Computer-Supported Collaborative Learning
DOI 10.1007/s11412-007-9034-0

Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning

Carolyn Rosé · Yi-Chia Wang · Yue Cui · Jaime Arguello · Karsten Stegmann · Armin Weinberger · Frank Fischer

Received: 2 June 2007 / Accepted: 26 November 2007

Abstract In this article we describe the emerging area of text classification research focused on the problem of collaborative learning process analysis both from a broad perspective and more specifically in terms of a publicly available tool set called TagHelper tools. Analyzing the variety of pedagogically valuable facets of learners' interactions is a time consuming and effortful process. Improving automated analyses of such highly valued processes of collaborative learning by adapting and applying recent text classification technologies would make it a less arduous task to obtain insights from corpus data. This endeavor also holds the potential for enabling substantially improved on-line instruction both by providing teachers and facilitators with reports about the groups they are moderating and by triggering context sensitive collaborative learning support on an as-needed basis. In this article, we report on an interdisciplinary research project, which has been investigating the effectiveness of applying text classification technology to a large CSCL corpus that has been analyzed by human coders using a theory-based multi-dimensional coding scheme. We report promising results and include an in-depth discussion of important issues such as reliability, validity, and efficiency that should be considered when deciding on the appropriateness of adopting a new technology such as TagHelper tools. One major technical contribution of this work is a demonstration that an important piece of the work towards making text classification technology effective for this purpose is designing and building linguistic pattern detectors, otherwise known as features, that can be extracted reliably from texts and that have high predictive power for the categories of discourse actions that the CSCL community is interested in.

Keywords Collaborative process analysis · Machine learning · Analysis tools

C. Rosé (✉) · Y.-C. Wang · Y. Cui · J. Arguello
Language Technologies Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA
e-mail: cprose@cs.cmu.edu

K. Stegmann · A. Weinberger · F. Fischer
University of Munich, Munich, Germany

From the very beginning, the identity of Computer Supported Collaborative Learning (CSCL) research has been defined to a considerable extent by sophisticated analyses of collaboration processes (e.g., Dillenbourg et al. 1995). The social interaction of computer-supported collaborative learners in discourse has been regarded as a "gold mine of information" (Henri 1992, p. 118) on how learners acquire knowledge and skills together (e.g., Wegerif 2006; Stahl 2006). The interest of educational researchers in this topic has evolved from early work ranging from assessing participation by counting the number of student contributions to an in-depth understanding of different qualities of interaction (De Wever et al. 2006; van der Pol et al. 2006; Webb 1989). Knowledge building and learning of collaborative learners has been linked to the process by which collaborative learners work on the learning task together (Fischer et al. 2002), how they construct arguments and argumentation sequences (Leitão 2000; Voss and Van Dyke 2001), and how they build on the contributions of their learning partners (Teasley 1997), which may involve receiving help or providing help to one another (Gweon et al. 2007). Analyzing these different facets of learners' interaction is a time consuming and effortful process. Improving automated analyses of such highly valued processes of collaborative learning by using recent text classification technologies would make it a less arduous task to obtain insights from corpus data. This endeavor also holds the potential for enabling substantially improved on-line instruction both by providing teachers and facilitators with reports about the groups they are moderating (Rosé et al. 2007) and by triggering context sensitive collaborative learning support on an as-needed basis (Kumar et al. 2007; Wang et al. 2007b).

We have been regarding recent advances in text classification technology in the realm of computational linguistics as an extremely promising means to that end. In this article, we report on an interdisciplinary research project, which has been investigating the effectiveness of applying text classification technology to a large CSCL corpus that had been analyzed by human coders using a theory-based multi-dimensional coding scheme (Weinberger and Fischer 2006).

In what follows, we will first describe the motivation for using text classification technology to automate some aspects of the CSCL corpus analysis. We then address the methodological issues of reliability and validity, and the practical issue of coding speed. We will then discuss some technical challenges that we have addressed in this work as well as providing an evaluation that demonstrates the achievements as well as the remaining limitations of our current technical approach. And finally, we describe practical consequences of our interdisciplinary research approach in terms of the TagHelper application, which is a corpus analysis environment built on top of the Weka machine learning toolkit (Witten and Frank 2005). We explore the functionality provided by the TagHelper tools environment to researchers analyzing collaborative learning process data. Specifically, in this article we explore the design and implementation of context-oriented features, otherwise known as linguistic pattern detectors, that reflect the thread structure of the discourse. We conclude with a discussion of some directions for ongoing work.

Motivation for automatic corpus analysis

Popular tools for facilitating the analysis of corpus data provide functionality for assisting analysts in their task of finding meaningful patterns in corpus data once it has been coded. Tools such as HyperResearch, MacShapa, or Nvivo, are commonly used by behavioral researchers to analyze their data either using word counting or key phrase matching

approaches, or to apply categorical coding schemes by hand and then visualize patterns of hand assigned codes. Tools developed specifically for analysis of collaborative learning interactions rarely include support for annotating raw text either automatically or semi-automatically (Luckin 2002; Hmelo-Silver and Chernobilsky 2004).

In early attempts to support corpus analysis efforts with automatic text analysis technology, health communication researchers have augmented hand-coding with automated content analysis techniques, primarily using dictionary-based methods, the most popular of which is Pennebaker's Linguistic Inquiry and Word Count (LIWC) (Pennebaker 2003). In this approach, collections of words are organized into scales that are supposed to indicate specific mental states such as negative emotion or confidence. Shallow text processing tools such as Pennebaker's LIWC scales are the state-of-the-art in text analysis in support of behavioral research, particularly for social psychology research involving language interactions. Because of their popularity, simplicity, and ease of use, and because the history of automatic corpus analysis began with approaches such as these word counting approaches, we provide a discussion here on the trade-offs between word counting approaches and categorical analysis approaches.

Linguistic inquiry and word count (LIWC)

LIWC indicators that are designed to measure latent characteristics of authors such as emotional or psychological state based on vocabulary usage are reported to have been successfully calibrated with a wide range of behaviors over multiple types of studies (e.g., Pennebaker 2003). Nevertheless, they have limitations that must be taken into account methodologically. LIWC indicators have typically been used in studies where the external variables of interest are health outcomes or health related behavior. In studies where consistent stories based on calibrations of LIWC indicators with external variables are reported, the corpora used were created under very controlled circumstances, always only within the experimental condition of a study in which the genre and topic of the writing were determined by the experimental manipulation. When these tight constraints are removed, the story becomes much less clear. For example, Pennebaker and Francis (1996) present results from a study with two different conditions. The experimental variation lay in the change of the topic participants wrote about. In this study, the LIWC indicators made opposite predictions about behavioral outcomes and emotional states in the experimental condition in comparison to the control condition. Discrepancies like this occur because there are many linguistic factors besides the emotional state of the author or speaker that affect the frequencies of word usage. For example, many words have multiple meanings and only convey negative emotion in some contexts and not in others. For example, the words "bake" and "roast" used while talking about the weather convey a feeling of discomfort, whereas in the context of a discussion about cooking, they do not. Base frequencies of terms also vary between topics. Thus, a difference in frequency of a term may either indicate a difference in the emotional state of the author or simply a difference in topic. If LIWC predictors were truly indicative of emotional state independent of topic, and fluctuations in emotional state predict corresponding fluctuations in health and behavior outcomes, it is difficult to reconcile the difference in the direction of predictions between conditions reported in that paper. Nevertheless, if one accepts that LIWC indicators are merely proxies that can be used for estimating measurement of psychological state within very narrowly constrained contexts, then the pattern makes sense. However, this limitation has strong negative implications for the applicability of LIWC indicators within naturalistic communication settings in which there is a wide variation in the communicative goals

motivating individual contributions, such as in naturalistic on-line learning environments where students may interact about a wide range of topics in connection with a variety of activities over time.

Analysis of collaborative learning interactions have demonstrated that what happens on the process level is important for predicting what cognitive benefits participants in a conversation take away from it (e.g., King 2007). More complex learning is supposed to occur in "spirals of reciprocity", where learners are intensely engaged with one another (Salomon and Perkins 1998). In particular, learners can attain new levels of understanding during interactions where more complex cognitive activities occur, such as analytical thinking, integration of ideas and reasoning. These include activities such as elaborating on content (e.g., Webb 1989), explaining ideas and concepts (e.g., Chi et al. 1994), asking thought-provoking questions (e.g., King 1998, 1999), argumentation (e.g., Kuhn 1991), resolving conceptual discrepancies (e.g., Piaget 1985) and modeling one another's cognitive states. These activities may not be adequately represented by patterns of individual turns taken out of context. Modeling these processes instead requires categorical coding schemes building on precise definitions of categories (see Chi et al. 1994). Trained human coders are able to consistently apply well-defined coding schemes across multiple contexts. However, we acknowledge that applying coding schemes like this by hand is extremely tedious. And effectively writing rules by hand to reliably match against complex patterns, which is an option provided by some corpus analysis environments, is difficult as well.

Running example: Process analysis of argumentative knowledge construction

The goal of this paper is to develop text classification technology to address concerns specific to classifying sentences or other units of text using multi-dimensional coding schemes developed for work in the area of CSCL. Specifically with respect to CSCL, often only detailed process analyses reveal plausible interpretations of the effects of instructional support in computer supported collaboration environments (e.g., Weinberger 2003). Thus, as a running example of the type of coding scheme TagHelper is designed to apply, we describe one that was developed within a high profile CSCL project, refined through multiple iterations, and used fruitfully to yield insights into collaborative knowledge building processes (Weinberger and Fischer 2006; Weinberger et al. 2005). Not only is this coding scheme well established in the CSCL community and has shown to be valid and reliable being applied manually in several studies, even by different researchers than those who developed it (e.g., Schoor and Bannert 2007), it also provides a unique resource for investigating the capabilities of text classification technology to be used in this context. The coding scheme encompasses several independent dimensions with different demands for coding. At the same time, the complex coding scheme is an example of a coding scheme that consumes tremendous amounts of resources in terms of training coders as well as for actually applying the coding scheme manually.

The coding scheme was developed by Weinberger and Fischer (2006) for the purpose of addressing the question of how computer-supported collaboration scripts could foster argumentative knowledge construction in online discussions. Argumentative knowledge construction is based on the idea that learners acquire knowledge through argumentation with one or more learning partners, by better elaborating the learning material and by mutually refining ideas. Computer-supported collaboration scripts are scaffolds implemented within the interface of CSCL environments to specify, sequence, and assign discourse activities to participants. For instance, scripts could be realized with text prompts

implemented in the text input windows of CSCL environments, or they could be realized with interface widgets that enable or constrain certain types of interactions.

Collaboration scripts may focus on specific aspects of collaboration (Kollar et al. 2005; O'Donnell and Dansereau 1992; Stegmann et al. 2007). In this way, computer-supported collaboration scripts may apply on specific dimensions of argumentative knowledge construction. For example, a script for argument construction could support learners to ground and warrant their claims, or a social collaboration script can facilitate socio-cognitive conflict and its productive dissolution (Weinberger 2003). In the work that provides the context for our running example, these and other computer-supported collaboration scripts were varied experimentally. Throughout the time this coding scheme was being developed, altogether more than 750 students of Educational Science at the Ludwig–Maximilians University of Munich participated mainly in groups of three in a series of studies (a minority of studies were made with individuals and dyads). Students in all collaborative experimental conditions had to work together in applying theoretical concepts from Attribution Theory (Weiner 1985) to three case problems and jointly prepare an analysis for each case by communicating via web-based discussion forums. For example, one of the cases was about a student named Michael failing in mathematics and consequently being subject to different attribution patterns of parents, teacher and himself. Participants were asked to discuss the three cases against the background of Attribution Theory and to jointly compose at least one final analysis for each case. One of the collaboration scripts supported a peer-review like sequence of activities including drafting initial analyses individually, giving and receiving structured feedback, responding to feedback, and finally writing an improved case analysis (Weinberger et al. 2005).

In the light of the broad variety of theoretical approaches and specific foci within the research areas of collaborative learning where argumentative knowledge construction has been explored, it has become evident that it must be evaluated on multiple process dimensions (Weinberger and Fischer 2006). Hence, the design of the coding scheme we work with in this article draws from a variety of theoretical approaches and focuses on several specific conceptualizations of argumentative knowledge construction. These include (1) epistemic activity, formal quality regarding argumentation, which further specializes into the (2) micro-level of argumentation and the (3) macro-level of argumentation, and (4) social modes of co-construction. Independent of these theoretically grounded dimensions, the segments have been coded whether they were or were not (5) a reaction to a previous contribution. For experimental reasons (to be able to conduct a manipulation check), there is also a (6) dimension on which student responses to script prompts are coded for appropriateness and a (7) "quoted" dimension, which distinguishes between new contributions and quoted text as it is typically being automatically copied in replies to previous messages. In accordance with the respective theoretical perspectives, the number of categories differs between dimensions from 2 (e.g., quoted) to 35 (e.g., epistemic).

More precisely, the coding scheme by Weinberger and Fischer (2006) includes seven dimensions.

1. *Epistemic activity* (35 categories). How learners work on the knowledge building task, e.g., what content they are referring to or applying in their analysis.
2. *Micro-level of argumentation* (4 categories). How an individual argument consists of a claim which can be supported by a ground with warrant and/or specified by a qualifier.
3. *Macro-level of argumentation* (6 categories). Argumentation sequences are examined with respect to how learners connect single arguments and create an argumentation pattern together (for example, consisting of an argument, a counter-argument, and an integration).

4. *Social modes of co-construction* (21 categories). To what degree or in what ways learners refer to the contributions of their learning partners.
5. *Reaction* (3 categories). Reactions to elicitation and consensus-building (classes which are in the social modes of co-construction dimension).
6. *Appropriate response to prompts in the learning environment* (4 categories). How learners make use of prompts, i.e., whether learners uses the scripted prompts in the intended manner.
7. *Quoted discourse* (2 categories). Distinguishes between new contributions and quoted contributions.

The complete process analysis we have conducted to date comprises about 250 discussions of the participants. Trained coders categorized each segment using this multi-dimensional coding scheme. Three groups of about six coders were each trained to apply the coding scheme to the discourse data. One quarter of the total (human) resources of the research project that produced this data was used for this hand coding of the collaborative process data used in our experiments during the years when this data was being collected and analyzed. It is because hand analyses like this are so arduous and time consuming that we believe technology for automating this process holds so much promise for accelerating research in this community. Note that future studies using the same materials can now use the classification models trained on the hand coded data so that the initial investment of time can dramatically reduce human effort in future studies.

From the complete coded corpus we have run our text classification experiments with a subset of this coded data, using altogether 1,250 coded text segments. In all cases, every segment was assigned exactly one code from each of the seven dimensions. Because the corpus we use has been coded with seven independent dimensions drawing on different types of knowledge, working with it provides a valuable opportunity for exploring how differences in the nature of the coding scheme applied to a text corpus affects the relative performance of alternative text classification algorithms. Nevertheless, the technology we employ is general purpose and can be applied to a wide range of coding schemes. Thus, this particular coding scheme should simply be regarded as an example of the level of sophistication that can be achieved with this technology.

Text classification approaches

Text classification is an application of machine learning technology to a structured representation of text. In the past decade and even earlier, research on text classification and text mining has been a major focus of research in the field of computational linguistics. Typical text categorization applications include assigning topics to news articles (Lewis et al. 2004), web pages (Craven et al. 1998), or research articles (Yeh and Hirschman 2002). Machine learning algorithms can learn mappings between a set of input features and a set of output categories. They do this by examining a set of hand coded "training examples" that exemplify each of the target categories. The goal of the algorithm is to learn rules by generalizing from these examples in such a way that the rules can be applied effectively to new examples. Work in this area has yielded impressive results in a wide range of application areas and allows working towards automating the application of categorical coding schemes.

As discussed above, the discourse of collaborative learners can be coded on multiple dimensions with multiple classes on each dimension. To contextualize our effort, we review other research on multi-class classification, where multi-class classification refers to

classification tasks where a choice must be made between three or more possible labels for each instance. From a high level, there are at least two promising strategies for working towards optimal performance at automatic multi-class classification. On the one hand, the *feature based approach* consist of the idea for identifying text features that generalize well across categories so that the rules that define what constitutes each code and distinguishes it from the others can be as simple as possible. On the other hand, the *algorithmic approach* is to develop more and more powerful algorithms with the capability to learn very subtle distinctions. We have used both approaches in our work. Thus, these two parallel threads will be a running theme throughout the remainder of this article.

One important contribution of this work is a demonstration that the feature based approach has a stronger and more consistent effect on classification performance across dimensions in the Weinberger and Fischer coding scheme. Thus we argue that the direction of seeking features that are useful for increasing performance on coding schemes that have been developed in the CSCL community might be the most promising direction for expanding the impact of the work presented in this article.

The feature based approach More attention to the selection of highly predictive features has been done for text classification problems where very subtle distinctions must be made or where the size of spans of text being classified is relatively small. Both of these are true for our work. Perhaps the most similar application to what we are addressing in this article is the work on *conversation act recognition*, where conversational contributions are classified in terms of the role the contribution plays in a running discourse. Classifying spoken utterances into dialogue acts or speech acts has been a common way of characterizing utterance function since the 1960s, and many automatic approaches to this type of analysis have been developed since (e.g., Serafin and Di Eugenio 2004; Stolcke et al. 2000). Other applications of sentence classification technology include identifying rhetorical relations in legal documentation (Hachey and Grover 2005) or distinguishing subjective versus objective statements (Wiebe and Riloff 2005). Some recent approaches focus on the problem of assigning sentences to classes that represent an idea that might occur within an essay (Rosé et al. 2003; Rosé and VanLehn 2005). In all of these applications, the unit of analysis is typically a single utterance rather than a whole document, which has an impact on what solutions prove most successful. Because of this, more emphasis is made on the selection of highly predictive features, such as indicative grammatical relations or inclusion of unique or colorful words, than on the algorithms employed. For example, Wiebe and colleagues (2004) describe a series of in-depth explorations of a wide range of linguistic feature types. These investigations involve features derived from grammatical relations, simple lexical characteristics, and shallow extraction patterns. The idea has been to search for features that can be reliably extracted from text and that provide high precision clues for distinguishing subjective and objective sentences. Rosé and VanLehn (2005) describe a comparison of sentence level text classification using only word level features with one that makes use of word level features combined with grammatical relations extracted from the texts using the CARMEL interpretation framework (Rosé 2000).

In this article we will explore the use of a variety of commonly used types of features, which are also available in the publicly downloadable version of TagHelper tools[1], in

[1] TagHelper tools can be downloaded from http://www.cs.cmu.edu/~cprose/TagHelper.html.

addition to some other special purpose features we will discuss later where we evaluate our feature based approach to exploiting context for increasing classification performance.

- *Punctuation*. Punctuation may be stripped out of the attribute space, or it can be used as a feature. Sometimes punctuation can be a useful predictor. For example, punctuation can be a proxy for the mood of a text, distinguishing questions like "you think the answer is that Michael does not like math?" from statements like "you think the answer is that Michael does not like math." It can also be a marker of uncertainty. Furthermore, the inclusion of a comma might mark that a contribution is relatively more elaborated than one without a comma.
- *Unigrams and bigrams*. A unigram is a single word, and a bigram is a pair of words that appear next to one another. Unigrams are the most typical type of text feature. Bigrams may carry more information. They capture certain lexical distinctions such as the difference in meaning of the word *stable* between "stable attribution" and "horse stable".
- *POS bigrams*. Part-of-speech bigrams are similar to the word bigrams discussed above except that instead of pairs of words, they are pairs of grammatical categories. They can be used as proxies for aspects of syntactic structure. Thus, they may be able to capture some stylistic information such as the distinction between "the answer, which is …" vs "which is the answer".
- *Line length*. Oftentimes lengthy contributions in chat data contain elaborated explanations, which are important to detect in learning science research. Thus, length of contribution can sometimes serve as a proxy for depth or level of detail.
- *Contains non-stop word*. This flag can be a predictor of whether a conversational contribution is contentful or not, which can be useful when processing chat data rather than newsgroup style data. For example, making a distinction between contributions like "ok sure" versus "the attribution is internal and stable". Often the categories that are appropriate for non-contentful contributions are distinct from those that apply to contentful ones. So this can be a useful distinguishing characteristic.
- *Stemming*. Stemming is a technique for removing inflection from words in order to allow some forms of generalization across lexical items, for example the words stable, stability, and stabilization all have the same lexical root.
- *Rare words*. Removing rarely occurring features is a simple way of stripping off features that are not likely to contribute to useful generalizations. This keeps the size of the feature space down, which aids in effective rule learning.

The algorithmic approach For coarse grained text categorization of large spans of text, such as whole documents or web pages, the primary focus has been on developing more and more powerful machine learning algorithms. Even in our case, where the spans of text are relatively small and the distinctions are in some cases fine grained and subtle, developing more powerful algorithms may have a substantial impact on performance. Here we review related work on developing effective algorithms for multi-class classification. We will explore two main lines of work on multi-class classification, both of which we will evaluate in this article. The first line of work focuses on approaches for effectively extending binary classification approaches into multi-class classification approaches. This type of approach necessitates resolving conflicts between individual binary classifiers that may provide contradicting predictions (Fuernkranz 2002). The second line of work is exploiting sequential dependencies between spans of text using sequential learning algorithms (Collins 2002; Lafferty et al. 2001).

As Fuernkranz (2002) describes the state-of-the-art in classification techniques, many contemporary learning algorithms are by nature binary, i.e., distinguishing between positive and negative examples, even though many real problems are multi-class classification tasks, i.e., distinguishing between many types of conversational contributions. The reason for the proliferation of binary classification techniques could be because of limitations inherent in popular classification algorithms or because of the prevailing paradigm for evaluating learning algorithms, which is that the goal is to learn a model from positive and negative examples. Generally, work on multi-class classification builds on and extends work on binary classification in different ways. In multi-class classification, the multi-class problem is broken down into multiple binary classification problems, and the solutions are then combined so that a single class label is assigned to a span of text. Below we describe our novel *Confidence Restricted Cascaded Binary Classification* approach (CR-CBC; Dönmez et al. 2005), which is a multi-class classification approach developed within this tradition.

As mentioned, another algorithmic approach that is relevant for our work is the recent development of *sequential learning techniques* (Lafferty et al. 2001; Collins 2002). These are approaches that attempt to gain discriminatory power by considering the context in which a span of text occurs, where the context is defined by the codes assigned to some number of previous spans of text. While this is a limited notion of context, it has proven useful for some discourse based classification tasks (e.g., Carvalho and Cohen 2005) because of the natural way in which it captures the notion of sequential relevance. Sequential relevance is the notion that one discourse act can set up the expectation than one or a small number of other discourse acts will be offered by a respondent either immediately or in close proximity to the initiating act (Schegloff and Sacks 1973). Specifically we experiment with the Collins Perceptron Learner (Collins 2002), which we have had success with in connection with a coding scheme designed to analyze synchronous tutorial dialogues, and which is a popularly used sequential learning algorithm. Because our most successful classification results have been with Support Vector Machines (SVM; Vapnik 1995), we also discuss results with an adaptation of the SVMstruct algorithm, which has been configured for sequential learning (Tsochantaridis et al. 2004).

Application of text classification approaches in CSCL

Very little work has been done so far on automating categorical corpus analysis within the CSCL community. However, in the broader field of educational technology, there has been quite a lot of research on using language technologies more generally, especially in the areas of automatic essay grading (Burstein et al. 1998, 2001; Page 1968; Page and Petersen 1995; Landauer and Dumais 1997; Foltz et al. 1998; Laham 2000) and tutorial dialogue systems (Graesser et al. 1998; Rosé et al. 2001; Aleven et al. 2003; Evens and Michael 2003; Litman et al. 2006; VanLehn et al. 2007). What is different about the work presented here is that we focus on an analysis of the process of conversation rather than the content.

Nevertheless, while the specific goals of our work encompass new problems within the field of CSCL, some notable first steps towards the more specific goal of automatic process analysis of conversational data have been made previously within that sphere. For example, Soller and Lesgold (2000) and Goodman and colleagues (2005) present work on automatically modeling the process of collaborative learning by detecting sequences of speech acts that indicate either success or failure in the collaborative process. Similarly, Cakir et al. (2005) present an approach for identifying sequential patterns in coded collaborative learning data. What is different about our approach is that we start with raw text and detect features within the text itself that are indicative of different local aspects of

the collaboration. Erkens and Janssen (2006) present an approach using "hand constructed" rules in Dutch for a single dimensional coding scheme consisting of 29 categories from five basic communication act types. While this work is no small achievement and has practical value, it is specific to a single coding scheme, and works only in Dutch. In contrast, we present a general approach where rules for novel coding schemes can be learned from tagged examples, which can rather easily be ported to additional languages. Thus, rather than presenting a competing approach, we present an approach that is distinct and complementary to that presented in prior work related to collaborative learning process analysis.

Obviously a fully-automatic or even semi-automatic system, which could support coding of the natural language corpus data, would facilitate and potentially improve categorical forms of corpus analysis. The analysis of discourse in studies on collaborative learning could be simplified and can potentially be made faster. The role of the researcher in the analysis of discourse processes can be reduced primarily to simply checking the automatic coding and making corrections if necessary, freeing researchers' resources for other tasks that cannot easily be automated in the research process, like drawing conclusions from the automatic coding.

Methodological issues related to automatic corpus analysis

Automatic and semi-automatic process analysis is a relatively new direction for the field of CSCL, and as such requires some scrutiny from a methodological perspective. Important issues must be addressed such as validity, i.e., whether the automatic coding accomplished by the computer is really capturing the essence of what was intended by human analysts who designed the scheme, and reliability, i.e., how faithfully the automatic codes match those of expert human analysts. These are important issues if this automatic coding is to be used to draw conclusions with respect to important research questions related to collaborative processes. Issues of efficiency must also be addressed. Because some hand coded data must be provided to train the prediction models, and because some codes cannot currently be assigned reliably even with substantial training data, which necessitates checking the automatically coded data for errors, it is prudent to consider how to identify the circumstances under which a substantial savings of time and effort can be achieved using automatic text processing support, and in which circumstances it is preferable to conduct the analysis manually.

Validity

When human coders apply categorical coding schemes, they bring insights with them from their human intellect. Human language is highly complex, encoding meaning on multiple levels, and carrying very subtle nuances that are difficult to formally capture with a rule-based model. Interpretation of language involves using cultural sensitivity to style and lexical choice, applying world knowledge, integrating meaning across spans of text, and often making inferences about what is implied in addition to what is literally stated. In contrast, regardless of approach, machine coding will always be based on rigid rules that are necessarily an over-simplification of the reasoning processes that humans rely on for their interpretation. Note that word counting approaches such as LIWC, which were discussed earlier, are an extreme case of this over-simplification. This simplification threatens the face validity of the coding that can be accomplished automatically because

this word based approach may not be measuring what it is purported to be measuring. Using an example from our own work, we have used LIWC to examine the language behavior of five different tutors who participated in a series of calculus problem solving studies (Gweon et al. 2006). We evaluated tutor effectiveness by comparing them with respect to the average learning gains of the students they tutored. Based on this analysis, we determined that the more effective tutors scored higher on LIWC's confidence scale. When we examined which words from the tutors' contributions the associated LIWC word list was matching against, the most frequent word was "factor", which came up inside discussions about algebra. Thus, the LIWC confidence scale was not ranking tutors based on their confidence at all, but rather their tendency to supplement their calculus tutoring with basic algebra concepts such as factoring. Thus, word counting approaches like LIWC that make their assessment based on individual words taken out of context should be used with caution. We see from our calculus example that they are not guaranteed to reflect accurately the mental states they were designed to assess.

On the other hand, in order to achieve acceptable human agreement with application of categorical coding schemes, it is often necessary to limit both the extent that context is taken into account and the extent to which inferences are made beyond what is literally stated in text. Thus, even with this type of analysis, there may be similar validity concerns since important nuances from distant context may be missed. By explicitly reflecting upon the extent to which subjectivity is used in making judgments, one can evaluate the extent to which this concern threatens the validity of the coding that can be achieved automatically. This suggests that analysts should carefully consider the level of subjectivity in their judgment when deciding whether to rely on automatic coding support.

The primary issue that raises questions with respect to validity in automatic coding is that the cognitive process by which human analysts assign labels to spans of text according to a human designed categorical coding scheme can never fully be replicated by a computer. Because of this, even when an acceptable level of agreement is reached between a human coder and an automatic coder, it will likely be true that the cases where humans are most likely to disagree with each other are not the same as those where a computer is most likely to disagree with a human. In our experiments, computer algorithms make some mistakes that would be highly unlikely to be made by a human, while it is also true conversely that they are able to detect subtle regularities in judgments that would go unnoticed by a human analyst. This may have implications for the conclusions that will be drawn from the automatic coding. Logically, however, one must consider that the number of cases where idiosyncratic errors occur in automatic coding must necessarily be small in those cases where agreement between human coders and automatic coders is high. Thus, while it will always be the case that automatic coders follow a different process from the cognitive processes human coders engage in, we can address this potential threat to validity by seeking to increase the reliability of the automatic coding. Furthermore, one can explicitly evaluate where these disagreements are occurring during a validation stage in the analysis process and use this error analysis as a basis for determining whether it is safe to believe the conclusions that are drawn from the automatic coding. Note that human coding can also suffer from similar validity issues, especially in cases of "over training", where coders rely on very shallow text features in order to artificially boost their level of agreement with other human coders.

In the remainder of this section we explore some specific aspects of our coding scheme where issues of validity are raised. These are meant to serve as examples of the type of consideration that is needed when considering an automatic coding approach.

Ontological versus linguistic consistency: The epistemic dimension In the running example on process analysis of argumentative knowledge construction, the discourse data were coded on multiple dimensions using the coding scheme discussed earlier (Weinberger and Fischer 2006). Here we discuss specifically how it was coded with regard to the epistemic activity dimension. On this dimension, argumentative knowledge construction processes are analyzed with respect to the questions of how learners work on the task, including information on what content they are referring to. Thus, this is necessarily dependent to some extent on the specific task that this coding scheme was designed for. Categories are defined in terms of task specific knowledge. One important distinction on the epistemic activity dimension is to what extent learners work on the task or digress off task. There were 35 separate categories on this dimension, 18 of which have 10 or fewer instances in the corpus of 1,250 segments, which is less than a tenth of one percent of the corpus. The design of the set of codes on this dimension followed from the idea that in order to solve a problem, learners may need to *construct a problem space*, *construct a conceptual space*, and *construct relations between the conceptual and problem space*.

With the *construction of the problem space*, learners are to acquire an understanding of the pedagogical content knowledge related to the problem they are working on. Therefore, this dimension was coded to indicate when learners select and elaborate individual components of the problem case information. The *construction of the conceptual space* serves to communicate an understanding of the theory they are learning, in this case specifically Attribution Theory. Thus, most of the categories on this dimension correspond to a specific pair of concepts. When learners connect individual theoretical concepts or distinguish them from one another, the code associated with this pairing of ideas is assigned to their contribution. The *construction of relations between conceptual and problem space* indicates to what extent learners are able to *apply* theoretical concepts adequately. This code was assigned to segments including concept-problem information pairs (i.e., one item from the problem space and one item from the conceptual space). Overall, 27 different relations between conceptual and problem space were distinguished in the coding scheme.

While 27 of the classes in the Epistemic dimension represented specific content expressed connecting evidence from case studies with concepts from the theory students were applying to their analysis, some of the remaining categories were very differently construed. For example, one category was created to indicate off-topic conversation, and another was for epistemic activity that included concepts *not* specifically related to the given theory. On a linguistic level, these epistemic activities might look highly different but nevertheless belong to the same category. For instance, "In my opinion, the parents exemplify something through their own life. The son Michael imitates this by model-based learning." and "Michael simply has no interest in Math" would both be coded with the same category although their meaning is very different. Within both sentences, a theoretical concept (model-based learning vs interest) is used to explain Michael's actions. However, these concepts are not part of the conceptual space related to Attribution Theory, which the students were supposed to be applying. These theoretical concepts were derived from prior-knowledge instead of deduced from the learning material in the current unit. Therefore, they were coded as "application of prior-knowledge concepts to case information".

The difficult classes were those defined by what they were *not* rather than what they were. They did not refer to a specific topic or idea. From a linguistic standpoint, these categories are defined quite differently from the other 27, although ontologically they all belong on the epistemic dimension. Because of this, it is these categories that are most difficult to achieve an acceptable level of performance with automatic coding. For example, learners may draw on a broad range of prior knowledge and experiences, which are hard to predict, charter, and

integrate into the model that is being trained by the machine learning algorithms. What they are capable of doing is recognizing indications of typical applications of prior knowledge that were labeled as such in the training data. Thus, to the extent that students are consistent about which prior knowledge they apply, and to the extent that they express this in consistent ways, the trained model may appear to be functioning properly. But the trained models will break down in cases where students possess unusual prior knowledge that will enable them to contribute unique ideas not appearing in the training data. A human analyst familiar with the domain would easily recognize these as applications of prior knowledge, but an automatic model would not be capable of this.

Context dependence: The social dimension The social modes of co-construction dimension indicates to what degree or in what ways learners refer to the contributions of their learning partners (see Weinberger and Fischer 2006). In this dimension there are five types of social modes with an increasing degree of referring to the contribution of their learning partners, namely externalization, elicitation, quick consensus building, integration-oriented consensus building, and conflict-oriented consensus building. The prevalence of the type of referring behavior where students build on one another's ideas has been found to be positively correlated with outcomes of collaborative knowledge construction (Teasley 1997). Learners may explicate their knowledge, e.g., by contributing a new analysis of a problem case. *Externalizations* are statements introducing only new ideas or topics, and neither refer to preceding contributions of peers nor aim to elicit information from the learning partners. Learners may use the learning partner as a resource and seek information (*elicitation*) within the discourse in order to better analyze a problem case. Learners need to build at least a minimum consensus regarding the learning task in a process of negotiation in order to improve collaboration. There are different styles of reaching consensus, however. *Quick consensus building* means that learners accept the contributions of their partner prematurely without any discussion. *Integration-oriented consensus building*, in contrast, means that learners approximate and integrate each other's perspective, synthesize their ideas, and jointly try to make sense of a task. *Conflict-oriented consensus building* takes place when learners critically negotiate perspectives in objecting or rejecting the contributions of their partners during the process of consensus building. The 5 general categories on the social modes dimension subsume 21 more specific categories, seven of which have ten or fewer instances in the corpus. These seven categories with fewer than ten instances in this corpus represent less than a tenth of one percent of the corpus altogether. Therefore, also an aggregated version of the coding scheme for this dimension was used. In the aggregated version each of the main categories of the social modes of co-construction dimension subsume between two to six more specific categories from the original, more fine-grained version.

There are several issues related to distinguishing categories on this dimension that are potentially challenging for automated process analyses. One difficulty arises from the fact that the degree to which learning partners refer to the contributions of their learning partners obviously depends much on what has been contributed before. For example, the contribution "Michael simply is lazy!" is an externalization when in an initiating contribution. However, if this sentence is in response to "Michael believes that he is not talented", the same sentence is coded as conflict-oriented consensus building. Furthermore, the same sentence would be coded as quick consensus building if it is in response to "The reason in the case of Michael in fact is laziness", because the repetition is a kind of acceptance. In addition, instances of integration-oriented and conflict-oriented consensus building can differ very subtly from one another. For example, "It is his laziness, but lack of talent would also fit" would normally be coded as

integration-oriented consensus-building, but not if a partner simply suggested that laziness should be considered as in "It is his laziness, lack of talent would fit less well".

In the simplest automatic approaches, single segments of text are considered in isolation when assigning a code automatically. Thus, the model by which codes are assigned automatically does not have access to the primary knowledge that a human would rely upon to make the distinction. Instead, the trained model relies entirely upon regularities in how text segments are coded in the training data. Thus, the trained model may be able to pick up the subtle cues that distinguish "but lack of talent would also fit" from "lack of talent would fit less well", but would not be able to distinguish cases where the context would dictate that "Michael is simply lazy!" should be an externalization instead of conflict-oriented consensus building. To the extent that regularities are found in the order in which ideas tend to be contributed to group discussions and pieces of evidence tend to emerge in specific contexts, a sequential learning approach may achieve a certain level of performance with a simple representation of the text, although the process by which codes are assigned is entirely different from that of the human process. Because of these limitations of a simple classification approach that only considers characteristics of individual text segments out of context in the construction of the feature space, may not achieve an acceptable level of performance.

Reliability

Because achieving high reliability is one way of safeguarding against the threats to validity discussed in the previous section, reliability of coding is explicitly measured and scrutinized as part of a our methodology. Reliability of categorical coding schemes is typically evaluated using the Kappa statistic (Cohen 1960), which measures the amount of agreement there is between two codings of the same data, controlling for agreement by chance. Standards for acceptable levels of agreement differ between sub-communities of behavioral researchers. A Kappa value of 0.4 is an acceptable level of agreement according to Fleiss and Cohen (1973). However, that is substantially lower than the more typical standard of 0.8 or at least 0.7, which is advocated by Krippendorf (1980). We advocate upholding Krippendorff's more stringent standard of reliability with automatic coding because of the possible threats to validity discussed above. Recently Krippendorff (2004) has criticized the usage of Kappa in cases where the distribution of categories is very different between coders since in this case, the Kappa value will tend to appear higher than is justified. Nevertheless, where Kappa values are high, and percent agreement is even higher, any differences in distribution of categories will normally be small in magnitude. Because Cohen's Kappa is used in most of the studies in CSCL, we prefer this indicator if the preconditions are fulfilled. And indeed, in our experiments reported in this paper, we verified that the magnitude of any differences in distribution of categories between the gold standard coding and the automatically generated codes was negligible. Nevertheless, use of other metrics that do not fall prey to the same shortcomings is also advisable.

High reliability in coding may be achieved in at least two ways. One is by making the technology more accurate. The other is by checking over some or all of the automatically coded data to ensure that it has been coded in a reasonable way. This second approach clearly comes with a practical cost in terms of time, which we will address explicitly in the next subsection. But it raises other questions as well in terms of a potential negative bias that may be introduced in the mind of a human coder when exposed to the errorful automatic coding, to the extent that they may be more inclined to leave a code unchanged

unless they are certain that it is incorrect. These concerns have been explored in Gweon et al. (2005). Gweon et al. (2005) present a lab study in which they compare analysts coding texts with no predicted codes with analysts correcting codes where a random sampling of 50% of the texts were coded correctly, and the other 50% were coded with a randomly selected incorrect code. The performance of the analysts in the two conditions was compared with respect to speed and accuracy. While there was no detectable difference in coding speed, there was a significantly higher accuracy in the case of the analysts that were provided with automatic coding predictions, although these predictions were wrong 50% of the time. Nevertheless, while these are encouraging results, since the coding scheme used in the Gweon et al. study was considerably simpler than the Weinberger and Fischer coding scheme, it is possible that the results would not generalize completely. Further experimentation is necessary to completely eliminate the concern that under some circumstances a negative bias could be introduced by providing analysts with errorful predicted codings. However, these results do provide some reassurance that incorrect predictions can be detected and corrected by human analysts.

Efficiency

Since efficiency is the primary motivation for automatic corpus analysis, efficiency should always be seriously evaluated when selecting between a fully manual approach, a fully automatic approach, or a semi-automatic analysis approach. Efficiency must be taken into account at three stages in the process, namely the stage at which some training data is coded by hand, the stage at which the reliability of the coding is checked, and at the stage when automatically coded data is checked and potentially corrected. If the first option of a fully manual approach is not to be selected, the total amount of time spent with these three activities should not exceed the amount of time it would take to code the entire corpus by hand. For fully automatic coding, more time will typically be spent in the first stage than in the case where automatic coding will be checked and possibly corrected since more training data is required to build a model that achieves a high enough level of reliability. While more time is spent in the initial phase, the pay-off comes later since any amount of additional data can then be coded with no human effort. In this case, the big payoff comes when many studies are conducted with the same materials or the same coding scheme. In the other case where it is assumed that a human analyst will check and correct the automatic coding, less time is typically spent in the initial phase.

Gweon et al. (2005) found that with a specific type of menu-based coding interface where the automatic predictions may be viewed at all times and changed with a simple menu selection, the break even point for checking and correcting codes was 50%. The amount of time that was saved from simply checking a code rather than coding from scratch was the same as the amount of time that was wasted if a checked code turned out to be incorrect and needed to be corrected. Typically 50% coding accuracy is not difficult to achieve with automatic classification technology. Thus, normally some advantage can be achieved with automatic predictions with a relatively small investment in time to set up the training data. The magnitude of that advantage depends upon the proportion of data coded by hand as well as the accuracy of the resulting prediction model. The exact amount of hand coded data required varies depending upon the composition of the training data. If the patterns in the data that characterize the desired distinctions are very consistent, then less hand coding is typically required. In cases where the coding scheme is much more complex, or the patterns are highly idiosyncratic, then more hand coding is typically required.

An analyst need not commit prematurely to one or the other of these approaches. Rather, an analyst may code a portion of data and then automatically code the remainder of the data using the trained model. That analyst may then choose to check and correct only a subset of the automatically coded data. That corrected data can then be added to the training set to create a larger training set, which can then be used to train a more accurate prediction model to use to replace the codes on unchecked data. This "bootstrapping" approach can be used to limit the amount of data that must be coded by hand to the minimum required to achieve an acceptable performance even in the case where this amount cannot be determined a priori. Additional data can be coded by hand in small increments until an acceptable performance is reached.

A novel algorithmic approach for avoiding mistakes on difficult cases

Applying automatic text classification technology to coding schemes such as the Weinberger and Fischer (2006) coding scheme comes with challenges beyond those typically faced in prior applications of text classification technology to problems such as dialogue act tagging mentioned earlier. Typically, the coding schemes developed for collaborative process analysis are primarily motivated by theoretical considerations and may be dependent to some extent on contextual features related to the task or tasks they were designed in connection with. This top-down approach provides a solid foundation for defining what the categories should be and what they mean. However, there are challenges that come from a theoretical rather than an empirical foundation for the design of categories. In particular, some of the categories of conversational events may never or rarely occur in the actual data.

The typical approach taken within the computational linguistics community in the work related to dialogue act tagging has conversely been far more empirical. Although the abstract idea of a speech act is a general linguistic notion motivated by theory, in practice the set of dialogue acts that have been the target of computational linguistics work on dialogue act tagging have been largely empirically motivated. Thus, the problem of skewed distributions of coding categories, which is already a problem in connection with tasks such as dialogue act tagging, is nevertheless less of an issue than it is in tasks such as automatic collaborative learning process analysis. Other challenges have already been outlined above (see section "Methodological issues related to automatic corpus analysis"). Thus automatic process analysis is far from a trivial application of text classification technology. In this section we discuss how we have addressed some of the technical challenges we have faced in our work. We began our exploration using as a test-bed the Minorthird text-learning toolkit (Cohen 2004), which contains a large collection of configurable machine learning algorithms that can be applied to text classification tasks. In our later experiments leading up to the development of the current TagHelper tools package, we have also used the Weka toolkit (Witten and Frank 2005). These two toolsets have provided a convenient framework in which to conduct our research. We measure our success in terms of agreement with the hand-coded gold standard corpus with the help of the Kappa statistic as an accepted standard for measuring coding reliability. Our criterion for success is reaching a level of agreement with a gold standard as measured by Cohen's Kappa that is 0.7 or higher, since this is a recognized and rather high cut-off criterion for acceptability in terms of reliability of coding in behavioral research communities. We discuss this issue in greater depth in section "Methodological issues related to automatic corpus analysis."

The first technique we developed and evaluated in our previous work (Dönmez et al. 2005) was aimed at improving the accuracy of an algorithm referred to as the Voted Perceptron classification algorithm. Voted perceptrons are known to perform well on text, as are Support Vector Machines (SVMs), which we make use of in the work we report later in the paper. For purposes of this paper, it is not necessary for readers to understand the details of these two algorithms. What is important to note is that our novel classification algorithm, which we refer to as *cascaded binary classification* (CBC) (Dönmez et al. 2005), uses the voted perceptron algorithm as a building block. In this approach, we apply the binary classifiers according to their rank order in terms of accuracy over a separate set of data, and assign a code corresponding to the first binary classifier in the rank ordering that predicts positive. One can think of this as an approach to avoiding errors on low frequency classes or classes where there is a high likelihood of making a mistake.

The baseline approach of standard Voted Perceptron classification achieved an acceptable Kappa value with respect to dimensions macro-level of argumentation (κ= 0.70), reaction (κ=0.84), appropriateness of the response to prompts in the learning environment (κ=0.70), and quoted (κ=0.91). Further, early explorations of the cascaded binary classification algorithm showed some improvement on the dimension of the micro-level of argumentation (κ=0.76). The epistemic dimension (κ=0.49)and the dimension of the social modes of co-construction (κ=0.55) remained recalcitrant. A further finding was that it was possible to increase our reliability on these two dimensions by only committing a code to the subset of data where the most reliable classifiers predicted positive identification. We refer to this modified approach as *Confidence Restricted Cascaded Binary Classification* (CR-CBC). With this approach, we were able to achieve a kappa of 0.68 on the social modes of co-construction dimension over just the subset of data (50%) where a code was assigned. However, the severe limitation of that approach was that we were not able to identify the most important codes on that dimension in the data, nor were we able to assign a code to half of the data. Thus, while the cascaded binary classification approach showed promise to reducing the amount of time it would take a human to code the data, it would not be acceptable for on-line monitoring of collaborative learning interactions. If an approach like this were used that missed the most important codes, at best it would not present useful information from its on-line monitoring, and at worst it would present a misleading view of the interactions it was processing.

Evaluating a feature based and algorithm based approach to exploiting context in automatic coding

On the social modes of co-construction dimension where the definitions of the categories refer to the context in which a text segment appears, the purely algorithmic approach discussed in the previous section failed to serve as a satisfying solution, especially with respect to the ultimate goal of using on-line analysis to trigger adaptive interventions to support collaborative learning. We conjectured that the reason why we failed to find a fully satisfying solution in a purely algorithmic approach was that the codes on the most recalcitrant dimensions must rely on making inferences from the context in which a segment of text appears. In this section we explore ways in which we can leverage the context in the automatic analysis. Clearly it is not practical to attempt to take context into account in a formal way by computing every possible logical connection between each span of text and every preceding span in search of connections. Here we report on our experimentation with two main ways of using context information, one involving an

extension to the feature space used and the other employing an extension of the basic algorithmic approach.

What we mean by feature space is the set of features extracted from the texts that we provide to the classification algorithms to use in making their predictions. For example, a feature space consisting of only unigram features, would have a feature corresponding to each word that ever occurred in the corpus. For each instance in the data, the value of each feature would be *one* if the corresponding word ever occurred in the corresponding span of text and *zero* otherwise. This manner of representing texts with collections or vectors of features is familiar within the computational linguistics community, but may be far less familiar to readers from the CSCL community.

In all experiments presented in this section, we employ a consistent methodology of tenfold cross-validation, which is a standard evaluation methodology in the computational linguistics community. In this approach, we first divide the data into 10 subsets, where each data point is randomly assigned to one of the ten subsets. Each of the ten subsets are then used in turn as testing data, with the other nine subsets concatenated together and used as training data. In this way, we can use the maximum amount of data for training and yet avoid testing on the same data we trained on. A simple feature selection algorithm called chi-squared attribute selection is applied to the training data on each iteration to rank the features. The top 100 of these features is used as input to the training algorithm to build the model that is then applied to the testing set. Narrowing down the set of features provided to the classification algorithm is a good way to bias it to learn general rules. The average performance over the whole set of data is then computed by averaging the performance obtained using this methodology on each of the 10 testing iterations. In order to reduce variance due to idiosyncrasies in random distribution of data into the ten subsets used for cross-validation, we perform this cross-validation ten times, and average performance across these ten runs.

In the remainder of this section we present our process of first selecting a baseline classification algorithm, then selecting the composition of a baseline feature space, and then systematically comparing the contribution of a subset of novel context based features in comparison with an algorithmic approach to leveraging context. This same decision making process could be used by readers in their own exploration using tools such as TagHelper tools with their own data and their own coding scheme. As a methodological point the reader should keep in mind that there is a danger in using the same set of data in repeated rounds of experimentation in this way that the resulting configuration could in some way be tuned to idiosyncrasies of the set of data used in the experimentation. Thus, as an added validation step to ensure the generality of the result, it is prudent to evaluate the resulting configuration on a completely independent set of data if one is available. In the absence of such a validation, the reader should keep in mind that the absolute value of the level of performance achieved may appear slightly higher than it would be on a completely independent set of data. Nevertheless, all of our comparisons reported here are valid with respect to the relative performance between approaches since in all cases, all factors other than what is being manipulated are held constant.

Note that for the experiments in this section, we have utilized an aggregated version of the social modes of co-construction dimension of the coding scheme. These aggregated codes refer to the five alternative forms of consensus building behavior discussed in "Context dependence: the social dimension". Further, we like to note that in our experiments reported in this paper, we verified that the magnitude of any differences in distribution of categories between the gold standard coding and the automatically generated codes was negligible. Hence, we prefer Cohen's kappa, which is broadly used

in research in CSCL, instead of using the more seldom Krippendorf's alpha for our reliability tests.

Baseline results

We begin by exploring the best performance we can get with these three standard classification algorithms across all seven dimensions using the simplest possible feature space, specifically a feature space composed of unigram features, which were defined earlier in the article as features that correspond to single words found in the texts. We have used three standard classification algorithms that are widely available in off-the-shelf machine learning packages, including both the Minorthird and Weka packages we have used for our own experimentation, namely Naïve Bayes (NB), the Weka implementation of support vector machines, which is referred to as SMO, and decision trees (DT). These algorithms are straightforward to apply and produce models that can be examined in order to determine which pieces of information were useful in making their classifications. They also work well on a variety of types of data that we have experimented with. While these algorithms are powerful, this is not enough to achieve good performance. Beyond this, what is needed is a good set of features. In other words, features that are strongly predictive and general enough that they can be used to build effective classification rules. Figure 1 displays the relative performance of the three standard algorithms over the seven different dimensions of the coding scheme. Table 1 contains the mean and standard deviation of the best performing of these three algorithms over each of the seven dimensions. The lowest performing dimension is the social modes of co-construction dimension. Note that with this simple feature space representation, we only achieve an acceptable kappa on three dimensions.

The performance statistics displayed in Fig. 1 were computed using the simplest possible feature space. However, as described previously in the article, TagHelper tools provides

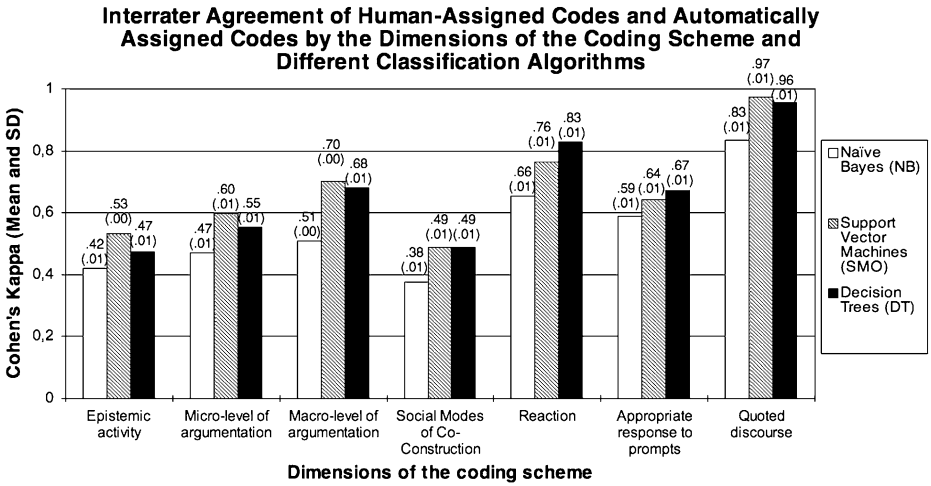


Fig. 1 This *bar chart* displays the performance in terms of kappa agreement between human assigned codes and automatically generated codes on all seven dimensions of the Weinberger and Fischer coding scheme using Naïve Bayes (NB), support vector machines (SMO), and decision trees (DT). The standard deviation in all cases is less than 0.01

Table 1 Best average kappa obtained using support vector machines (SMO) or decision trees (DT), using the top 100 unigram features on each of the seven dimensions of the Weinberger and Fischer (2006) coding scheme

Dimension	Used algorithm	Best average kappa	Standard deviation
Epistemic activity	SMO	0.53	0.004
Micro-level of argumentation	SMO	0.60	0.034
Macro-level of argumentation	SMO	0.70	0.004
Social modes of co-construction	SMO	0.48	0.011
Reaction	DT	0.82	0.009
Response to prompts	DT	0.67	0.006
Quoted discourse	SMO	0.97	0.009

functionality for customizing the feature space. A typical approach is to experiment with a broad range of combinations of available features to determine which combination provides the classification algorithms with the most leverage. This serves two purposes. First, it gives the algorithms the best advantage in terms of performance. And second, examining which types of features provide the best computational advantage can provide information about the nature of the data since features that provide leverage are features that distinguish data in one class from data in another class.

Using the functionality readily available on the TagHelper tools interface, we were able to compare performance with eight different feature spaces, as displayed in Fig. 2. This set of eight different combinations of types of features, all of which can be extracted from texts using TagHelper tools, systematically samples the space of possibilities in a broad but shallow manner. This broad sampling approach allows us to quickly find an effective combination of types of features made available by TagHelper tools. Those eight feature spaces include: Unigrams, Unigrams plus a line length feature, Unigrams plus part-of-speech bigrams, unigrams plus bigrams, unigrams plus punctuation, unigrams with

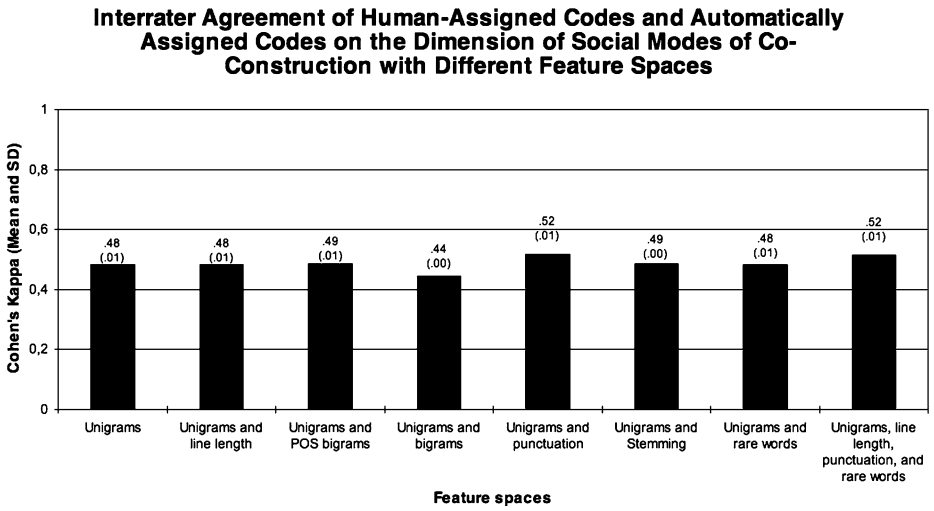


Fig. 2 This *bar chart* displays level of performance achieved by support vector machines (SMO) on the social-modes of co-construction dimension using alternative feature sets constructed with TagHelper tools. The standard deviation in all cases is less than 0.01

stemming, unigrams with rare features removed, and unigrams plus the length feature with rare features removed. We see here that adding bigrams substantially reduces performance. This is most likely because there are many more bigram features than unigram features, so each one is relatively rare, making generalization more difficult. However, despite not being very useful in this specific case, they substantially increase the feature space size. Normally, a substantial increase in feature space size will make it more difficult for the algorithm to converge on an effective model unless the added features have high predictive value. From this exploration, we settle on unigrams plus punctuation as our set of base features to use for experimentation in the remainder of this section.

The focus of our experimentation in this section is leveraging the context in which a span of text occurs in order to increase the performance of the automatic corpus analysis. The dimension of the social-modes of co-construction, and the dimensions of the macro-level and micro-level of argumentation are those in which one can imagine context playing an important role in classification. While the categories on the epistemic dimension can be determined rather independently of other features than those, which could by found in the very same segment, the remaining three dimensions may benefit from additional context-based features. Thus, we investigate alternative approaches for leveraging context across all three of these dimensions. From a scientific standpoint, this comparison across dimensions is interesting since these dimensions may refer to the context of a segment of text in different ways. And thus, it may be the case that the approach that works best on each of these dimensions will differ. We will begin with an illustration of why this might be the case. Figure 3 presents an example thread with two messages, each contributed by a different student. Within each message we see multiple segments, each of which are assigned a code on three different dimensions that we are concerned with in this section, namely social-modes of co-construction, macro-level of argumentation, and micro-level of argumentation. The first two segments, contributed by Robin, constitute one message,

Author	Social Modes	Macro	Micro	Text	German
Robin	Externalization	Argument	Claim	Michael blames his poor achievements on a lack of giftedness in mathematics.	Michael macht seine mangelnde Begabung auf dem Gebiet der Mathematik fuer seine dauerhaft schlechten Leistungen verantwortlich
Robin	Externalization	Argument	Warrant	From this one can conclude that his attribution is internal and stable. Internal because it comes from within himself. And stable because it is something that can't be changed.	Daraus kann man folgern dass hier eine internal stabile Attribution stattfindet Internal weil sie im Schueler selbst lokalisiert ist
Cornelia	Quote	Quote	Quote	> Michael blames his poor achievements on a lack of giftedness in mathematics. From this one can conclude that his attribution is internal and stable. Internal because it comes from within himself. And stable because it is something that can't be changed.	>>> Michael macht seine mangelnde Begabung auf dem >>> Gebiet der Mathematik fuer seine dauerhaft schlechten >>> Leistungen verantwortlich Daraus kann man folgern dass hier >>> eine internal stabile Attribution stattfindet >>> Internal weil sie im Schuele
Cornelia	Quick consensus	Non-argumentative: Evaluation	Nothing	Wow, that was a really good work. Right on!	Wow was fuer eine Bearbeitung Respekt
Cornelia	Conflict-oriented	Integration	Claim	From the case I could not however directly infer that Michael thinks the task is too difficult for him. Instead I thought Michael thinks that he is too dumb for mathematics.	Aus dem Fall konnte ich aber nicht direkt ableiten dass Michael die Aufgaben fuer zu schwer haelt ich dachte eher dass er sich egal was fuer eine Aufgabenstellung einfach fuer zu doof fuer Mathe haelt
Cornelia	Conflict-oriented	Non-argumentative: Planing	Nothing	Therefore, I did not include something about that in my contribution.	Deswegen habe ich das auch nicht in meine Bearbeitung eingefuegt

Fig. 3 Example coded segment of text consisting of two messages, namely a thread initial message consisting of two segments, and a child message consisting of five segments

which is a thread initial message. The second two segments, contributed by Cornelia, constitute a second message, which is a child message of the first message.

If we examine the texts of the segments as well as the codes assigned on the three dimensions, we see that in some cases, there is enough evidence in the linguistic structure of a text segment itself to determine what would be a reasonable code to assign. For example, note that the Integration that is in the third segment of Robin's message is identifiable as such even out of context because of the linguistic markers that flag it as such. However, other determinations require considering the context in which a segment appears. For example, because Robin's message is a thread initial message, the contributions cannot be building on those of the learning partners, so these must count as externalizations on the social-modes of co-construction dimension. Thus, there are constraints that come from where a message appears within the thread structure. Furthermore, some constraints appear to result from the sequential placement of segments within a message. For example, the assignment of the first two segments to claim and warrant, rather than the reverse, namely warrant and then claim may largely be due to the ordering of those two segments. We assume the second segment is meant to support the first. Additionally, it requires considering the relationship between the segment and the context to identify Cornelia's third segment as being conflict-oriented. In a different context, such as when a learning partner suggested that some content was not appropriate to include in a case analysis, this could signal agreement rather than disagreement.

Structure of the data

Let us now consider the structure of the data we are working with, which enables us to leverage context in different ways. Figure 4 illustrates the two level structure of our newsgroup style data. The circles represent messages, and the lines represent parent-child relationships between messages in the threaded discussion board. The bubbles attached to some circles display a close up view of the text contained within the attached messages represented by the circles. Notice that the message from Cornelia responds to the message from Robin, and thus is further down the thread structure from it.

As indicated within the bubbles that present a close up view of the message texts, each message is composed of several segments. In our coding methodology, each message is segmented into spans of text referred to as epistemic units (Weinberger and Fischer 2006). As already discussed above, each of these units of text is then assigned one code from each of seven dimensions in our multi-dimensional coding scheme. Thus, each message potentially has a sequence of codes on each dimension, one code per dimension for each unit of text. Thus, there are two levels of context information in the structured data that we have. First, there is the course grained thread structure, with parent-child relationships between messages. And secondly, there is the sequence of codes that are assigned to units of text within a message. We draw upon both of these sources of context information in our approach.

Feature based approach

Thread structure features One of the contributions of our work is the construction of novel types of features that can be extracted from our newsgroup style data, and which reflect the threaded structure of that data. A similar previous approach is one where hand-coded annotations reflecting discourse structure were used to improve performance on a dialogue act tagging task (Serafin and Di Eugenio 2004). The simplest context-oriented feature we can add based on the threaded structure is a number indicating the depth in the thread where a message appears. We refer to this feature as *depth*. This is expected to improve

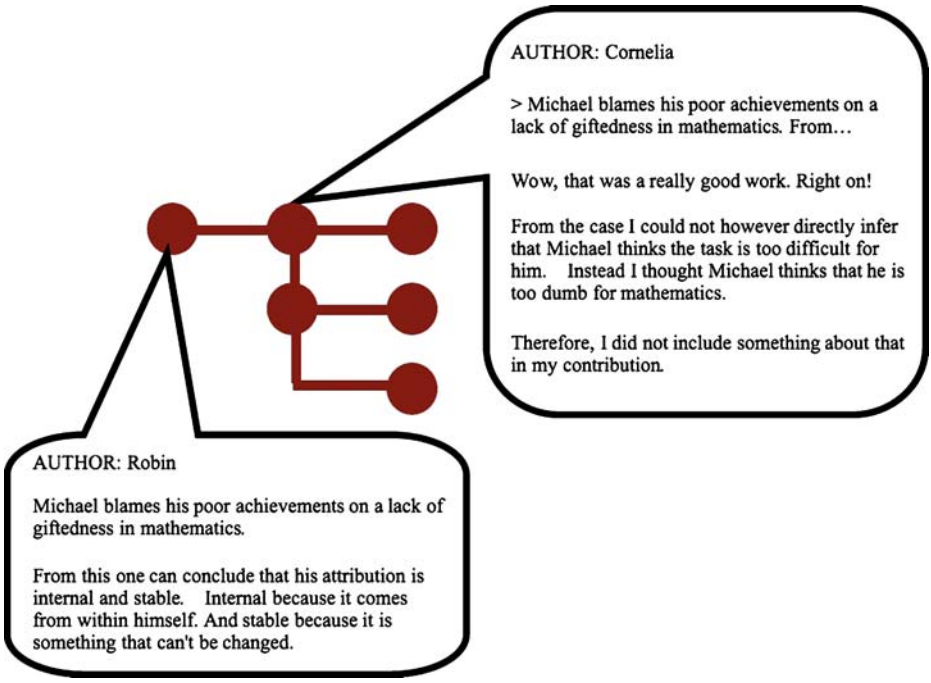


Fig. 4 This figure displays the two level structure of the newsgroup style interaction in the Weinberger and Fischer corpus. Note that each oval represents a message. The black oval is a thread initial message. The *large circles* display what is inside of a message, specifically that each message is composed of sequence of segments of text

performance somewhat since some codes within the aggregated version of the social modes of co-construction coding dimension never appear in thread initial messages. Other context oriented features related to the thread structure are derived from relationships between spans of text appearing in the parent and child messages. One such feature is meant to indicate how semantically related a span of text is to the spans of text in the messages higher up on the thread that were posted by other participants. This is computed using a simple vector similarity measure referred to as the cosine distance between the vector representation of the span of text and that of each of the spans of text in the parent message or any message higher up on the thread. The value of this feature is the smallest such distance. This feature can be thought of as indicating how related the current span of text is to something in the discourse context contributed by a different student.

Sequence-oriented features

We hypothesized that the sequence of codes of the aggregated version of the social modes of co-construction dimension within a message follows a semi-regular structure, as illustrated in Fig. 5.

In particular, the CSCL environment inserts prompts into the message buffer that students use. Students fill in text underneath these prompts. Sometimes they quote material from a previous message before inserting their own comments. We hypothesized that

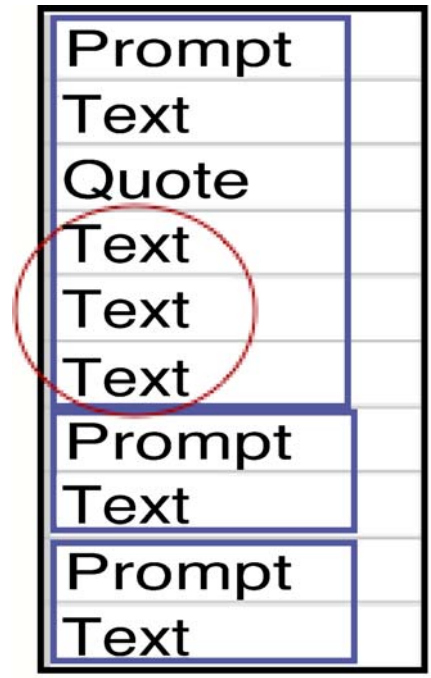


Fig. 5 This figure displays the structure of a message. Note that there are three types of spans of text, namely prompts (from the collaboration script), quotes (from previous messages on the thread), and text (new segments contributed in the message). The *circled portion* is a region where the text is likely to refer to previously contributed segment of text, since it occurs underneath a quote

whether or not a piece of quoted material appears before a span of text might influence what code of the aggregated version of the social modes of co-construction dimension is appropriate. Thus, we constructed the *fsm* feature, which indicates the state of a simple finite-state automaton that only has two states. The automaton is set to initial state (q_0) at the top of a message. It makes a transition to state (q_1) when it encounters a quoted span of text. Once in state (q_1), the automaton remains in this state until it encounters a prompt. On encountering a prompt it makes a transition back to the initial state (q_0). The purpose of this is to indicate places where student comments fall between quoted material and the next prompt, since these are regions where students are likely to make a comment in reference to something another student has already said. In Fig. 5, the segments that correspond to this state are circled.

Table 2 presents a summary of our predictions about which ways of leveraging context would be effective on which dimensions, and which ways proved to be effective in practice, respectively.

Evaluating context-oriented features The final feature space representation combines features provided by the TagHelper tools in addition to the context oriented features described above. Our evaluation demonstrates that our proposed context oriented-features can increase kappa statistics in predicting the aggregated version of the social modes of co-construction dimension. For this evaluation, we compared the same classification algorithm with four different sets of features. One is only trained with the baseline features extracted directly from TagHelper tools that we determined as a first step in our decision making process to be the best choice for a baseline set of features. Three other feature spaces evaluated here include one that include the thread structure features, one that includes the

Table 2 Predictions about which types of features will lead to a significant improvement in performance on which dimensions as well as results from experimentation demonstrating which approaches to leveraging context were in fact effective on which dimensions

Dimension	Thread features		Sequence features		Sequential learning	
	Predicted effect	Actual effect	Predicted effect	Actual effect	Predicted effect	Actual effect
Micro-level of argumentation	Yes	Yes	Yes	Yes	Yes	Yes
Macro-level of argumentation	No	Yes	Yes	No	Yes	No
Social modes of co-construction	No	Yes	Yes	No	Yes	No

sequence-oriented features, and one that includes both. This allows us to test the hypotheses expressed above regarding the separate effects of the two types of context based features we have created. We predicted that both forms of context based features would yield a significant improvement on the social modes of co-construction dimension. However, we predicted that on the two argumentation dimensions that thread structure features would have relatively little effect. In contrast, we expected that we would see an effect of the sequence oriented feature on the two argumentation dimensions because our impression of the data was that there were typical ways in which parts of complex arguments were arranged into logical sequences. The results showed significant improvements from both types of features, however not all of our predictions turned out to be correct, as displayed in Table 2, which highlights the importance of an experimental approach in applying text classification technology in a specific context. An experimental process offers new insight into the composition of ones data and reveals places where intuition may turn out to be incorrect.

To run our evaluation, we used Weka's implementation of support vector machines, which is referred to as SMO (Witten and Frank 2005). Results are shown in Fig. 6.

We can see that the kappa value increases from 0.52 to 0.69 on the social modes of co-construction dimension. All pairwise contrasts were statistically significant. The best result included all of the context based features, but the biggest effect was achieved using the thread structure features. This was consistent with our expectations. On the other two dimensions, however, only thread structure features successfully produced a statistically significant improvement, which was contrary to our prediction. In particular, thread structure features were effective at increasing performance across dimensions. We did not expect to see this in the two lower level argumentation dimensions that mainly encode structural relationships within single messages. However, in hindsight, we understand that some portions of complex argumentation, like counter-examples, are more likely to refer to already mentioned information, whereas others, such as claims are not. Contrary to expectation, sequence oriented features only had an effect on the social modes of co-construction dimension, which indicates that students were not as formulaic in their ordering or parts of complex arguments as our informal impression of the data had indicated. Besides offering evidence that improvements in performance can be obtained through creation of new types of features, these results show how machine learning experiments can be used to gain greater insight into the structure and composition of ones data.

Evaluating the sequential learning approach

Despite the fact that sequence-oriented features showed no effect on the two argumentation dimensions, we predicted an effect of sequential learning on these two dimensions. Our reasoning was that sequential learning can capture ordering constraints between codes in a more general way than our sequence oriented feature, which is specific to the placement of codes in close proximity to quoted material. For example, we predicted that students might offer claims before warrants. Similarly, we expected that students might present counter-arguments after an argument. Thus, because sequential learning algorithms capture ordering constraints in a general way, we predicted that if we would see an effect of sequential learning on any of the three dimensions, it would be more likely to be on the argumentation dimensions, and not the social modes of co-construction dimension.

Specifically we tested our baseline features and augmented feature set using the Collins Perceptron Learner (Collins 2002), which we have had success with in connection with a coding scheme designed to analyze synchronous tutorial dialogues. By setting the history size to 0, the Collins Perceptron Learner behaves like a non-sequential learner. With a history size greater than that, it behaves like a sequential learner, taking into consideration the previous codes leading up to the current segment. We achieved the best results with a history size of 1. However, even with this setting, we were not able to exceed the performance we achieved with SMO augmented with context oriented features, as displayed in Fig. 7 in comparison with results displayed in Fig. 6.

For the comparison presented in Fig. 7, we tested four separate configurations: (1) base features no history, (2) base context features no history, (3) base features history of 1, and (4) base context features history of 1. Similar to the comparison presented in Fig. 6, using context oriented features was significantly better than not using context oriented features. However, the improvement from using the history was only statistically significant in the case where only base features were used. This could have potentially been predicted since

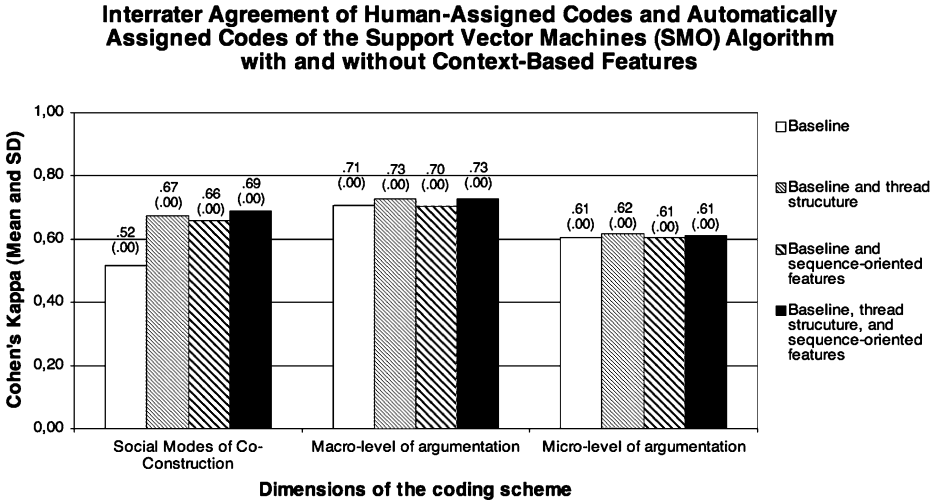


Fig. 6 This bar chart displays the relative performance of support vector machines (SMO) on three dimensions (i.e., social modes of co-construction, macro-level of argumentation, and micro-level of argumentation), using four different feature sets (i.e., base features only, base features plus thread structure features, base features plus sequence features, and base features plus both thread structure and sequence features). The standard deviation in all cases is less than 0.01

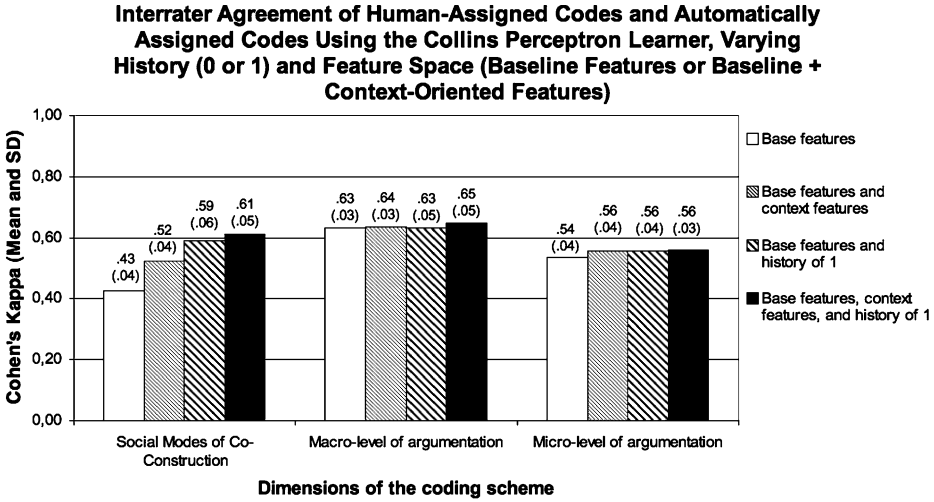


Fig. 7 This *bar chart* displays the relative performance of the Collins Perceptron Learning on three dimensions (i.e., social modes of co-construction, macro-level of argumentation, and micro-level of argumentation), using two history settings (i.e., 0 length history and 1 length history) and two different feature sets (i.e., base features only, and base features plus both thread structure and sequence features)

the context features in some sense remove the need for the context provided by the sequential learning algorithm. However, we do not observe any effect of sequential learning on the two lower level argumentation dimensions, contrary to our prediction. Thus, we find further evidence that students were not very consistent with respect to the order in which they contributed the component parts of their argumentation.

Since SMO is a maximum margin learning algorithm and the Collins Perceptron Learner is not, we also ran the above experiment using a version of SVM that can be configured for sequential learning, namely SVMstruct (Tsochantaridis et al. 2004). However, this experiment similarly failed to produce a statistically significant improvement in performance for sequential learning in any dimension using either feature space.

Thus, with the current data set, our finding is that for the examined dimensions of the used coding framework the context oriented features, especially thread structure features, are more important for improving classification performance than the use of more sophisticated machine learning technology, such as sequential learning algorithms. Furthermore, we see the value in taking an experimental approach to feature space design, since our experimentation revealed places where our impressions of how students were constructing their arguments based on informal observations turned out not to be accurate.

Automating discourse segmentation

Segmentation of discourse means that the text is being divided into units of analysis or so called segments. Some of these units of analysis are set by the participant in the discourse and typically do not need to be agreed upon by raters, e.g., messages, sentences, or even words. However, one of the most challenging aspects of the Weinberger and Fischer (2006) coding scheme, both for human annotation and automatic annotation, is the type of fine-grained segmentation. Rather than corresponding to linguistic structural features of the contributions, the rules for the segmentation developed by Weinberger and Fischer (2006)

are based on the information conveyed. One unit of text is the amount of text it takes to express something that counts as an epistemic activity, i.e., may be assigned one code on the Epistemic dimension. Because of this, the unit of analysis is referred to as an "epistemic unit". Often, an epistemic unit is a single sentence. However, it happens very frequently that either more than one sentence or less than one sentence counts as an epistemic unit. Thus, punctuation alone does not turn out to be a very reliable predictor for this segmentation. Note that the accuracy of segmentation might substantially alter the results of the categorical analysis because it can have a dramatic effect on the feature based representation that the text classification algorithms base their decisions on. In the analyses presented above, we used pre-segmented discourse material, i.e., the smallest units to be coded on the seven dimensions were still identified by human analysts. We have, however, started to work towards automatic segmentation to enhance the capabilities of automatic and semi-automatic coding. For example, automated real-time analyses to support groups directly or by a facilitator who uses the results of the automated coding are only possible if segmentation is also done automatically.

In our current approach, we use a "sliding window" of three symbols, which may either be words or punctuation, to extract decision points from our data. This technique is illustrated in Fig. 8. In this approach, the window first contains the first three tokens in the text, namely "I lost overview". In the next iteration, it contains tokens two through four, which in this case includes "lost overview somehow". In the third iteration it contains tokens three through five, which in this case is "lost overview!". Each of these windows corresponds to one decision point, which is either labeled as a boundary instance or a non-boundary instance. Whenever there is a boundary between the first and second token within a window, the instance corresponding to that window is coded as a boundary. In other cases, it is coded as not a boundary.

Just as in the case where we were training models to assign codes from our coding scheme to spans of text, we need to extract features from the text in order to enable an effective model to be learned. We extracted a number of features related to the instances to use as predictors for the boundary/non-boundary distinction.

Here is a list of the features we used for prediction, which were largely motivated by our knowledge about the grammatical structure of German:

1. The three symbols within the window
2. A binary indicator that notes whether punctuation occurred adjacent to the second symbol
3. A binary indicator that notes whether there have been at least two capitalized words since the last boundary
4. A binary indicator that notes whether there have been at least three non-capitalized words since the last boundary
5. A binary indicator that notes whether we have seen fewer than half the number of symbols as the average segment length since the last boundary

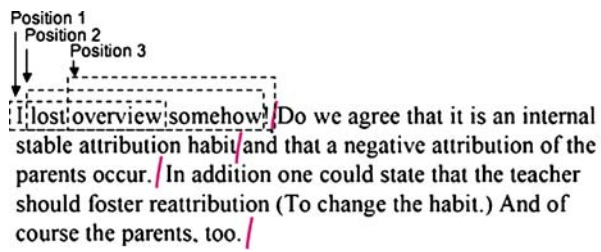


Fig. 8 This is an illustration of the sliding window technique used for segmentation

6. A binary indicator that notes whether we have seen fewer than half the average number of symbols between punctuations since the last punctuation mark

We trained a prediction model using the J48 algorithm, which is one of Weka's Decision Tree (DT) learning algorithms, and evaluated its performance using cross validation. We achieved a percent accuracy of 96%, which corresponds to a precision of 0.59, a recall of 0.37, and a kappa of 0.44, where precision is percentage of predicted boundaries that are in fact boundaries, and recall refers to the percentage of correct boundaries that are predicted as boundaries. The automatic segmenter assigns fewer boundaries overall than were assigned by hand. Overall it assigned only 66% as many as the human annotators assigned. Clearly this is sub-optimal performance, and we are still exploring ways in which the segmentation results can be improved.

Conclusions and current directions

In this paper, we presented an overview of our work on automatic collaborative learning process analysis from the past years laying the foundation for a new technology. The specific objective of the interdisciplinary project has been to explore the intersection between the technology research area of text classification and the behavioral research area of CSCL. Beyond simply being an application of existing technology to a well defined framework for discourse analysis, this collaboration has yielded interesting new technical challenges and solutions (Dönmez et al. 2005; Rosé et al. 2005; Stegmann et al. 2006; Wang et al. 2007), questions about behavioral research methodology (Gweon et al. 2005), and finally questions about design of new forms of collaboration support that may be enabled by this technology (Gweon et al. 2006; Wang et al. 2007b; Kumar et al. 2007). Moreover, beyond developing technology to further our own research agenda, our vision has been to provide tools to the broader community of researchers who collect and code corpus data as an important part of their research.

Our specific goal has been to extend and apply current text classification technology to CSCL, exploring which classification techniques are most effective for improving the performance on different types of coding dimensions used in the CSCL community. We have given an overview of this problem and have introduced some of our work in this area. From the perspective of computational linguistics, our results to date are encouraging but they also demonstrate that the technology requires additional improvement. In particular, new approaches need to be explored to improve reliability over some remaining difficult dimensions. We have presented our investigations towards increasing performance on the social modes of co-construction dimension as well as the Micro-level and Macro-level of Argumentation dimensions. For the social modes of co-construction dimension and the micro-level and macro-level of argumentation dimensions, we have proposed and evaluated two different approaches for improving classification performance with a context-oriented coding scheme. We showed that our novel context-oriented features indeed can improve the performance of various learning algorithms, including both non-sequential and sequential ones. However, we did not observe a general increase in performance due to using a sophisticated sequential learning algorithm such as the Collins Perceptron Learner. We believe an important generalization of our findings could be that effectively applying machine learning to the problem of automatic collaborative learning process analysis requires designing or selecting appropriate features that can approximate the linguistic mechanisms that are implicit in the design of the categorical coding scheme that is used. Thus, insight into the structure of the language behavior itself

that is captured in the coding scheme is what is most needed to move forward with this line of research.

In sum, the work presented in this article should be seen as evidence that large areas of research on computer-supported collaborative learning, specifically that involving systematic analyses of discourse, can benefit strongly from exploiting recent advances in computational linguistics. In one example domain (educational psychology), we showed that some processes, which are highly valued in CSCL research, could be automatically identified in text messages with a level of reliability considered acceptable for agreement between human coders. Among these processes are, for example, transactively referring to each other, formulating counter-arguments, or collaboratively applying scientific concepts to solve problems. Of course, further research must be done to ensure validity and generalizability of these results.

However, we see these results as encouraging with respect to more economically analyzing collaboration processes, to support human instruction in real time, and to more dynamically implement computer-supported instruction, e.g., instruction involving collaboration scripts. With the help of text classification technology, instructional support such as computer-supported collaboration scripts (Kollar et al. 2006) could be faded in or out of CSCL environments in a much more dynamic, context sensitive way than it is currently possible. Our recent evaluations of simple forms of this dynamic support have demonstrated substantial benefits of this type of support (Kumar et al. 2007; Wang et al. 2007b).

Another possible direction for applying machine learning technology in support of corpus analysis work would be to explore the trained models more deeply to determine which features provide the greatest predictive power for the classification. The three machine learning algorithms provided by TagHelper tools all produce models that can be examined from this perspective. If it turned out that one or a small number of key words provided most of the predictive power for replicating a human coder's analysis, this might indicate that the human coder was making judgments based on superficial characteristics of the text rather than using human insight. Depending upon the nature of the coding scheme, this might raise validity concerns about the human coding. Thus, machine learning could potentially be used as a tool to evaluate the quality of human coding.

Our work has the potential for impact beyond the boarders of the CSCL community. A new application area for employing text classification technology, and which may provide an interesting avenue for taking our work full circle from the computational linguistics community, to the CSCL community, and back to the computational linguistics community, is the emerging area of *conversation summarization*. Conversation summarization is a relatively new area of computational linguistics, building on a long history of expository text summarization. While typical applications of summarization technology are largely oriented towards extracting the most contentful sentences from a document, or collecting the most contentful sentences across multiple documents reporting about the same event (Kupiec et al. 1995; Carbonell and Goldstein 1998; Gong and Liu 2001), conversation summarization is different. Behavioral studies about dialogue summarization show that what people consider important to include in a summary about a dialogue may include aspects of the nature of the conversation in addition to a condensed version of the information that was communicated (Roman et al. 2006). Insights gained from process analysis of conversation logs from collaborative learning studies could enable the construction of summaries to support group moderators who do not have time to follow all of the details of every conversation occurring in parallel in an on-line learning environment. Current work in conversation summarization is already moving in this direction (Zechner 2001; Zhou and Hovy 2006; Wang et al. 2007). We have already begun

to make progress towards using TagHelper tools to enable the development of monitoring tools for group learning facilitators (Rosé et al. 2007; Joshi and Rosé 2007).

Acknowledgement This work has grown out of an initiative jointly organized by the American National Science Foundation and the Deutsche Forschungsgemeinschaft to bring together educational psychologists and technology experts from Germany and from the USA to build a new research network for technology-supported education. This work was supported by the National Science Foundation grant number SBE0354420 to the Pittsburgh Science of Learning Center, Office of Naval Research, Cognitive and Neural Sciences Division Grant N00014-05-1-0043, and the Deutsche Forschungsgemeinschaft. We would also like to thank Jaime Carbonnel, William Cohen, Pinar Dönmez, Gahgene Gweon, Mahesh Joshi, Emil Albright, Edmund Huber, Rohit Kumar, Hao-Chuan Wang, Gerry Stahl, Hans Spada, Nikol Rummel, Kenneth Koedinger, Erin Walker, Bruce McLaren, Alexander Renkl, Matthias Nueckles, Rainer Bromme, Regina Jucks, Robert Kraut, and our very helpful anonymous reviewers for their contributions to this work.

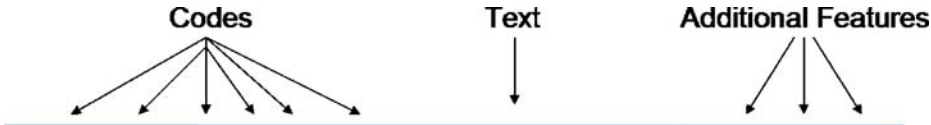
Appendix

TagHelper application

TagHelper's basic classification functionality is available to researchers to use in their own work. Because we have observed the popular usage of Microsoft Excel as an environment in which behavioral researchers commonly do their corpus analysis work, we have adopted that as our standard file format. The TagHelper application and documentation can be downloaded free of charge from http://www.cs.cmu.edu/~cprose/TagHelper.html. Note that the version of TagHelper that is publicly available at the time of writing this article is designed to work with pre-segmented English, German, or Chinese texts, although additional licensing is required for enabling the Chinese functionality. And we plan to add other languages as potential users express interest. In this section we illustrate the process by which an analyst can use TagHelper for an analysis task. Setting up the data for analysis is simple. As in the example in Fig. 9 below, analysts can prepare their data by reading the text segments into a single column in an Excel worksheet. That column should be labeled "text". Each column to its left will be treated as a dimension in the coding scheme. Additional columns to the right are treated as extra features that can be used in the predictive model, such as the context based features evaluated above. Wherever a "?" appears, the corresponding text will be treated as uncoded on the associated dimension, which TagHelper tools will code using a model trained from the coded examples. Any other value appearing in that column will be treated as a specific code. And the associated example will be treated as an example representing that code.

Once the data is prepared and loaded into TagHelper, a number of customizations can be performed that may affect the classification performance. The analyst can experiment with these different options to find a setting that works well with their data. Interested parties may contact the first author to enquire about workshops, tutorials, and on-line training courses related to effectively using this functionality.

With the click of a button, TagHelper then trains a model for each dimension using the coded data and then applies this model to all of the uncoded instances. It then inserts those predicted labels into the output file and additionally inserts a column with confidence values for all of the predictions. These confidence values can be used by analysts to identify instances that are most likely to contain errors so that analysts can be strategic about how they spend their coding time. After checking and correcting a portion of the automatically labeled data, the analyst may remove the predicted labels from the remaining examples as



AllDimensions.xls

	A	B	C	D	E	F	G	H	I	J
1	soc	atol	rea	quote	pro	epi	text	deep1	deep2	FSMstate
2	soc.201	?	?	quote.0	pro.0	epi.6	Michaels Eltern demotivieren ihren Sohn	2	2	0
3	soc.201	?	?	quote.0	pro.0	epi.10	indem sie seine Misserfolge mit einer mangelnden eigenen Begabung erklaeren die schon in der Familie liegen soll	2	2	0
4	?	atol.149	rea.140	quote.0	pro.0	epi.15	Das geschaffte Schuljahr hat er wahrscheinlich seiner Lehrerin zu verdanken die ihn zu groesserem eigenen Engagement motivierte	2	2	0
5	soc.0	atol.149	rea.0	quote.0	pro.0	epi.13	indem sie seinen Misserfolg auf variable Ursachen seine Faulheit zurueckfuehrte	2	2	0
6	?	atol.146	rea.140	quote.0	pro.0	?	es ist guenstig misserfolge auf variable ursachen wie anstrengung zurueckzufuehren nicht aber auf stabile wie mangelnde begabung	2	2	0
7	soc.223	atol.0	rea.140	?	pro.0	epi.232	ps es stoert hoffentlich nicht dass ich alles KLEIN schreibe	2	2	0
8	soc.0	atol.0	rea.0	?	pro.0	epi.0	>>> Michaels Eltern demotivieren ihren Sohn indem >>> sie seine Misserfolge mit einer mangelnden eigenen >>> Begabung erklaeren die schon in der Familie liegen soll >>> Das geschaffte Schuljahr hat er wahrscheinlich >>> seiner Lehrerin zu verdanken die ihn zu groesserem >>> eigenen Engagement motivierte indem sie seinen Misserfolg >>> auf variable Ursachen seine Faulheit zurueckfuehrte	3	2	1
9	soc.39	atol.0	rea.140	quote.0	pro.0	epi.228	der meinung bin ich auch gut gemacht	3	2	1
10	soc.201	atol.149	rea.140	quote.0	pro.0	epi.6	Ich denke dass die Eltern von Michael laut Attributionstheorie nicht gerade guenstig reagiert haben	2	2	0

Fig. 9 This figure illustrates how to set up data for input to TagHelper tools in Excel. Note that the *highlighted* cells show how to indicate that an example is uncoded. The uncoded examples will be assigned a code by TagHelper by comparing those examples to the already coded examples

well as removing the columns containing confidence values, and then may load the file back into TagHelper for another round of training.

TagHelper tools is an application that makes use of the functionality provided by the Weka toolkit (Witten and Frank 2005). In order to use the customizations available in TagHelper tools purposefully it is important to understand that machine learning algorithms induce rules based on patterns found in structured data representations. Internally, each row of the input file provided to TagHelper will be converted into what is known as an instance inside of Weka. An instance is a single data point. It is composed of a list of attribute–value pairs. An analyst has two types of options in customizing the behavior of TagHelper. One is to manipulate the structured representation of the text, and the other is to manipulate the selection of the machine learning algorithm. These two choices are not entirely independent of one another. An insightful machine learning practitioner will think about how the representation of their data will interact with the properties of the algorithm they select. Interested readers are encouraged to read Witten and Frank's (2005) book, which provides a comprehensive introduction to the practical side of the field of machine learning. Interested readers may also contact the first author about enrolling in a distance course that provides both conceptual instruction as well as mentoring on the process of using this technology in ones own work.

The first customization that analysts may manipulate in TagHelper is the selection of the machine learning algorithm that will be used. Dozens of options are made available through the Weka toolkit, but some are more commonly used than others, and thus we have only included the most commonly used ones in TagHelper tools. The three options that are most recommended to analysts starting out with machine learning are Naïve Bayes (NB), which

is a probabilistic model, SMO, which is Weka's implementation of Support Vector Machines, and J48, which is one of Weka's implementations of a Decision Tree (DT) learner. SMO is considered state-of-the-art for text classification, so we expect that analysts will frequently find that to be the best choice.

The remaining customization options affect the design of the attribute space. The standard attribute space is set up with one binary attribute per unique word in the corpus such that if that word occurs in a text, its corresponding attribute will get a value of 1, otherwise it will get a value of 0. Other features may be added, or the baseline features may be modified or paired down, in order to bias the algorithms to learn rules that are more valid or conceptually more closely related to the definitions of the target categories. The full range of feature options are explored earlier in this article in Section "The feature based approach".

By default, TagHelper tools is configured to write an extra file to its output directory with a report about how accurate it estimates its predictions are from the coded data provided. The analyst can use this functionality to experiment with several alternative settings of features and algorithms in order to determine which configuration produces the best performance with the analyst's data.

References

Aleven, V., Koedinger, K. R., & Popescu, O. (2003). *A tutorial dialogue system to support self-explanation: Evaluation and open questions. Proceedings of the 11th International Conference on Artificial Intelligence in Education (AI-ED 2003)* pp. 39–46. Amsterdam: IOS Press.

Berkowitz, M., & Gibbs, J. (1983). Measuring the developmental features of moral discussion. *Merrill-Palmer Quarterly*, *29*, 399–410.

Burstein, J., Kukich, K., Wolff, S., Chi, L., & Chodorow, M. (1998). Enriching automated essay scoring using discourse marking. *Proceedings of the Workshop on Discourse Relations and Discourse Marking*, Annual Meeting of the Association of Computational Linguistics, Motreal, Canada, pp. 15–21.

Burstein, J., Marcu, D., Andreyev, S., & Chodorow, M. (2001). Towards automatic classification of discourse elements in essays. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, Toulouse, France, pp. 98–105.

Cakir, M., Xhafa, F., Zhou, N., & Stahl, G. (2005). Thread-based analysis of patterns of collaborative interaction in chat. *Proceedings of the 12th international conference on Artificial Intelligence in Education (AI-Ed 2005)*, Amsterdam, The Netherlands, pp. 120–127.

Carbonell, J., & Goldstein, J. (1998). The use of MMR, diversity based reranking for reordering documents and producing summaries, Proceedings of ACM SIG-IR 1998.

Carvalho, V., & Cohen, W. (2005). On the collective classification of email "Speech Acts." *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* pp. 345–352. New York: ACM Press.

Chi, M. T. H., de Leeuw, N., Chiu, M. H., & LaVancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science*, *18*(3), 439–477.

Cohen, J. A. (1960). Coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, *20*, 37–46.

Cohen, W. (2004). *Minorthird: Methods for identifying names and ontological relations in text using heuristics for inducing regularities from data.* Retrieved from http://minorthird.sourceforge.net.

Collins, M. (2002). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pp. 1–8.

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., et al. (1998). Learning to extract symbolic knowledge from the World Wide Web. *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, pp. 509–516.

De Wever, B., Schellens, T., Valcke, M., & Van Keer, H. (2006). Content analysis schemes to analyze transcripts of online asynchronous discussion groups: A review. *Computers and Education*, *46*, 6–28.

Dillenbourg, P., Baker, M., Blaye, A., & O'Malley, C. (1995). The evolution of research on collaborative learning. In E. Spada, & P. Reiman (Eds.) *Learning in humans and machine: Towards an interdisciplinary learning science* (pp. 189–211). Oxford: Elsevier.

Dönmez, P., Rosé, C. P., Stegmann, K., Weinberger, A., & Fischer, F. (2005). Supporting CSCL with automatic corpus analysis technology. In T. Koschmann, D. Suthers, & T.-W. Chan (Eds.) *Proceedings of the International Conference on Computer Supported Collaborative Learning—CSCL 2005* (pp. 125–134). Taipei, TW: Lawrence Erlbaum.

Erkens, G., & Janssen, J. (2006). Automatic coding of communication in collaboration protocols. In S. A. Barab, K. E. Hay, & D. T. Hickey (Eds.) *Proceedings of the 7th International Conference of the Learning Sciences (ICLS)* (vol. 2, (pp. 1063–1064)). Mahwah, NJ: Lawrence Erlbaum Associates.

Evens, M., & Michael, J. (2003). *One-on-one tutoring by humans and machines*. Mahwah, NJ: Lawrence Earlbaum Associates.

Fischer, F., Bruhn, J., Gräsel, C., & Mandl, H. (2002). Fostering collaborative knowledge construction with visualization tools. *Learning and Instruction*, *12*, 213–232.

Fleiss, J. L., & Cohen, J. (1973). The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, *33*, 613–619.

Foltz, P., Kintsch, W., & Landauer, T. (1998). The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, *25*, 285–308.

Fuernkranz, J. (2002). Round robin classification. *Journal of Machine Learning Research*, *2*, 721–747.

Goodman, B., Linton, F., Gaimari, R., Hitzeman, J., Ross, H., & Zarrella, J. (2005). Using dialogue features to predict trouble during collaborative learning. *Journal of User Modeling and User Adapted Interaction*, *15*(102), 85–134.

Gong, Y., & Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis, Proceedings of ACM SIG-IR 2001.

Graesser, A. C., Bowers, C. A., Hacker, D. J., & Person, N. K. (1998). An anatomy of naturalistic tutoring. *Scaffolding of instruction*. In K. Hogan, & M. Pressley (Eds.) . Brooklyn, MA: Brookline Books.

Gweon, G., Rosé, C. P., Albright, E., & Cui, Y. (2007). Evaluating the effect of feedback from a CSCL problem solving environment on learning, interaction, and perceived interdependence. *Proceedings of CSCL 2007*.

Gweon, G., Rosé, C. P., Wittwer, J., & Nueckles, M. (2005). An adaptive interface that facilitates reliable content analysis of corpus data. *Proceedings of the 10th IFIP TC13 International Conference on Human-Computer Interaction (Interact'05)*, Rome, Italy.

Gweon, G., Rosé, C. P., Zaiss, Z., & Carey, R. (2006). Providing support for adaptive scripting in an on-line collaborative learning environment. *Proceedings of CHI 06: ACM conference on human factors in computer systems*. New York: ACM Press.

Hachey, B., & Grover, C. (2005). Sequence modeling for sentence classification in a legal summarization system. *Proceedings of the 2005 ACM Symposium on Applied Computing*, pp. 292–296.

Henri, F. (1992). Computer conferencing and content analysis. In A. Kaye (Ed.) *Collaborative learning through computer conferencing: The Najaden papers* (pp. 117–136). Berlin: Springer.

Hmelo-Silver, C., & Chernobilsky, E. (2004). Understanding collaborative activity systems: The relation of tools and discourse in mediating learning, *Proceedings of the 6th International Conference of the Learning Sciences (ICLS)*. Los Angeles, California pp. 254–261.

Joshi, M., & Rosé, C. P. (2007). Using transactivity in conversation summarization in educational dialog. In *Proceedings of the SLaTE Workshop on Speech and Language Technology in Education*.

King, A. (1998). Transactive peer tutoring: Distributing cognition and metacognition. Computer-supported cooperation scripts. *Educational Psychology Review*, *10*, 57–74.

King, A. (1999). Discourse patterns for mediating peer learning. In A., O'Donnell, & A. King (Eds.) *Cognitive perspectives on peer learning*. New Jersey: Lawrence Erlbaum.

King, A. (2007). Scripting collaborative learning processes: A cognitive perspective. In F. Fischer, I. Kollar, H. Mandl, & J. M. Haake (Eds.) *Scripting computer-supported collaborative learning: Cognitive, computational, and educational perspectives*. New York: Springer.

Kollar, I., Fischer, F., & Hesse, F. W. (2006). Collaboration scripts—a conceptual analysis. *Educational Psychology Review*, *18*(2), 159–185.

Kollar, I., Fischer, F., & Slotta, J. D. (2005). Internal and external collaboration scripts in webbased science learning at schools. In T. Koschmann, D. Suthers, & T. W. Chan (Eds.) *Computer supported collaborative learning 2005: The next 10 years* (pp. 331–340). Mahwah, NJ: Lawrence Erlbaum.

Krippendorf, K. (1980). *Content analysis: An introduction to its methodology.* Beverly Hills: Sage Publications.

Krippendorff, K. (2004). Reliability in content analysis: some common misconceptions and recommendations. *Human Communication Research*, *30*, 411–433.

Kuhn, D. (1991). *The skills of argument*. Cambridge: Cambridge University Press.

Kumar, R., Rosé, C. P., Wang, Y. C., Joshi, M., & Robinson, A. (2007). Tutorial dialogue as adaptive collaborative learning support. *Proceedings of the 13th International Conference on Artificial Intelligence in Education (AI-ED 2007)*. Amsterdam: IOS Press.

Kupiec, J., Pederson, J., & Chen, F. (1995). A trainable document summarizer, Proceedings of ACM SIG-IR 1995.

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

Laham, D. (2000). *Automated content assessment of text using latent semantic analysis to simulate human cognition*. PhD dissertation, University of Colorado, Boulder.

Landauer, T., & Dumais, S. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*, 211–240.

Leitão, S. (2000). The potential of argument in knowledge building. *Human Development*, *43*, 332–360.

Lewis, D., Yang, Y., Rose, T., & Li, F. (2004). RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, *5*, 361–397.

Litman, D., Rosé, C. P., Forbes-Riley, K., Silliman, S., & VanLehn, K. (2006). Spoken versus typed human and computer dialogue tutoring. *International Journal of Artificial Intelligence in Education Special Issue on the Best of ITS'04*, *16*, 145–170.

Luckin, R. (2002). Between the lines: Documenting the multiple dimensions of computer-supported collaboration. *Computers and Education*, *41*, 379–396.

O'Donnell, A. M., & Dansereau, D. F. (1992). Scripted cooperation in student dyads: A method for analyzing and enhancing academic learning and performance. In R. Hertz-Lazarowitz, & N. Miller (Eds.) *Interaction in cooperative groups. The theoretical anatomy of group learning* (pp. 120–141). Cambridge, MA: Cambridge University Press.

Page, E. B. (1968). The use of the computer in analyzing student essays. *International Review of Education*, *14*, 210–225.

Page, E. B., & Petersen, N. S. (1995). The computer moves into essay grading: Updating the ancient test. *Phi Delta Kappan*, *76*, 561–565.

Pennebaker, J. W. (2003). The social, linguistic, and health consequences of emotional disclosure. In J. Suls, & K. A. Wallston (Eds.) *Social psychological foundations of health and illness* (pp. 288–313). Malden, MA: Blackwell.

Pennebaker, J. W., & Francis, M. E. (1996). Cognitive, emotional, and language processes in disclosure. *Cognition and Emotion*, *10*, 601–626.

Pennebaker, J. W., Francis, M. E., & Booth, R. J. (2001). *Linguistic inquiry and word count: LIWC*. Mahwah, NJ: Erlbaum.

Piaget, J. (1985). *The equilibrium of cognitive structures: The central problem of intellectual development*. Chicago: Chicago University Press.

Roman, N., Piwek, P., & Carvalho, A. (2006). Politeness and bias in dialogue summarization: Two exploratory studies, in J. Shanahan, Y. Qu, & J. Wiebe (Eds.) Computing attitude and affect in text: Theory and Applications, the Information Retrieval Series. Dordrecht: Springer.

Rosé, C. P. (2000). A framework for robust semantic interpretation. *Proceedings of 1st Meeting of the North American Chapter of the Association for Computational Linguistics*.

Rosé, C., Dönmez, P., Gweon, G., Knight, A., Junker, B., Cohen, W., et al. (2005). Automatic and semi-automatic skill coding with a view towards supporting on-line assessment. *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AI-ED 2005)*. Amsterdam: IOS Press.

Rosé, C. P., Gweon, G., Arguello, J., Finger, S., Smailagic, A., & Siewiorek, D. (2007). Towards and interactive assessment framework for engineering design project based learning. *Proceedings of ASME 2007 International Design Engineering Technical Conferences an d Computers and Information in Engineering Conference*.

Rosé, C. P., Jordan, P., Ringenberg, M., Siler, S., VanLehn, K., & Weinstein, A. (2001). Interactive conceptual tutoring in atlas-andes. In J. D. Moore, C. L. Redfield, & W. L. Johnson (Eds.) *Artificial Intelligence in Education: AI-ED in the wired and wireless future, Proceedings of AI-ED 2001* (pp. 256–266). Amsterdam: IOS Press.

Rosé, C., Roque, A., Bhembe, D., & VanLehn, K. (2003). A hybrid text classification approach for analysis of student essays. *Proceedings of the HLT-NAACL 03 Workshop on Educational Applications of NLP* (pp. 68–75). Morristown, NJ: Association for Computational Linguistics.

Rosé C. P., & VanLehn, K. (2005). An evaluation of a hybrid language understanding approach for robust selection of tutoring goals. *International Journal of AI in Education*, *15*(4).

Salomon, G., & Perkins, D. N. (1998). Individual and social aspects of learning. Review of Research in Education, 23, 1–4.

Schegloff, E., & Sacks, H. (1973). Opening up closings. *Semiotica*, *8*, 289–327.

Schoor, C., & Bannert, M. (2007). Motivation and processes of social co-construction of knowledge during CSCL. Poster presented at the 12th Biennial Conference EARLI 2007, Budapest.

Serafin, R., & Di Eugenio, B. (2004). *FLSA:* Extending latent semantic analysis with features for dialogue act classification. *Proceedings of the Association for Computational Linguistics*. Morristown, NJ: Association for Computational Lingusitics.

Soller, A., & Lesgold, A. (2000). Modeling the Process of Collaborative Learning. *Proceedings of the International Workshop on New Technologies in Collaborative Learning*. Japan: Awaiji–Yumebutai.

Stahl, G. (2006). Group cognition: Computer support for building collaborative knowledge. Cambridge, MA: MIT Press.

Stegmann, K., Weinberger, A., & Fischer, F. (2007). Facilitating argumentative knowledge construction with computer-supported collaboration scripts. *International Journal of Computer-Supported Collaborative Learning*, *2*(4).

Stegmann, K., Weinberger, A., Fischer, F., & Rosé, C. P. (2006). *Automatische Analyse natürlich-sprachlicher Daten aus Onlinediskussionen* [*Automatic corpus analysis of natural language data of online discussions*]. Paper presented at the 68th Tagung der Arbeitsgruppe für Empirische Pädagogische Forschung (AEPF, Working Group for Empirical Educational Research) Munich, Germany.

Stolcke, A., Ries, K., Coccaro, N., Shriberg, J., Bates, R., Jurafsku, D., et al. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, *26*(3), 39–373.

Teasley, S. D. (1997). Talking about reasoning: How important is the peer in peer collaboration? In L. B. Resnick, R. Säljö, C. Pontecorvo, & B. Burge (Eds.) *Discourse, tools and reasoning: Essays on situated cognition* (pp. 361–384). Berlin: Springer.

Tsochantaridis, I., Hofmann, T., Joachims, T., & Altun, Y. (2004). Support vector learning for interdependent and structured output spaces. *Proceedings of the International Conference on Machine Learning 2004*.

van der Pol, J., Admiraal, W., & Simons, P. R. J. (2006). The affordance of anchored discussion for the collaborative processing of academic texts. *International Journal of Computer-Supported Collaborative Learning*, *1*(3), 339–357.

VanLehn, K., Graesser, A., Jackson, G. T., Jordan, P., Olney, A., & Rosé, C. P. (2007). Natural language tutoring: A comparison of human tutors, computer tutors, and text. *Cognitive Science*, *31*(1), 3–52.

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

Voss, J. F., & Van Dyke, J. A. (2001). Argumentation in psychology. *Discourse Processes*, *32*(2 & 3), 89–111.

Wang, Y. C., Joshi, M., & Rosé, C. P. (2007). A feature based approach for leveraging context for classifying newsgroup style discussion segments. *Proceedings of the Association for Computational Linguistics*.

Wang, H. C., Rosé, C. P., Cui, Y., Chang, C. Y., Huang, C. C., & Li, T. Y. (2007b). Thinking hard together: The long and short of collaborative idea generation for scientific inquiry. *Proceedings of Computer Supported Collaborative Learning (CSCL 2007)*, New Jersey.

Webb, N. M. (1989). Peer interaction and learning in small groups. *International Journal of Educational Research*, *13*, 21–39.

Wegerif, R. (2006). A dialogic understanding of the relationship between CSCL and teaching thinking skills. *International Journal of Computer-Supported Collaborative Learning*, *1*(1), 143–157.

Weinberger, A. (2003). *Scripts for computer-supported collaborative learning. Effects of social and epistemic cooperation scripts on collaborative knowledge construction*. Ludwig-Maximilian University, Munich. Retrieved from http://edoc.ub.uni-muenchen.de/archive/00001120/01/Weinberger_Armin.pdf.

Weinberger, A., & Fischer, F. (2006). A framework to analyze argumentative knowledge construction in computer-supported collaborative learning. *Computers & Education*, *46*(1), 71–95.

Weinberger, A., Reiserer, M., Ertl, B., Fischer, F., & Mandl, H. (2005). Facilitating computer-supported collaborative learning with cooperation scripts. In R. Bromme, F. W. Hesse, & H. Spada (Eds.) *Barriers and Biases in network-based knowledge communication in groups*. Dordrecht: Kluwer.

Weiner, B. (1985). An attributional theory of achievement motivation and emotion. *Psychological Review*, *92*, 548–573.

Wiebe, J., & Riloff, E. (2005). Creating Subjective and Objective Sentence Classifiers from Unnanotated Texts, *Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2005)*, Springer LNCS, vol. 3406.

Wiebe, J., Wilson, T., Bruce, R., Bell, M., & Martin, M. (2004). Learning Subjective Language. *Computational Linguistics*, *30*(3), 277–308.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques* (2nd ed.). Elsevier: San Francisco. ISBN:(ISBN 0-12-088407-0).

Yeh, A., & Hirschman, L. (2002). Background and overview for KDD Cup 2002 task 1: Information extraction from biomedical articles. *SIGKDD Explorations*, *4*, 87–89.

Zechner, K. (2001). Automatic generation of concise summaries of spoken dialogues in unrestricted domains. Proceedings of ACM SIG-IR 2001.

Zhou, L., & Hovy, E. (2006). On the summarization of dynamically introduced information: Online discussions and blogs. In Proceedings of AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs, Stanford, CA.