Learning Features that Predict Cue Usage

Citation for published version:

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Publisher's PDF, also known as Version of record

Published In:

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Learning Features that Predict Cue Usage

Barbara Di Eugenio∗  Johanna D. Moore†  Massimo Paolucci‡
University of Pittsburgh
Pittsburgh, PA 15260, USA
{dieugeni,jmoore,paolucci}@cs.pitt.edu

Abstract

Our goal is to identify the features that predict the occurrence and placement of discourse cues in tutorial explanations in order to aid in the automatic generation of explanations. Previous attempts to devise rules for text generation were based on intuition or small numbers of constructed examples. We apply a machine learning program, C4.5, to induce decision trees for cue occurrence and placement from a corpus of data coded for a variety of features previously thought to affect cue usage. Our experiments enable us to identify the features with most predictive power, and show that machine learning can be used to induce decision trees useful for text generation.

1 Introduction

Discourse cues are words or phrases, such as because, first, and although, that mark structural and semantic relationships between discourse entities. They play a crucial role in many discourse processing tasks, including plan recognition (Litman and Allen 1987), text comprehension (Cohen, 1984; Hobbs, 1987; Mann and Thompson, 1986; Reichman-Adar, 1984), and anaphora resolution (Grosz and Sidner, 1986). Moreover, research in reading comprehension indicates that felicitous use of cues improves comprehension and recall (Goldman, 1988), but that their indiscriminate use may have detrimental effects on recall (Millis, Graesser, and Haberlandt, 1993).

Our goal is to identify general strategies for cue usage that can be implemented for automatic text generation. From the generation perspective, cue usage consists of three distinct, but interrelated problems: (1) occurrence: whether or not to include a cue in the generated text, (2) placement: where the cue should be placed in the text, and (3) selection: what lexical item(s) should be used.

Prior work in text generation has focused on cue selection (McKeown and Elhadad, 1991; Elhadad and McKeown, 1990), or on the relation between cue occurrence and placement and specific rhetorical structures (Rosner and Steed, 1992; Scott and de Souza, 1990; Vander Linden and Martin, 1995). Other hypotheses about cue usage derive from work on discourse coherence and structure. Previous research (Hobbs, 1985; Grosz and Sidner, 1986; Schiffrin, 1987; Mann and Thompson, 1988; Elhadad and McKeown, 1990), which has been largely descriptive, suggests factors such as structural features of the discourse (e.g., level of embedding and segment complexity), intentional and informational relations in that structure, ordering of relata, and syntactic form of discourse constituents.

Moser and Moore (1993; 1997) coded a corpus of naturally occurring tutorial explanations for the range of features identified in prior work. Because they were also interested in the contrast between occurrence and non-occurrence of cues, they exhaustively coded for all of the factors thought to contribute to cue usage in all of the text. From their study, Moser and Moore identified several interesting correlations between particular features and specific aspects of cue usage, and were able to test specific hypotheses from the literature that were based on constructed examples.

In this paper, we focus on cue occurrence and placement, and present an empirical study of the hypotheses provided by previous research, which have never been systematically evaluated with naturally occurring data. We use a machine learning program, C4.5 (Quinlan, 1993), on the tagged corpus of Moser∗ Learning Research & Development Center
Computer Science Department, and Learning Research & Development Center
Intelligent Systems Program
and Moser and Moore to induce decision trees. The number of coded features and their interactions makes the manual construction of rules that predict cue occurrence and placement an intractable task.

Our results largely confirm the suggestions from the literature, and clarify them by highlighting the most influential features for a particular task. Discourse structure, in terms of both segment structure and levels of embedding, affects cue occurrence the most; intentional relations also play an important role. For cue placement, the most important factors are syntactic structure and segment complexity.

The paper is organized as follows. In Section 2 we discuss previous research in more detail. Section 3 provides an overview of Moser and Moore’s coding scheme. In Section 4 we present our learning experiments, and in Section 5 we discuss our results and conclude.

2 Related Work

McKeown and Elhadad (1991, 1990) studied several connectives (e.g., but, since, because), and include many insightful hypotheses about cue selection; their observation that the distinction between but and although depends on the point of the move is related to the notion of core discussed below. However, they do not address the problem of cue occurrence.

Other researchers (Rössner and Stede, 1992; Scott and de Souza, 1990) are concerned with generating text from “RST trees”, hierarchical structures where leaf nodes contain content and internal nodes indicate the rhetorical relations, as defined in Rhetorical Structure Theory (RST) (Mann and Thompson, 1988), that exist between subtrees. They proposed heuristics for including and choosing cues based on the rhetorical relation between spans of text, the order of the relata, and the complexity of the related text spans. However, (Scott and de Souza, 1990) was based on a small number of constructed examples, and (Rössner and Stede, 1992) focused on a small number of RST relations.

(Siegel and McKeown, 1994) have applied machine learning to disambiguate between the discourse and sentential usages of cues; however, they do not consider the issues of occurrence and placement, and approach the problem from the point of view of interpretation. We closely follow the approach in (Litman, 1996) in two ways. First, we use C4.5. Second, we experiment first with each feature individually, and then with “interesting” subsets of features.

3 Relational Discourse Analysis

This section briefly describes Relational Discourse Analysis (RDA) (Moser, Moore, and Glendening, 1990), the coding scheme used to tag the data for our machine learning experiments.

RDA is a scheme devised for analyzing tutorial explanations in the domain of electronics troubleshooting. It synthesizes ideas from (Grosz and Sidner, 1986) and from RST (Mann and Thompson, 1988). Coders use RDA to exhaustively analyze each explanation in the corpus, i.e., every word in each explanation belongs to exactly one element in the analysis. An explanation may consist of multiple segments. Each segment originates with an intention of the speaker. Segments are internally structured and consist of a core, i.e., that element that most directly expresses the segment purpose, and any number of contributors, i.e., the remaining constituents.

For each contributor, one analyzes its relation to the core from an intentional perspective, i.e., how it is intended to support the core, and from an informational perspective, i.e., how its content relates to that of the core. The set of intentional relations in RDA is a modification of the presentational relations of RST, while informational relations are similar to the subject matter relations in RST. Each segment constituent, both core and contributors, may itself be a segment with a core:contributor structure. In some cases the core is not explicit. This is often the case with the whole tutor’s explanation, since its purpose is to answer the student’s explicit question.

As an example of the application of RDA, consider the partial tutor explanation in (1). The purpose of this segment is to inform the student that she made the strategy error of testing inside part3 too soon. The constituent that makes the purpose obvious, in this case (1), is the core of the segment. The other constituents help to serve the segment purpose by contributing to it. (1)-C) is an example of subsegment with its own core:contributor structure; its purpose is to give a reason for testing part2 first.

The RDA analysis of (1) is shown schematically in Figure 1. The core is depicted as the mother of all the relations it participates in. Each relation node is labeled with both its intentional and informational relation, with the order of relata in the label indicating the linear order in the discourse. Each relation node has up to two daughters: the cue, if any, and

\footnote{For more detail about the RDA coding scheme see (Moser and Moore, 1996, Moser and Moore, 1997).}

\footnote{To make the example more intelligible, we replaced references to parts of the circuit with the labels part1, part2 and part3.}
Although you know that part1 is good,
B. you should eliminate part2 before troubleshooting inside part3.

This is because C. part2 is moved frequently
and thus 2. is more susceptible to damage than part3.

Also, and D. it is more work to open up part3 for testing
and E. the process of opening drawers and extending cards in part3 may induce problems which did not already exist.

Figure 1: The RDA analysis of (1)

the contributor, in the order they appear in the discourse.

Coders analyze each explanation in the corpus and enter their analyses into a database. The corpus consists of 854 clauses comprising 668 segments, for a total of 780 relations. Table 1 summarizes the distribution of different relations, and the number of cued relations in each category. Joints are segments comprising more than one core, but no contributor; clusters are multiunit structures with no recognizable core:contributor relation. (1-B) is a cluster composed of two units (the two clauses), related only at the informational level by a temporal relation. Both clauses describe actions, with the first action description embedded in a matrix ("You should"). Cues are much more likely to occur in clusters, where only informational relations occur, than in core:contributor structures, where intentional and informational relations co-occur ($\chi^2 = 33.367$, $p < .001$, $df = 1$). In the following, we will not discuss joints and clusters any further.

An important result pointed out by (Moser and Moore, 1995) is that cue placement depends on core position. When the core is first and a cue is associated with the relation, the cue never occurs with the core. In contrast, when the core is second, if a cue occurs, it can occur either on the core or on the contributor.

4 Learning from the corpus

4.1 The algorithm

We chose the C4.5 learning algorithm (Quinlan, 1993) because it is well suited to a domain such as ours with discrete valued attributes. Moreover, C4.5 produces decision trees and rule sets, both often used in text generation to implement mappings from function features to forms. Finally, C4.5 is both readily available, and is a benchmark learning algorithm that has been extensively used in NLP applications, e.g. (Litman, 1996; Mooney, 1996; Vander Linden and Di Eugenio, 1996).

As our dataset is small, the results we report are based on cross-validation, which (Weiss and Kulikowski, 1991) recommends as the best method to evaluate decision trees on datasets whose cardinality is in the hundreds. Data for learning should be divided into training and test sets; however, for small datasets this has the disadvantage that a sizable portion of the data is not available for learning. Cross-

---

3We will discuss only decision trees here.
<table>
<thead>
<tr>
<th>Type of relation</th>
<th>Total</th>
<th># of cued relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core:Contributor</td>
<td>406</td>
<td>181</td>
</tr>
<tr>
<td>Joints</td>
<td>64</td>
<td>19</td>
</tr>
<tr>
<td>Clusters</td>
<td>310</td>
<td>276</td>
</tr>
<tr>
<td>Total</td>
<td>780</td>
<td>476</td>
</tr>
</tbody>
</table>

Table 1: Distributions of relations and cue occurrences

validation obviates this problem by running the algorithm N times (N=10 is a typical value): in each run, \((N-1)/N\)th of the data, randomly chosen, is used as the training set, and the remaining \(1/N\)th used as the test set. The error rate of a tree obtained by using the whole dataset for training is then assumed to be the average error rate on the test set over the N runs. Further, as C4.5 prunes the initial tree it obtains to avoid overfitting, it computes both actual and estimated error rates for the pruned tree; see (Quinlan, 1993) for details. Thus, below we will report the average estimated error rate on the test set, as computed by 10-fold cross-validation experiments.

### 4.2 The features

Each data point in our dataset corresponds to a core:contributor relation, and is characterized by the following features, summarized in Table 2.

- **Segment Structure.** Three features capture the global structure of the segment in which the current core:contributor relation appears.
  - *(Con)Trib(utor)-pos(ition)* captures the position of a particular contributor within the larger segment in which it occurs, and encodes the structure of the segment in terms of how many contributors precede and follow the core. For example, contributor (1-D) in Figure 1 is labeled as B1A3-2after, as it is the second contributor following the core in a segment with 1 contributor before and 3 after the core.
  - Inten(tional)-structure indicates which contributors in the segment bear the same intentional relations to the core.
  - Infor(mational)-structure. Similar to intentional structure, but applied to informational relations.

- **Core:contributor relation.** These features more specifically characterize the current core:contributor relation.
  - Inten(tional)-rel(ation). One of concede, convince, enable.
  - Infor(mational)-rel(ation). About 30 informational relations have been coded for. However, as preliminary experiments showed that using them individually results in overfitting the data, we classify them according to the four classes proposed in Moser, Moore, and Glendening (1996): causality, similarity, elaboration, temporal. Temporal relations only appear in clusters, thus not in the data we discuss in this paper.
  - Sym(tactic)-rel(ation). Captures whether the core and contributor are independent units (segments or sentences); whether they are coordinated clauses; or which of the two is subordinate to the other.
  - Adjacency. Whether core and contributor are adjacent in linear order.

- **Embedding.** These features capture segment embedding. Core-type and Trib-type qualitatively, and Above/Below quantitatively.
  - Core-type/(Con)Trib(utor)-type. Whether the core/the contributor is a segment, or a minimal unit (further subdivided into action, state, matrix).
  - Above/Below encode the number of relations hierarchically above and below the current relation.

### 4.3 The experiments

Initially, we performed learning on all 406 instances of core:contributor relations. We quickly determined that this approach would not lead to useful decision trees. First, the trees we obtained were extremely complex (at least 50 nodes). Second, some of the subtrees corresponded to clearly identifiable subclasses of the data, such as relations with an implicit core, which suggested that we should apply learning to these independently identifiable subclasses. Thus, we subdivided the data into three subsets:

- Core1: core:contributor relations with the core in first position.
<table>
<thead>
<tr>
<th>feature type</th>
<th>feature</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment structure</td>
<td>Trib-pos</td>
<td>relative position of contrib in segment + number of contribs before and after core</td>
</tr>
<tr>
<td></td>
<td>Inten-structure</td>
<td>intentional structure of segment</td>
</tr>
<tr>
<td></td>
<td>Infor-structure</td>
<td>informational structure of segment</td>
</tr>
<tr>
<td>Core:contributor relation</td>
<td>Inten-rel</td>
<td>enable, convince, concede</td>
</tr>
<tr>
<td></td>
<td>Info-rel</td>
<td>4 classes of about 30 distinct relations</td>
</tr>
<tr>
<td></td>
<td>Syn-rel</td>
<td>independent sentences / segments, coordinated clauses, subordinated clauses are core and contributor adjacent?</td>
</tr>
<tr>
<td>Adjacency</td>
<td></td>
<td>segment, minimal unit</td>
</tr>
<tr>
<td></td>
<td>Core-type</td>
<td>segment, minimal unit</td>
</tr>
<tr>
<td>Embedding</td>
<td>Trib-type</td>
<td>number of relations hierarchically</td>
</tr>
<tr>
<td></td>
<td>Above / Below</td>
<td>above / below current relation</td>
</tr>
</tbody>
</table>

Table 2: Features

- **Core2**: core:contributor relations with the core in second position

- **Impl(icit)-core**: core:contributor relations with an implicit core

While this has the disadvantage of smaller training sets, the trees we obtain are more manageable and more meaningful. Table 3 summarizes the cardinality of these sets, and the frequencies of cue occurrence.

We ran four sets of experiments. In three of them we predict cue occurrence and in one cue placement.

4.3.1 Cue Occurrence

Table 4 summarizes our main results concerning cue occurrence, and includes the error rates associated with different feature sets. We adopt Litman’s approach (1996) to determine whether two error rates \( E_1 \) and \( E_2 \) are significantly different. We compute 95% confidence intervals for the two error rates using a \( t \)-test. \( E_1 \) is significantly better than \( E_2 \) if the upper bound of the 95% confidence interval for \( E_1 \) is lower than the lower bound of the 95% confidence interval for \( E_2 \).

For each set of experiments, we report the following:

1. A baseline measure obtained by choosing the majority class. E.g., for **Core1** 58.9% of the relations are not cued; thus, by deciding to never include a cue, one would be wrong 41.1% of the times.

2. The best individual features whose predictive power is better than the baseline: as Table 4 makes apparent, individual features do not have much predictive power. For neither **Core1** nor **Impl-core** does any individual feature perform better than the baseline, and for **Core2** only one feature is sufficiently predictive.

3. (One of) the best induced tree(s). For each tree, we list the number of nodes, and up to six of the features that appear highest in the tree, with their levels of embedding. Figure 2 shows the tree for **Core2** (space constraints prevent us from including figures for each tree). In the figure, the numbers in parentheses indicate the number of cases correctly covered by the leaf, and the number of expected errors at that leaf.

Learning turns out to be most useful for **Core1**, where the error reduction (as percentage) from baseline to the upper bound of the best result is 32%; error reduction is 19% for **Core2** and only 3% for **Impl-core**.

The best tree was obtained partly by informed choice, partly by trial and error. Automatically trying out all the \( 2^{11} = 2048 \) subsets of features would be possible, but it would require manual examination of about 2,000 sets of results, a daunting task. Thus, for each dataset we considered only the following subsets of features.

1. All features. This always results in C4.5 selecting a few features (from 3 to 7) for the final tree.

2. Subsets built out of the 2 to 4 attributes appearing highest in the tree obtained by running

4. All our experiments are run with grouping turned on, so that C4.5 groups values together rather than creating a branch per value. The latter choice always results in trees overfitted to the data in our domain. Using classes of informational relations, rather than individual informational relations, constitutes a sort of a priori grouping.

5. The trees that C4.5 generates are right-branching, so this description is fairly adequate.
### Dataset

<table>
<thead>
<tr>
<th></th>
<th># of relations</th>
<th># of cued relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core1</td>
<td>127</td>
<td>52</td>
</tr>
<tr>
<td>Core2</td>
<td>155</td>
<td>100</td>
</tr>
<tr>
<td>Impl-core</td>
<td>124</td>
<td>29</td>
</tr>
<tr>
<td>Total</td>
<td>406</td>
<td>181</td>
</tr>
</tbody>
</table>

Table 3: Distributions of relations and cue occurrences

### Table 4: Summary of learning results

<table>
<thead>
<tr>
<th></th>
<th>Core1</th>
<th>Core2</th>
<th>Impl-core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>41.1</td>
<td>35.4</td>
<td>23.5</td>
</tr>
<tr>
<td>Best features</td>
<td>∅</td>
<td>∅</td>
<td>∅</td>
</tr>
<tr>
<td>Best tree</td>
<td>25.6±1.24 (15)</td>
<td>27.4±1.28 (18)</td>
<td>22.1±0.57 (10)</td>
</tr>
<tr>
<td>0. Trib-pos</td>
<td></td>
<td></td>
<td>0. Core-type</td>
</tr>
<tr>
<td>1. Trib-type</td>
<td></td>
<td></td>
<td>1. Infor-struct</td>
</tr>
<tr>
<td>2. Syn-rel</td>
<td></td>
<td></td>
<td>2. Inten-rel</td>
</tr>
<tr>
<td>3. Core-type</td>
<td></td>
<td></td>
<td>4. Core-type</td>
</tr>
<tr>
<td>4. Above</td>
<td></td>
<td></td>
<td>5. Below</td>
</tr>
</tbody>
</table>

C4.5 on all features.

3. In Table 3, three features — Trib-pos, Inten-struct, Infor-struct — concern segment structure, eight do not. We constructed three subsets by always including the eight features that do not concern segment structure, and adding one of those that does. The trees obtained by including Trib-pos, Inten-struct, Infor-struct at the same time are in general more complex, and not significantly better than other trees obtained by including only one of these three features. We attribute this to the fact that these features encode partly overlapping information.

Finally, the best tree was obtained as follows. We build the set of trees that are statistically equivalent to the tree with the best error rate (i.e., with the lowest error rate upper bound). Among these trees, we choose the one that we deem the most perspicuous in terms of features and of complexity. Namely, we pick the simplest tree with Trib-Pos as the root if one exists, otherwise the simplest tree. Trees that have Trib-Pos as the root are the most useful for text generation, because, given a complex segment, Trib-Pos is the only attribute that unambiguously identifies a specific contributor.

Our results make apparent that the structure of segments plays a fundamental role in determining cue occurrence. One of the three features concerning segment structure (Trib-Pos, Inten-Structure, Infor-Structure) appears as the root or just below the root in all trees in Table 3. More importantly, this same configuration occurs in all trees equivalent to the best tree (even if the specific feature encoding segment structure may change). The level of embedding in a segment, as encoded by Core-type, Trib-type, Above and Below also figures prominently.

Inten-rel appears in all trees, confirming the intuition that the speaker’s purpose affects cue occurrence. More specifically, in Figure 2, Inten-rel distinguishes two different speaker purposes, convince and enable. The same split occurs in some of the best trees induced on Core1, with the same outcome: i.e., convince directly correlates with the occurrence of a cue, whereas for enable other features must be taken into account. Informational relations do not appear as often as intentional relations; their discriminatory power seems more relevant for clusters. Preliminary experiments show that cue occurrence in clusters depends only on informational and syntactic relations. Finally, Adjacency does not seem to play any substantial role.

---

6We can’t draw any conclusions concerning concede, as there are only 24 occurrences of concede out of 406 core:contributor relations.
4.3.2 Cue Placement

While cue occurrence and placement are interrelated problems, we performed learning on them separately. First, the issue of placement arises only in the case of Core2; for Core1, cues only occur on the contributor. Second, we attempted experiments on Core2 that discriminated between occurrence and placement at the same time, and the derived trees were complex and not perspicuous. Thus, we ran an experiment on the 100 cued relations from Core2 to investigate which factors affect placing the cue on the contributor in first position or on the core in second; see Table 5.

We ran the same trials discussed above on this dataset. In this case, the best tree — see Figure 3 — results from combining the two best individual features, and reduces the error rate by 50%. The most discriminant feature turns out to be the syntactic relation between the contributor and the core. However, segment structure still plays an important role, via Trib-pos.

While the importance of Syn-rel for placement seems clear, its role concerning occurrence requires further exploration. It is interesting to note that the tree induced on Core1 — the only case in which Syn-rel is relevant for occurrence — includes the same distinction as in Figure 3, namely, if the contributor depends on the core, the contributor must be marked, otherwise other features have to be taken into account. Scott and de Souza (1990) point out that “there is a strong correlation between the syntactic specification of a complex sentence and its perceived rhetorical structure.” It seems that certain

Figure 2: Decision tree for Core2 — occurrence

| Table 5: Cue placement on Core2 |
|-----------------|-----------------|
| Baseline        | 43%             |
| Best features   | Syn-rel: 24.1±0.69 |
|                 | Trib-pos: 40±0.88 |
| Best tree       | 20.6±0.97 (5)   |
|                 | 0. Syn-rel       |
|                 | 1. Trib-pos      |
syntactic structures function as a cue.

5 Discussion and Conclusions

We have presented the results of machine learning experiments concerning cue occurrence and placement. As (Litman, 1996) observes, this sort of empirical work supports the utility of machine learning techniques applied to coded corpora. As our study shows, individual features have no predictive power for cue occurrence. Moreover, it is hard to see how the best combination of individual features could be found by manual inspection.

Our results also provide guidance for those building text generation systems. This study clearly indicates that segment structure, most notably the ordering of core and contributor, is crucial for determining cue occurrence. Recall that it was only by considering Core1 and Core2 relations in distinct datasets that we were able to obtain perspicuous decision trees that significantly reduce the error rate.

This indicates that the representations produced by discourse planners should distinguish those elements that constitute the core of each discourse segment, in addition to representing the hierarchical structure of segments. Note that the notion of core is related to the notions of nucleus in RST, intended effect in (Young and Moore, 1994), and of point of a move in (Elhadad and McKeown, 1990), and that text generators representing these notions exist.

Moreover, in order to use the decision trees derived here, decisions about whether or not to make the core explicit and how to order the core and contributor(s) must be made before deciding cue occurrence, e.g., by exploiting other factors such as focus (McKeown, 1985) and a discourse history.

Once decisions about core:contributor ordering and cue occurrence have been made, a generator must still determine where to place cues and select appropriate lexical items. A major focus of our future research is to explore the relationship between the selection and placement decisions. Elsewhere, we have found that particular lexical items tend to have a preferred location, defined in terms of functional (i.e., core or contributor) and linear (i.e., first or second relatum) criteria (Moser and Moore, 1997). Thus, if a generator uses decision trees such as the one shown in Figure 3 to determine where a cue should be placed, it can then select an appropriate cue from those that can mark the given intentional/informational relations, and are usually placed in that functional-linear location.

To evaluate this strategy, we must do further work to understand whether there are important distinctions among cues (e.g., so, because) apart from their different preferred locations. The work of Elhadad (1990) and Knott (1996) will help in answering this question.

Future work comprises further probing into machine learning techniques, in particular investigating whether other learning algorithms are more appropriate for our problem (Mooney, 1996), especially algorithms that take into account some a priori knowledge about features and their dependencies.

Figure 3: Decision tree for Core2 — placement
Acknowledgements

This research is supported by the Office of Naval Research, Cognitive and Neural Sciences Division (Grants N00014-91-J-1694 and N00014-93-I-0812). Thanks to Megan Moser for her prior work on this project and for comments on this paper; to Erin Glendening and Liina Pyhkkänen for their coding efforts; to Haiqin Wang for running many experiments; to Giuseppe Carenini and Steffi Brüninghaus for discussions about machine learning.

References


