press.

Marcu, D. (1996). Building up rhetorical structure trees, *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI–96)*, Vol. 2, Portland, Oregon, pp. 1069–1074.

Marcu, D. (1997). The rhetorical parsing of natural language texts, *35th Annual Meeting of the Association for Computational Linguistics (ACL/EACL-97)*, Association for Computational Linguistics, Madrid, pp. 96–103.

Marcu, D. (1999). A formal and computational synthesis of Grosz and Sidner's and Mann and Thompson's theories, *Proceedings of the Workshop on Levels of Representation in Discourse (LORID'99) in Discourse, Edinburgh*, Edinburgh.

Marcu, D. (2000). *The theory and practice of discourse parsing and summarization*, Cambridge, MA: MIT Press.

Marcu, D. & Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, Philadelphia.

Marcu, D., Amorrortu, E. & Romera, M. (1999). Experiments in constructing a corpus of discourse trees, *in* M. Walker (ed.), *Towards Standards and Tools for Discourse Tagging: Proceedings of the Workshop*, Association for Computational Linguistics, Somerset, New Jersey, pp. 48–57.

McKelvie, D., Brew, C. & Thompson, H. (1997). Using SGML as a basis for data-intensive NLP, *Proceedings of the 4th Conference on Applied Natural Language Processing (ANLP-97)*, Washington, D.C.

Moore, J. D. & Pollack, M. E. (1992). A problem for RST: The need for multi-level discourse analysis, *Computational Linguistics* **18**(4): 537–544.

Morris, J. & Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text, *Computational Linguistics* **17**(1): 21–48.

O'Donnell, M. (2000). RSTTool 2.4 – a markup tool for Rhetorical Structure Theory, *Proceedings of the 1st International Natural Language Generation Conference*, Mitzpe Ramon, Israel.

Passerini, A., Pontil, M. & Frasconi, P. (2002). From margins to probabilities in multiclass learning problems, *Proceedings of the 15th European Conference on Artificial Intelligence*, Lyon, France.

Polanyi, L. (1988). A formal model of the structure of discourse, *Journal of Pragmatics* **12**: 601–638.

Polanyi, L. (1996). The linguistic structure of discourse, *Technical Report CSLI-96-200*, Stanford University: Center for the Study of Language and Information.

Pollard, C. & Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*, Chicago: University of Chicago Press.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, Machine Learning, Palo Alto, CA: Morgan Kaufmann.

Ratnaparkhi, A. (1998). *Maximum Entropy Models for Natural Language Ambiguity Resolution*, PhD thesis, University of Pennsylvania, Philadelphia.

Rehm, G. (1998). *Vorüberlegungen zur automatischen Zusammenfassung deutschsprachiger Texte mittels einer SGML- und DSSSL- basierten Repräsentation von RST-Relationen*, Master's thesis, Universität Giessen.

Rehm, G. (1999). Automatische Textannotation. Ein SGML- und DSSSL Ansatz zur angewandten Textlinguistik, *in* H. Lobin (ed.), *Text im digitalen Medium. Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering*, Westdeutscher Verlag, Wiesbaden, pp. 155–178.

Reitter, D. (2002a). Rhetorical theory in LaTeX with the 'rst' package, `http://www.reitter-it-media.de/compling/` as of 02/2003.

Reitter, D. (2002b). Statistical part-of-speech guessing for German: Support vector classifiers versus voting, *Proceedings of the 12th Student Conference on Computational Linguistics (TaCoS-2002)*, Potsdam, Germany.

Reitter, D. & Stede, M. (to appear 2003). Step by step: underspecified markup in incremental rhetorical analysis, *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora (LINC-03), in: Proceedings of EACL 2003*, Budapest.

Richmond, K., Smith, A. & Amitay, E. (1997). Detecting subject boundaries within text: A language independent statistical approach, *in* C. Cardie & R. Weischedel (eds), *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Somerset, New Jersey, pp. 47–54.

Rösner, D. & Stede, M. (1992). Customizing RST for the automatic production of technical manuals, *in* R. Dale, E. Hovy, D. Rösner & O. Stock (eds), *Aspects of automated natural*

language generation. Proceedings of the 6th International Workshop on Natural Language Generation, Berlin/Heidelberg: Springer, Trento, Italy.

Sampson, G. (1993). The SUSANNE corpus, *ICAME Journal* **17**: 125–127.

Sanders, T. & van Wijk, C. (1996). PISA: A procedure for analyzing the structure of explanatory texts., *Text* **16**(1): 91–132.

Schauer, H. (2000a). From elementary discourse units to complex ones, *in* L. Dybkjær, K. Hasida & D. Traum (eds), *Proceedings of SigDial2000, in: Proceedings of ACL-2000 the Special Interest Group on Discourse and Dialogue of the Association for Computational Linguistics*, Association for Computational Linguistics, Hong Kong: Morgan Kaufmann Publishers, pp. 46–55.

Schauer, H. (2000b). Referential structure and coherence structure, *Proceedings of TALN 2000 — the 7th Conference on Automatic Natural Language Processing*, Association pour le Traitement Automatique des Langues, Lausanne, Switzerland, pp. 327–336.

Schauer, H. & Hahn, U. (2000). Phrases as carriers of coherence relations, *in* L. R. Gleitman & A. K. Joshi (eds), *Proceedings of the 22nd Annual Meeting of the Cognitive Science Society*, Cognitive Science Society, Lawrence Erlenbaum Associates, Mahwah, New Jersey, Philadelphia, PA, pp. 429–434.

Schilder, F. (2002). Robust discourse parsing via discourse markers, topicality and position, *Natural Language Engineering*.

Schiller, A., Teufel, S. & Thielen, C. (1995). Guidelines fur das Tagging deutscher Textkorpora mit STTS, *Technical report*, Universität Stuttgart and Universität Tübingen.

Schölkopf, B. (1998). Support Vector Machines - a practical consequence of learning theory. in: M.A. Hearst et.al., Trends and Controversies - Support Vector Machines., *IEEE Intelligent Systems* **13**(4): 18–28.

Schölkopf, B. & Smola, A. J. (2002). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, Cambridge, MA: MIT Press.

Schopenhauer, A. (1830). *Without Title. Published under 'Eristische Dialektik' and 'Die Kunst, Recht zu behalten'*.

Scott, D. R. & Sieckenius de Souza, C. (1990). Getting the message across in rst-based text generation, *in* R. Dale, C. Mellish & M. Zock (eds), *Current Research in Natural Language Generation*, Academic Press, London, pp. 47–73.

Siegel, S. & Castellan, N. (1988). *Nonparametric Statistics for the Behavioral Sciences*, New York: McGraw-Hill.

Skut, W., Krenn, B., Brants, T. & Uszkoreit, H. (1997). An annotation scheme for free word order languages, *Proceedings of the Fifth Conference on Applied Natural Language Processing ANLP-97*, Washington, DC.

Stede, M. & Umbach, C. (1998). DiMlex: a lexicon of discourse markers for text generation and understanding, *Proceedings of the 17th International Conference on Computational Linguistics (COLING '98)*, Montréal, pp. 1238–1242.

Vapnik, V. N. (1995). *The nature of statistical learning theory*, New York: Springer.

Abstract in German – deutsche Zusammenfassung

Rhetorische Analyse mit Rhetorical Structure Theory Der Autor fast jeden Textes will informieren, überzeugen oder in Frage stellen. Wesentliches Mittel bei der Realisierung dieser Intentionen ist die Argumentation: Aussagen werden begründet, näher ausgeführt, bewiesen oder widerlegt. Gegensätzliches wird gezeigt, Zugeständnisse gemacht, Ereignisse in ihrer zeitlichen Reihenfolge erzählt. Solche und weitere rhetorische Mittel verknüpfen sich, Aussage für Aussage, zu einem Text. Ein solcher Text ist *kohärent* und unterscheidet sich unter anderem dadurch zu einer losen Sammlung einzelner Sätze. Die Kohärenzstruktur eines Textes kann analysiert werden — man spricht hier von rhetorischer Analyse. Allgemein angenommen werden rhetorische Relationen, die zwischen Textabschnitten – zumindest satzwertige Phrasen – gelten. Auf diese Weise kann eine Baumstruktur von rhetorischen Relationen gebildet werden. Eine der einflussreichsten Theorien, Rhetorical Structure Theory, wurde von Mann & Thompson (1988) entwickelt. Sie sagt zwei Typen von rhetorischen Relationen voraus: *hypotaktische* und *parataktische* Relationen. Hypotaktische Relationen gelten zwischen zwei Textabschnitten, von denen einer als Nucleus, der andere als Satellit ausgezeichnet wird. Der Nucleus trägt die zentrale Bedeutung, auf die weitere Relationen (auf höherer Baumebene) verweisen können. Der Satellit ist in seiner Bedeutung optional. Parataktische Relationen gelten zwischen zwei oder mehr gleichwertigen Textabschnitten.

Rhetorical Structure Theory (RST) wurde in der Sprachverarbeitung zunächst als zugrundeliegendes Modell zum Generieren natürlichsprachlicher Texte eingesetzt, im Besonderen, um Entscheidungen über Diskursmarker zu treffen (Scott & Sieckenius de Souza, 1990; Rösner & Stede, 1992; Hovy, 1993). Die wichtigste Anwendung in der rhetorischen Analyse liegt im Verarbeiten großer Dokumentmengen, insbesondere in der Zusammenfassung von Dokumenten. Bisherige Systeme basieren überwiegend auf dem Erkennen von Diskursmarkern anhand Contraint-orientierter Diskursmarkerlexika (Corston-Oliver, 1998c) und auf einer formalen Analyse der Eigenschaften rhetorischer Struktur (Marcu, 1996). Lernende Systeme treffen Entscheidungen über den inkrementellen Strukturaufbau aufgrund eines Sprachmodells, das mit Hilfe von strukturellen A-priori-Annahmen durch die Beobachtung von Oberflächenmerkmalen generiert wird (Marcu, 2000).

Andere Theorien im Umfeld rhetorischer Analyse sind dynamisch (Grosz & Sidner, 1986; Grosz et al., 1995) und kennen saliente Konzepte an spezifischen Diskurspositionen Kamp & Reyle (1993).

Diese Arbeit widmet sich der Frage, wie rheorische Relationen aufgrund von Oberflächenmerkmalen erkannt werden können. Welche rhetorische Hinweise sind im Text verborgen? Sind die allgemein angenommenen und verwendeten Textmerkmale tatsächlich relevant, d.h. tragen sie zur Erkennungsleistung bei?

Wenn Texte manuell rhetorisch analysiert werden, resultieren häufig ambige Analysen

oder Entscheidungen, die nicht mit Sicherheit getroffen werden können. Carlson et al. (2001) erreichen bei ihrer Korpus-Sammlung eine Übereinstimmungsrate (Kappa-Koeffizient) von 0.8, was zeigt, dass eine Reihe von rhetorischen Relationen unterschiedlich durch die Annotatoren interpretiert werden. Ähnlich verhält es sich bei automatisierten Analysesystemen. Zahlreiche Relationen werden nicht signalisiert. Nahezu alle Signale ("wenn", "als", "aber", "während") sind hochgradig ambig. Um potentielle Signale zu erkennen, sind verschiedene Ebenen linguistischer Analyse nötig. Diese inkrementell annotieren zu können ist ein wesentlicher Faktor für die praktische Realisierbarkeit dieser Systeme.

URML, ein Format zur unterspezifizierten Repräsentation rhetorischer Analysen Mit "Underspecified Rhetorical Markup Language" (URML) stellen wir einen neuen XML-basierten Formalismus vor, der eine elegante inkrementelle Annotation inspesondere rhetorischer Daten erlaubt. Das Paradigma des "Parsewaldes" erlaubt die effiziente Unterspezifizierung von Strukturbäumen. Im Parsewald wird jeder Knoten (d.h. jede Instanz einer Relation) dargestellt, nachdem er analysiert und bewertet wurde. Relationsargumente (Nuclei, Satelliten) werden als Referenzen angegeben. Alternative Relationen können nebeneinander existieren. URML wird im vorgestellten System als Repräsentation sowohl des Korpus, als auch zur intermediären Serialisierung von Analyseergebnissen eingesetzt. Der Potsdam-Korpus (Kapitel 3) wie auch der Korpus in Carlson et al. (2001) werden in URML dargestellt. Vollständige Analysen können automatisch in Druckqualität visualisiert werden (Anhang C).

Korpora Zwei rhetorisch annotierte Korpora dienen dem Training und der Auswertung unseres Analyseansatzes. Der große, englischsprachige Korpus von Zeitungstexten des Wall Street Journal wurde in Carlson et al. (2001) vorgestellt. Ferner erstellen wir einen eigenen, deutschsprachigen rhetorischen Korpus (Kapitel 3), der 173 rhetorisch annotierte Zeitungstexte enthält. Als Grundlage der Annotation verwenden wir eine Untermenge der in der ursprünglichen Rhetorical Structure Theory postulierten Relationen. Die Annotationen wurden jeweils durch einen zweiten Annotator validiert.

Automatische rhetorische Analyse Wir beschreiben die Architektur eines Analysesystems, das auf Support-Vektor-Klassifizierern (Abschnitt 5.4) aufbaut. Diese Klassifizierer treffen Mikro-Entscheidungen über die Anbindung, rhetorische Relation und Nuklearität von Konstituenten oder Teilkonstituenten. Diese Entscheidungen werden aufgrund verschiedener Oberflächenmerkmale getroffen. Wir motivieren die Auswahl der Merkmale zunächst linguistisch und aufgrund bekannter rhetorischer Stilformen des journalistischen Textgenres. Zur Klassifikation müssen Paare von Textabschnitten in ein strukturell konstantes Muster von Merkmalen (Merksmalsvektor) überführt werden (Abschnitte 5.3.2,5.3.3).

Wir trainieren ein Ensemble von binären SVM-Klassifizierern, die eine Bewertung über die Konfidenz in die eigene Entscheidung abgeben können. Die beste Entscheidung – in der Bewertung der Klassifizierer – wird als Ergebnis übernommen. Wir beschreiben außerdem einen Algorithmus, der in zwei Phasen rhetorische Strukturen aufbaut (Abschnitt 5.3.4): Zunächst wird der Suchraum aufgrund lokaler Entscheidungen eingeschränkt (Beam-Suche). In der zweiten Phase können nicht-lokale Eigenschaften berücksichtigt werden: Jeder Knoten in der angenommen rhetorischen Struktur wird in seinem Kontext erneut evaluiert, so dass

ganze Analysen bewertet werden können. Das System wählt die bestbewertete.

Evaluation Wir evaluieren das vorgestellte Analysesystem mit Schwerpunkt auf die wichtigste und schwierigste Klassifikationsleistung: Das Zuweisen einer rhetorischen Relation zu zwei oder mehr bekannten Textabschnitten. Das System erreicht eine Erkennungsleistung von 61.8 Prozent im englischsprachigen Korpus bei einer Basisleistung (baseline) von 33.6 Prozent, die sich aus der Zuweisung der häufigsten Relation ergibt. (Marcu, 2000) erreicht eine Präzision von 57.9 Prozent bei der Erkennung von Relationen bei perfekter Segmentierung. Dieser Wert ist allerdings nur bedingt vergleichbar, unter anderem weil er sich auf einen anderen, nicht veröffentlichten Korpus bezieht. In der Evaluation des Beitrags einzelner Oberflächenmerkmale zum Gesamtergebnis kann für Diskursmarker und für Interpunktionszeichen ein signifikanter, nicht-redundanter Beitrag zur Erkennungsleistung gezeigt werden.

Statement

This is to certify that the thesis comprises only my original work except where indicated, and due acknowledgement has been made in the text to all other material use.

Dublin, February 2003, David Reitter