David Reitter

Simple Signals for Complex Rhetorics: On Rhetorical Analysis with Rich-Feature Support Vector Models

Abstract
Most text displays an internal coherence structure, which can be analyzed as a tree structure of relations that hold between short segments of text. We present a machine-learning governed approach to such an analysis in the framework of *Rhetorical Structure Theory*. Our rhetorical analyzer observes a variety of textual properties, such as cue phrases, part-of-speech information, rhetorical context and lexical chaining. A two-stage parsing algorithm uses local and global optimization to find an analysis. Decisions during parsing are driven by an ensemble of support vector classifiers. This training method allows for a non-linear separation of samples with many relevant features. We define a chain of annotation tools that profits from a new underspecified representation of rhetorical structure. Classifiers are trained on a newly introduced German language corpus, as well as on a large English one. We present evaluation data for the recognition of rhetorical relations.

1 Introduction

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, or 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. One of the central properties of a *text* is that every one part of the text builds on another one. A measure for this property is *coherence*.

There are different types of rhetorical *relations* that connect small chunks of coherent text. How can we automatically recognize them? The motivation for doing this is, that in almost all cases this seems imperative in order to understand a text.

Natural language technology has devoted increasingly more 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 selecting and evaluating information, as in summarization or semantic indexing.

Linguistic accounts for rhetorical structure have the luxury to assume the availability of world knowledge and inference in 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

in: LDV-Forum, GLDV-Journal for Computational Linguistics and Language Technology, 18(1/2):38-52, 2003.

can such an analyzer be built and optimized? Which rhetorical clues are hidden in text? How much can the clues help to improving recognition performance?

A contribution to an answer to 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.

2 Accounts for Rhetorical Structure

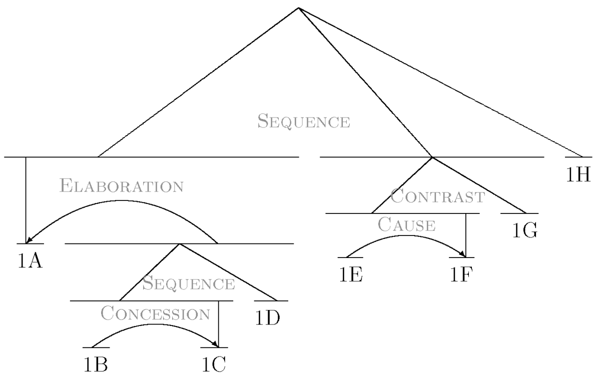
My approach is informed from various fields in rhetorical analysis, parsing and machine learning. In the following, I will describe the foundations of rhetorical text analysis.

2.1 Rhetorical Structure

The work on *Rhetorical Structure Theory* (RST; Mann&Thompson, 1988) builds on tree-like structures that represent rhetorical relationships within the text. Among the children of each tree node, rhetorical relations hold. The theory defines a set of relations that can hold between given spans of text. Common relations include ELABORATION, which is defined as one span that gives additional information regarding the facts presented in the other, and CONTRAST, which is defined as the contents of two or more spans being presented as incompatible (Figure 1).

Since the relations were defined to reflect coherence phenomena, their definitions are not constrained to refer to properties from classical linguistic or computational processing levels. Constraints based on meaning of discourse segments and on world knowledge interact with the presumed writer's or speaker's intentions. 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. The idea of relations to hold between text spans has become a paradigm. Trees of relations are also a common observation in texts. Regarding the relations, the theory never claimed that this set be closed.

The tree-structured analysis seems to build on prevalent rhetorical relations: this suggests there is, even though many analyses can be drawn from a text, only one structure that is faithful to the writer's intentions. It may be questioned whether there is a “primary rhetorical intention” (Grosz&Sidner, 1986). In the approach shown here, we derive a rhetorical interpretation that most likely matches the writer's intention. However, the algorithm can output several interpretations, along with their score.



[Yesterday, the delegates chose their new representative.]^{1A} [Even though Smith received only 24 votes,]^{1B} [he accepted the election with a short speech.]^{1C} [Then the assembly applauded for three minutes.]^{1D} [Due to the upcoming caucus meeting,]^{1E} [the subsequent discussion was very short.]^{1F} [Nonetheless the most pressing questions could be resolved.]^{1G} [The meeting was closed at 7pm.]^{1H}

Figure 1: An example analysis, translated from the German corpus.

2.2 Rhetorical parsing

The *Rhetorical Structure Theory Analyzer* RASTA (Corston-Oliver, 1988) identifies rhetorical relations in text. It looks at various clues from syntactic and partial semantic analyses. Syntactic analysis is used to eliminate ambiguities, where the discourse function of a phrase depends on its syntactic context. RASTA defines *necessary* and *contributing* criteria for each discourse relation.

Marcu (2000) discusses several methods of parsing. His method is driven by automatically derived or manually defined rules. For each discourse marker, these rules state the distribution of nucleus and satellite (left/right of marker), the relations that can be signaled, and the size of textual unit. Relations are, alternatively, also hypothesized according to word co-occurrence. Marcu&Echihabi (2002) investigate the unsupervised learning of classification decisions. Large quantities of sentences are extracted automatically by applying search patterns to corpora.

3 URML, an XML-based standard for rhetorical annotations

3.1 Motivation

With the rise of machine learning and parsing algorithms that detect rhetorical structure and evaluate the results of manual or automatic annotation, it became clear that corpus data should be easy to read by machines and humans. For this reason, the SGML class of annotation formats (a super class of XML) has become increasingly popular in linguistics and, in particular, in rhetorical theory (Rehm, 1998; Lobin, 1999; O'Donnell, 2000).

Systems that perform rhetorical analysis often employ a tool-chain architecture. Step-by-step, they add information, narrowing down search-space. An underspecified representation is needed as interface between the tools. Similarly, manual annotation is not necessarily uniform. Several valid analyses may be drawn for a text and need eventually to be represented in a corpus.

Our solution to these issues is Underspecified Rhetorical Markup Language (Reitter&Stede, to appear). It is an XML based data format that is extensible, readable and writable with standardized methods. It makes few, yet well-defined assumptions about a rhetorical theory that predicts relations to hold between spans of text, but allows for disjunctive and under-specified analyses.

3.2 Rhetorical annotation in URML

Individual documents consist of minimal discourse units, followed by relation nodes 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.

URML uses a referential annotation system instead of in-situ markup, which would not separate segmentation and relations, and which would impose a specific theory on all tools involved. 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 identifiers 7B, 7C, 7D refer to minimal discourse units (cf. Fig. 1).

```
<hypRelation type="concession" id="7L">
```

```
      <satellite id="7B" />
      <nucleus id="7C" />
</hypRelation>
<parRelation type="contrast" id="7M">
      <nucleus id="7L" />
      <nucleus id="7D" />
</parRelation>
```

This syntax 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. If the specific relation between two spans is unknown, a `relation` tag can be used without the `type` attribute.[1]

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.

3.3 Binary relations

Our annotation scheme represents binary relations and assumes that only each nucleus is connected to only one nucleus. This departure from the original theory is not such a major step as it may appear, since the nucleus is known to contribute the essential meaning in comparison to the satellite, and that the nuclearity of a multi-satellite schema can be preserved in a hierarchical binary-branching structure.

Empirical evidence we gained in the collection of the Potsdam Corpus supports this restriction. Annotators could always find a sufficient binary interpretation. The choice between the two binary possibilities for a two-satellite-one-nucleus schema was usually made from referential clues. In the LDC corpus, only 9 out of 17962 relations share a nucleus with others of different type.

Our format defines a symbolic system that may represent ambiguities. The axioms that define a well-formed tree analysis are left open to the client application. At the end of a typical analysis process, we would like to represent such a well-formed structure. It can be identified as such in URML.

[1] A "document-type definition" grammar describing the format of documents, rhetorical and morpho-syntactical annotations can be obtained from http://www.ling.uni-potsdam.de/cl/rst/.

4 Rhetorical Analysis

The automatic detection of rhetorical structure poses a series of challenges. First and foremost, a deep semantic analysis with sufficient coverage seems unfeasible, so, as a design decision, we will concentrate on the integration of various surface cues. These rhetorical signals are found on various linguistic levels (morphosyntax, punctuation, lexical choice, discourse marker). Unfortunately, there is is no one-to-one mapping between signals and rhetorical relations. For instance, *even though* can signal either a CONCESSION or a CONTRAST[2] relation. Also, a dash between two segments is no clear sign for an ELABORATION; instead, a CONCESSION could hold.

As a research hypothesis: language combines various surface cues to signal rhetorical structure. We will determine how far these clues can be used to derive rhetorical relations and, indirectly, structure.

Besides the cues, we need to define the analyzer architecture. We see parsing as a series of classification decisions. As the chart parser proceeds segment by segment, it tries to hypothesize rhetorical relations between the tree nodes that have been found. Here, we need to find one optimal or several good decisions, based on rhetorical signals. *Classification instances* (section 4.1) provide a uniform, local unit, in which signals can be observed. Potential signals are collected from each such unit as *features* (section 4.2). We can then classify each classification instance through a mathematical machine learning model, *support vector machines* (section 4.3). We finally sketch out a parsing algorithm based on classifiers (section 4.4).

4.1 Relations split up in classification instances

The statistical classifiers observe linguistic properties that occur in the text spans. A *feature* is a function that assigns a scalar value to a linguistic pattern, which is a piece of text. Features need to be enumerable; the set of features must be finite. Features are always applied to a structurally equivalent context. We define this context with a simple intermediate data structure, the classification instance. Each paratactic relation node is split up in several classification instances; hypotactic relation nodes are converted to one classification instance.

[2] In a hypotactic CONCESSION, the writer acknowledges an (apparently) incompatible fact in the satellite, but claims that the fact in the nucleus holds nevertheless. For items in a paratactic CONTRAST relation, the writer builds on the incompatibility of the facts.

A transformation function □ maps each relation <*i*, *r*, □, *n*> onto a set of classification instances. *i* is a unique index, *r* is the relation, □ the list of indexes identifying satellites and nuclei and *n* identifies the nucleus in hypotactic relations. A classification instance <*r*, □, *g*> holds a list of text span indexes □ and a single span *g*. The role of □ and *g* differs in hypotactic and paratactic relations. For hypotactic relations (assumed to be binary) □ contains the nucleus and *g* represents the satellite. (Note that the linear precedence of the text spans is encoded as feature.) For all hypotactic *r*:

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

for all paratactic *r*:

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

Nota bene: The classification instances can not only be derived from existing relations, but also from hypothetical parsing edges during analysis.

4.2 Transforming classification instances into feature vectors

In each classification instance, we check the same set of features. Remember that a feature is a function to check a single property of the classification instance. We distinguish *local* features, which hold only information that is available within a classification instance, and *non-local* features, e.g. the depth of embedding or concept introduction. In the following, we describe the different hard-coded templates for features, which are instantiated using the training corpus to form a finite set of features.

Cue words and pronouns: The strongest clues consist of discourse markers. 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 this feature template, we demonstrate its instantiation: We define a class of features $\text{CUE}_{w,b,s}(c)$ that return a positional indicator for a cue word *w* within the first (*b*=0) or second (*b*=1) half of LEFT(*c*) (*s*=0) or RIGHT(*c*) (*s*=1). The *w* are automatically selected from the words in the training corpus according to the part-of-speech categories. The resulting cue features are the $\text{CUE}_{w,b,s}$ for all *w, b* and *s*. **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. In contrast, indefinite noun phrases do not refer in a strict semantic sense. **Punctuation**: The occurrence of a colon, a dash, a question mark and other punctuation marks is noted. **Part-of-Speech categories:** The part-of-speech categories of the words at the borders of segments may help disambiguate the type of relation (sentential). **Lexical similarity**: We use the similarity measure defined in topic chaining research (Hearst 1997) to reflect the semantic similarity of the two text spans considered. **Span lengths:** We hypothesize that some connectives have an argument-length-dependent distribution. Intuitively, 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.

These feature templates are motivated by linguistic knowledge and, for some features, simple corpus-based evaluations that showed a significant class distribution.

4.3 Support Vector Machine learning

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. Among the frameworks available for classification learning, support vector machines (SVM; Vapnik, 1995) have shown to deliver superior results in many applications. The following factors justify the use of SVMs in our case.

- It's a pattern recognition problem, so we deal with multi-class classification.
- We observe features that are inter-related, and exact qualitative and quantitative feature inter-dependence is not known. Naïve Bayes-based algorithms, a standard in statistical language processing, can only approximate the ideal classification solution in this case. SVMs are designed to solve highly non-linear problems.
- We incorporate a large number of features (>5000). For SVMs, the dimensionality of the feature space is not a variable in training time complexity.
- Compared to other machine learning tasks, we train on a small to medium sized set of examples (<10000). State-of-the-art training algorithms handle small training sets within reasonable time.

Support vector machines are based on two ideas. **Large-margin separation:** We position all training samples x_i, in feature space according to their assigned feature vectors. A (linear) classifier is defined as a hyper-plane $w \in R^N, b \in R$ (a plane of dimensionality (N-1), if N features are defined) that separates data points. The hyper-plane should equally and orthogonally divide the shortest connection between the two hulls of the class-related data points (Figure 2). Learning involves finding a solution to the following solvable optimization problem:

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

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

N.B.: only the vector product is used in the above formula and that the number of features is not a factor in the optimization problem. Choosing the □ means identifying a subset of all data vectors that is closest to the hyper-plane. Collobert&Bengio (2001) report an efficient training algorithm for SVMs that is used in our approach.

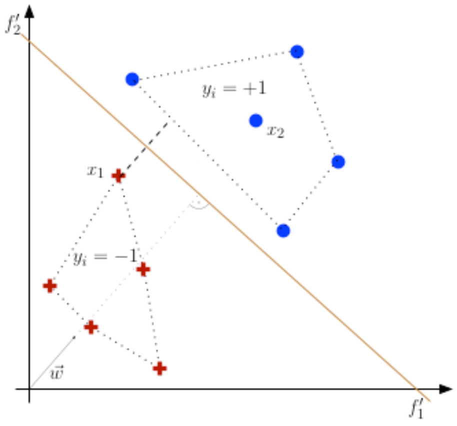


Figure 2: large-margin separation in the separable case

The kernel trick: Most data cannot be separated by means of a linear separation function. A kernel transforms the data from a non-linear into the linear feature space, where it can be separated by a hyper-plane. The kernel function we use is a radial basis function:

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

The function maps the vectors *x* and *y* (from non-linear space) directly to a scalar value that represents the dot product of the vectors in the linear target space. We avoid calculating the dot product, which allows for an efficient implementation.

Solving means to calculate, which side of the hyper-plane a data point is located on as well as its distance to the hyper-plane (confidence). Since we need to solve a multi-class problem, we train a binary SVM classifier for each class and chose the solution for which the according classifier assigns the highest confidence score.

4.4 Parsing algorithm

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). Finally, we can determine *attachment preference* for a text span by scoring alternate hypotheses. Those decision types are all integrated when it comes to rhetorical parsing.

Statistical-based rhetorical parsing borrows some ideas from rule-based chart parsing. We store all analyzed constituents (i.e. relation nodes) in a chart in URML representation, thus avoiding double work. Because we examine the use of non-local features, we need to deal with non-compositionality during parsing. The theoretical solution to this is to generate all possible trees and evaluate each of them as a whole. To speed things up, our parsing algorithm employs first local optimization and beam-search, then global optimization.

5 Evaluation

The evaluation demonstrates the performance of support vector classifiers as used in the parsing process. As training and test data, we use two rhetorical corpora in English and German.

5.1 Corpora

5.1.1 The LDC discourse treebank

The RST annotated collection of 385 English language newspaper articles presented by Carlson et al. (2001) is, as of this writing, the most extensive RST corpus. To prepare for rhetorical analysis, the corpus was converted to URML and part-of-speech tagged. All relations were converted to 16 more general relations according to a hierarchy used during corpus collection. The notion of nuclearity in the LDC corpus is more explicitly defined than in original RST, making additional relations necessary, including ones that occur in hypotactic and paratactic form. These were converted to paratactic ones in order to make evaluation possible.

5.1.2 Collecting the Potsdam 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. We had good reasons for that choice because of the following general design decisions. Texts should...

- expose a clear argumentative structure,
- be short enough to allow annotators to quickly comprehend their overall structure of a document and their argumentative goals, while also long enough to expose the kinds of relations that possibly hold between larger spans of texts,
- be coherent,
- belong to the same genre of text,
- be written in a language that complements efforts in other languages.

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. Automatic segmentation maintains a clear standard for segmentation and allows us to examine pure rhetorical relations and rhetorical structure. The two annotators received training with focus on 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. Annotation was carried out using the graphical annotation tool RSTTool 3.1 (O'Donnell, 2000). We chose 15 hypotactic and 5 paratactic relations along the lines of traditional RST. They can be grouped into 11 more general classes. The annotators completed 173 documents. As a first step towards validation of the data, annotators, they switched datasets and cross-validated all documents.

LDC Corpus (English)

relation	precision	recall
Attribution	64.3	79.4
Background	9.8	3.6
Comparison	60.0	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	45.0	26.5
Temporal	32.1	13.6
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	25.0	10.5
Contrastive	14.3	5.4
Evaluation	31.6	29.3
Evidence	28.8	24.1
Framework	27.3	25.0
Preparation	35.3	50.9
Result	10.0	2.5
Sequential	31.3	28.4
Specification	52.2	75.2
multi-class	accuracy 39.1	

Table 1: Performance for English and German

5.2 Performance analysis

The rhetorical analysis algorithm was subject to extensive evaluation on several levels. In the following, we focus on a singular task that is crucial to rhetorical analysis: The assignment of a relation to hold between known spans of text.

Classifiers were 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 61.8 percent. 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, Table 1). 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 were derived from the Potsdam Corpus. With this amount of data, classification accuracy for the LDC corpus is 56 percent (Figure 3).

We turn to a comparison of the results to a state-of-the-art approach. The relevant measure used Marcu (2000) is labeled recall (57.9) & labeled precision (56.3) for parsing results with assumed perfect segmentation. Our tests looked at structurally correct relations only, thus, labeled precision is the relevant figure to compare to.

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 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 specific relations (Table 1), we found that some harder problems, such as detecting the less frequent and ambiguously signaled CAUSE/RESULT relations, are not solved well by the classifier ensemble. Other rhetorical relations, however, are recognized with relatively high precision/recall, e.g. ELABORATION/SPECIFICATION and ATTRIBUTION. Little surprisingly, these are high-frequent relations in both corpora. 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 shows that punctuation is, after cue phrases, the feature that contributes most to overall classification accuracy. Provided that support vector classifiers do a good job integrating relevant features and that we use a broad variety of heuristics, the findings call for a more informed detection of features. We have not examined the computationally expensive use of a semantic net to achieve a better topic similarity measure. Even better decisions might be drawn from a general model that sees the choice of connectives as result of a represented underlying structure of referents.

The learning curve (Figure 3) shows how the performance of the relation assignment algorithm increases with the number of training samples (classification instances). It converges with the maximum number of training samples used, thus the corpus seems to be large enough for this kind of training task. The 18 binary SVM classifiers can be trained on 150 documents in reasonable time (216 min on 1 Mhz G4-CPU).

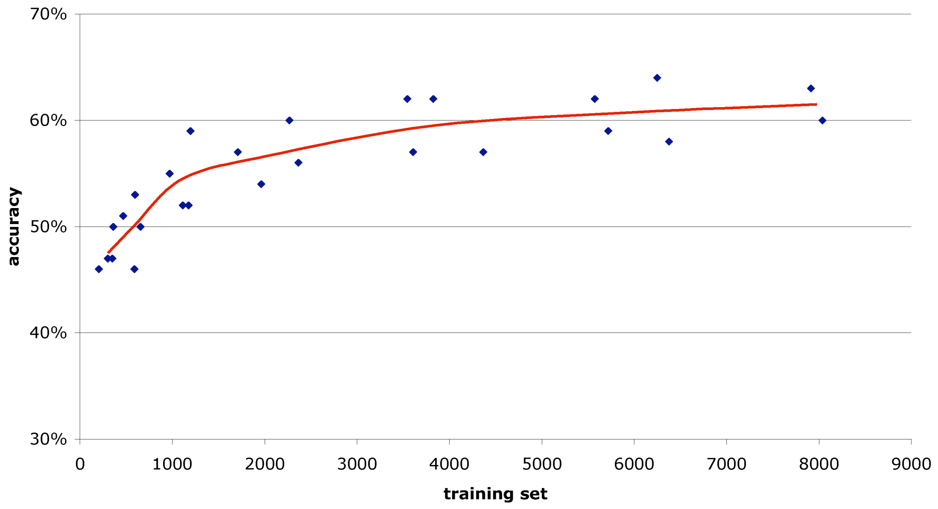


Figure 3: Learning curve of the rhetorical classifiers: number of training samples vs. classification accuracy

6 Conclusion

Most importantly, we have shown a framework for the rhetorical analysis of texts. In this task, support vector classifiers perform exceptionally well. An analysis of the performance of the features shows that many of the clues, commonly assumed to be correlated to rhetorical relations, cannot not significantly contribute to the detection of rhetorical relations. The algorithms shown are able to output several likely analysis in a compact representation.

Underspecified Rhetorical Markup Language is a flexible, XML based format that provides interfacing between tools in rhetorical analysis and corpus annotation. Data can be visualized using a separate package for LaTeX. As a stand-alone data format, it URML applied in the collection of a novel corpus of news commentaries.

Acknowledgements
The author would like to thank Manfred Stede and Deb Roy for their helpful comments on the research related to this paper. MIT Media Lab Europe (EU grant IST-2001-38685) and the University of Potsdam provided support for this work.

Bibliography

Berger, D./Reitter, D./Stede, M. (2002): XML/XSL in the dictionary: The case of discourse markers. *Proceedings of the NLPXML-2002 Workshop*, COLING 2002, Taiwan.

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 2001, Aalborg, Denmark.

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

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

Lobin, H. (1999): Intelligente Dokumente. Linguistische Repräsentation komplexer Inhalte für die hypermediale Wissensvermittlung, in H. Lobin, ed., *Text im digitalen Medium. Linguistische Aspekte von Textdesign, Texttechnologie und Hypertext Engineering*, Wiesbaden: Westdeutscher Verlag. 155-178.

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

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

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

O'Donnell, M. (2000): RSTTool 2.4 - A markup tool for Rhetorical Structure Theory. *Proceedings of the International Natural Language Generation Conference (INLG'2000),* Mitzpe Ramon, Israel. 253 -- 256.

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

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

Schilder, F. (2002): Robust discourse parsing via discourse markers, topicality and position. *Natural Language Engineering* 2&3(8).

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