*In Proceedings of AI in Education Conference (AIED 2007)*

# Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers

Sujith RAVI and Jihie KIM
*University of Southern California/Information Sciences Institute*
*4676 Admiralty Way, Marina del Rey, CA 90292 USA*

**Abstract.** On-line discussion is a popular form of web-based computer-mediated communication and is an important medium for distance education. Automatic tools for analyzing online discussions are highly desirable for better information management and assistance. This paper presents an approach for automatically profiling student interactions in on-line discussions. Using N-gram features and linear SVM, we developed "speech act" classifiers that identify the roles that individual messages play. The classifiers were used in finding messages that contain questions or answers. We then applied a set of thread analysis rules for identifying threads that may have unanswered questions and need instructor attention. We evaluated the results with three human annotators, and 70-75% of the predictions from the system were consistent with human answers.

**Keywords:** On-line discussion board, speech act, discussion assessment

## Introduction

Web-enhanced courses and distance education courses are becoming increasingly popular. Engagement in on-line discussions is an important part of student activities in distance education. Our work was motivated by lack of flexibility and poor assistance capability of existing discussion boards, and by the potential for utilizing natural language processing (NLP) techniques in discussion thread analysis. We are developing instructional tools for phpBB, a popular open-source bulletin board with good community support [1].

In this paper we present an approach for automatically classifying patterns of student interactions on discussion boards. In particular, the system identifies discussion threads that may have unanswered questions and need instructor attention. In profiling discussion threads and classifying patterns of student interactions, we adopt the theory of Speech Acts proposed by (Austin, 1962 [2]; Searle, 1969 [3]) and define a set of speech acts (SAs) that relate messages in discussion threads. That is, discussion threads are viewed as a special case of human conversation, and each message is classified with respect to previous messages in the same thread as a *question*, *answer*, *elaboration* or/and *correction*. We use word sequence features and SVM (Support Vector Machine) algorithms [4] in developing automatic *SA classifiers*. We have developed two speech act classifiers: question classifier and answer classifier. The question classifier identifies messages that play a role of asking questions and the answer classifier detects messages with answers in response to a previous message. Given the classification of individual messages, the *thread profiler* then applies a set of thread analysis rules for finding threads that have questions without corresponding answers. Such threads may need more attention from the instructor since students may have unanswered questions.

We applied these techniques in classifying discussion corpus from two undergraduate computer science courses on operating systems. We found that development of classifiers is very challenging because, unlike common corpora that are often used by NLP systems, discussions among undergraduate students are highly unstructured and noisy. Cleaning and preprocessing raw data, transforming them into more coherent data sets, and selecting useful features among many potential feature candidates were most challenging. The current SA classifiers that we have developed for questions and answers have accuracies of 88% and 73% respectively. The thread profiler uses the classification results in identifying the threads that may have unanswered questions. We evaluated the results with three human annotators, and 70-75% of the answers from the system were consistent with human answers.

In the next section we introduce the speech act categories that we have developed based on an analysis of student discussions. We then provide the details of the speech act classifiers including the steps for extracting features that are appropriate for classifying student discussions. The following section presents the thread profiler and shows how the SA classifiers are used in analyzing student interactions in discussion threads. Finally we present a summary and future work.

## 1. Modeling Threaded Discussions with Speech Acts

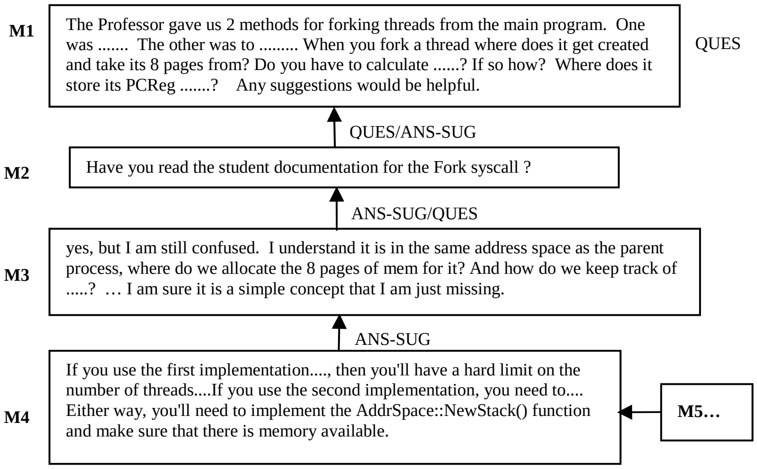
**Table 1.** Speech Act Categories for Individual Messages

| Speech Act | Description | % |
|---|---|---|
| ACK-SUP-COMP | An acknowledgement, compliment or support in response to a previous message | 8.5 |
| INFORM | Information, Command or Announcement | 6.7 |
| ANS-SUG | A simple or complex answer to a previous question. Suggestion or advice | 37.8 |
| CORR-OBJ | A correction or objection (or complaint) to/on a previous message | 9.7 |
| ELAB | An elaboration (of a previous message) or description, including elaboration of a question or an answer | 8.1 |
| QUES | A question about a problem, including question about a previous message | 29.2 |

Our discussion dataset is from an undergraduate Operating Systems course in Computer Science at the University of Southern California. The course is held every semester and the students have used the discussion board for the past five semesters. Our work focuses on the discussion corpus from the two most recent semesters. The corpus contains 475 threads with 1834 messages from 133 participants.

Unlike in a flat document set, in a threaded discussion each message plays a different role. For example, people may make arguments, support or object to points, or give suggestions. However, unlike prototypical collaborative argumentation where a limited number of members take part in the conversation with a strong focus on solving specific problems, online discussions have much looser conversational structure, possibly involving multiple anonymous discussants.

We have defined a set of speech acts (SAs) that relate pair of messages in discussion threads. A pair of messages may be labeled with more than one speech act. For example, the reply message could correct the original message as well as provide an answer. A message can have SAs with respect to more than one message. Table 1 provides a definition of each speech act. We have explored different sets of speech acts based on assessment of human annotators and found that these categories are less confusing than other finer or coarser grained categories [5, 6]. In our corpus, questions (29.2%) and answers (37.8%) comprised the biggest portion of the corpus. This is consistent with the use of the discussion board as a technical question and answer platform for class projects. Figure 1 shows an example of a discussion thread with a sequence of question and answer exchanges.



M1: The Professor gave us 2 methods for forking threads from the main program. One was ....... The other was to ......... When you fork a thread where does it get created and take its 8 pages from? Do you have to calculate ......? If so how? Where does it store its PCReg .......? Any suggestions would be helpful.

M2: Have you read the student documentation for the Fork syscall ?

M3: yes, but I am still confused. I understand it is in the same address space as the parent process, where do we allocate the 8 pages of mem for it? And how do we keep track of .....? … I am sure it is a simple concept that I am just missing.

M4: If you use the first implementation...., then you'll have a hard limit on the number of threads....If you use the second implementation, you need to.... Either way, you'll need to implement the AddrSpace::NewStack() function and make sure that there is memory available.

**Figure 1:** Example of a student discussion thread

## 2. Speech Act Classifiers: Identifying the Roles that Individual Messages Play

Our current work focuses on two SA classifiers: the question classifier (QC) and the answer classifier (AC). This section describes the details of the two classifiers.

### *2.1 Handling Incoherent and Noisy Discussion Data*

We found that discussion data from undergraduate students are highly incoherent and noisy. The raw data includes humors and personal announcements as well as technical text. Student messages are very informal and there are high variances in the way they present similar information. A lot of messages on programming assignment include programming code. Besides typical data preprocessing and cleaning steps taken by many NLP systems, such as stemming and filtering, our system performs additional steps for removing noise and reducing variances.

We first remove the text from previous messages that is automatically inserted by the discussion board system when the user clicks on a *"Reply to"* button. We also apply a simple stemming algorithm that removes *"s"* and *"es"* for plurals. For discussions on programming assignment, the discussion included programming code fragments. Each section of programming code or code fragment is replaced with a single term called `code`. We then use a transformation algorithm that replaces common words or word sequences with special category names. For example, many pronouns like *“I”*, *“we”*and *“you”* are replaced by the symbol `categ_person` and sequences of numbers by `categ_number_seq`. For words like *“which”*, *”where”*, *”when”*, *”who” and ”how”*, we used the term `categ_wh`. Similar substitution patterns were used for a number of categories like filetype extensions (*“.html”*, *“.c”*, *“.c++”*, *“.doc”*), URL links and others. Students also tend to use informal words (eg: *“ya”*, *“yeah”*, *“yup”*) and typographical symbols such as smiley faces as acknowledgement, support or compliment. We transform such words into consistent words or symbols. We also substitute words like *‘re*, *‘m*, *‘ve*, *don’t*, etc. with *“are”*, *“am”*, *“have”*, *“do not”*, etc. Finally, since speech act tends to rely more on surface word patterns rather than technical terms used, technical terms occurring in the messages were replaced by a single word `tech_term`.

### *2.2 Feature Selection*

For our classifiers, we use N gram features that represent all possible sequences of N terms. That is, unigrams (single word features), bigrams (sequence of 2 words), trigrams (sequence of 3 words) and quadrograms (4 word sequences) are used for training and building the classifiers. There were around 5,000 unigrams or unique words occurring in the training corpus. Since the data was very noisy and incoherent, the feature space (where each feature could be a word or word sequence) is very large.

**Table 2.** Top N-grams based on Information Gain

| Categ. | 1-gram | 2-grams | 3-grams | 4-grams |
|---|---|---|---|---|
| QC | ?<br>[categ_wh]<br>will<br>do<br>confused | do [categ_person]<br>[tech_term] ?<br>can [categ_person]<br>is there<br>? thanks | [categ_wh] should [categ_person]<br>[categ_person] was wondering<br>[or/and] do [categ_person]<br>is there a [tech_term] ? | do [categ_person] have to<br>do [categ_person] need to<br>[tech_term] [tech_term] [tech..] ?<br>is there a better<br>does this mean that |
| AC | yes<br>am<br>helps<br>but<br>depends | look at<br>[or/and] do<br>seems like<br>in [tech_term]<br>stated in | look at the<br>for example ,<br>. [categ_person] should<br>let [me/him/her/us] know<br>not seem to | [categ_person] am a [tech_term]<br>do [categ_person] have to<br>look at the [tech_term]<br>in the same [tech_term] |

We use Information Gain theory [7] for pruning the feature space and selecting features. Information gain value for a particular feature gives a measure of the information gained in classification prediction, i.e. how much the absence or the presence of the feature may affect the classification. First, we compute the information gain values for different N gram features extracted from the training data. For each feature, we compute two information gain values, one for QC and the other for AC. Subsequently, all the features (1-gram, 2-grams, 3-grams, and 4-grams) are sorted using the information gain. We use the top 200 features for each classifier. Some of the top N gram features for QC and AC are shown in Table 2.

### *2.3 SA Classifiers with SVM*

The training set consisted of 1010 messages. A half of them were from one semester and the other half from the other semester. The test set had 824 messages. All the threads were annotated by hand beforehand. The feature sets for training are extracted from the training corpus, as described above.

We use a linear SVM implementation [4] to construct SA classifiers. Linear SVM is an efficient machine learning technique used often in text classification and categorization problems. We constructed feature vectors for all the messages in the corpus. A feature vector of a message consisted of a list of values for individual features that represent whether the features existed in the message or not. We perform 5-fold cross validation experiments on the training data to set kernel parameter values for the linear SVM. After we ran SVM, the resulting classifiers (QC and AC) were then used to predict the SAs for the feature vectors in the test set. QC tells whether a particular message contains question content or not, and AC predicts whether a message contains answer content, i.e. answers or suggestions. The classification was then compared with the human annotations.

The resulting QC and AC had accuracies of 88% and 73% respectively. A similar pattern was observed among human annotators. The agreement ratio between the human annotators for questions (Agreement = 94.89%, kappa = 0.89) is higher than the one for answers (Agreement = 86.13%, kappa = 0.72) [8]. This could be because of higher degree of variances in the messages that contain answers or suggestions. Some of the answers and suggestions also take the form of questions (e.g. Message M2 in Figure 1), corrections, etc. making it difficult to classify the message.

## 3. Assessing When Interventions Are Needed: Thread Profiling with SA Classifiers

In order to assess discussion threads with SA classification results, we have developed a set of thread analysis rules. Our current analysis focuses on handling typical question answer patterns from students, as shown in Figure 2. That is, whether the given thread contains questions, and if it does, whether the questions were answered or not. The questions we formulated for thread assessment are the following:

**(Q1) Was there at least one question in the thread? (Y/N)**
**(Q2) Was the first question in the thread answered? (Y/N)**
**(Q3) Were all the questions answered? (Y/N)**
**(Q4) Were some of the questions answered? (Y/N)**

The thread analysis rules for answering these questions are shown in Table 3.

We randomly picked 55 threads from our test corpus and automatically classified each thread with our speech act classifiers and thread analysis rules. The threads that do not contain technical discussions such as humorous messages or personal information exchanges were not included. For each thread in the test corpus, we asked three humans to assess the thread and answer the above four assessment questions.

Some of the typical thread patterns that exist in the corpus are illustrated in Figure 2. The pattern (a) shows a thread that does not contain any question. Typically such a thread would contain only announcement messages. Many threads follow pattern (b) where the first message M1 is a question and subsequent messages M2 and M3 contain answers to this question. Pattern (c) has a Question-Answer-Question-Answer sequence, where all the questions in the thread are answered. Finally, the thread (d) illustrates the case of a *"hanging questions"*, where some of the questions are not answered. For example a student may try to understand the answer and ask a follow-up question (in M3) but does not receive any reply, leaving his/ her question unanswered.

**Table 3.** Thread Classification Rules

| Rule | Description |
|---|---|
| 1 | - If QC = yes for any message in thread, answer "YES", else answer "NO" for question Q1. |
| 2 | - Find the first message $m_i$ in the thread for which QC = yes<br>- Find all messages $m_j$ which are replies to $m_i$,<br>- If AC = yes for any of the messages $m_j$, answer "YES" for question Q2, else answer "NO". |
| 3 | - Find the set of all messages $M_q$ in the thread for which QC = yes<br>- For every message $m_i$ in the set $M_q$, find all the reply messages $m_j$<br>- If all the messages $m_i$ in $M_q$ have at least one reply message $m_j$ with AC=yes, then answer "YES" for question Q3, else answer "NO".<br>- If some (at least one) of the messages $m_i$ in $M_q$ have a reply message $m_j$ with AC=yes, then answer "YES" for question Q4, else answer "NO". |

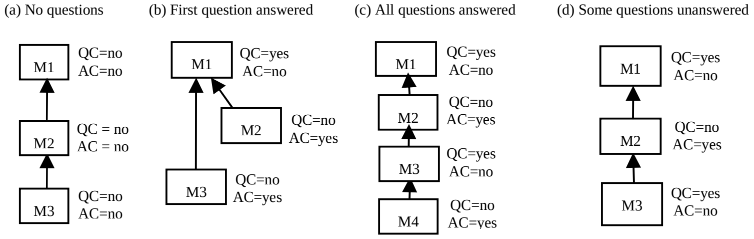

(a) No questions (b) First question answered (c) All questions answered (d) Some questions unanswered

**Figure 2**: Example Question-Answer patterns

The answers from the system were compared with human answers. The results (percentage of agreement) from the comparison are shown in Table 4. For Q1, the average agreement among humans is about the same as the one between humans and the system. Since QC provides a high accuracy, it supports Q1 very well. For Q2--Q4, 70-75% of the answers generated from the system were consistent with human answers. We believe that a lower system performance in the latter case could be due to lower classification accuracy returned by the AC. For example, some messages contain answers in the form of questions, and some of the answers were presented in an indirect and informal manner spanning long sentences, which made the classification using word N-gram features harder.

**Table 4.** Thread Classification Agreement between humans and the auto-classifier
(H = Human Annotators, M = automatic classifier system)

| Assessment Question | Average agreement among humans | Average agreement between M and H |
|---|---|---|
| Q1 | 92.7% | 92.7% |
| Q2 | 95.1% | 74.5% |
| Q3 | 85.3% | 70% |
| Q4 | 96.5% | 75.8% |

The results from this automatic thread profiling can potentially be used for ranking discussion threads. The threads with positive/Yes values for Q1 and negative/No values for Q2, Q3 and Q4 can be reported as the ones that may need instructor attention.

## 4. Related Work

There have been many attempts to assess collaborative activities. Various approaches of computer supported collaborative argumentation have been proposed. Machine learning techniques have been applied to train software to recognize when students have trouble sharing knowledge in collaborative interactions [9].

Rhetorical Structure Theory [10] based discourse processing has attracted much attention with successful applications in sentence compression and summarization. Most of the current work focuses on sentence-level text organization [11] or the intermediate step [12]. Analyzing and utilizing discourse information at a higher level, e.g., at the paragraph level, still remains a challenge to the natural language community. In our work, we utilize the discourse information at a message level.

There has been prior work on dialogue act analysis and associated surface cue words [13, 14]. There have also been previous approaches like modeling Dialogue Acts for automatic tagging and recognition of conversational speech [15] and related work in corpus linguistics where machine learning techniques have been used to find conversational patterns in spoken transcripts of dialogue corpus [16]. Although they are closely related to our speech act analysis, it is hard to directly map the existing results to our analysis. The interactions in our corpus are driven by problems or questions initiated by students and often very incoherent. Carvalho and Cohen (2005) [17, 18] presented a dependency-network based collective classification method to classify email speech acts. Due to high variances and incoherence of student discussions, we face more challenges in data processing and feature selection.

There have been efforts to analyze interactions in on-line communities. For example, Talk-to-me [19] can predict the likelihood of a message receiving a reply based on the content of the message and the message sender. Our work provides complementary capability by providing SA-based interaction analysis capabilities. Also our analysis work is driven by requirements from instructors and students rather than need of general on-line communities seeking information.

Graph-based algorithms have been used in text mining, clustering, and other similar problems [20]. They could potentially be used to profile or find patterns in discussion threads, where threads could be represented by graphs and messages within a thread could represent nodes in the graph.

## 5. Summary and Future Work

This paper presents an approach for profiling student interactions in on-line collaborative discussions. Our work focuses on technical discussions among undergraduate students. In profiling discussion threads, we adopted the Speech Act theory and developed speech act classifiers using N-gram features and SVM algorithms. A set of thread analysis rules are used in assessing whether the given discussion thread has unanswered questions.

We found that automatic profiling of undergraduate student discussions is very challenging due to high incoherence and noise in the data. Especially messages that contain long sentences, informal statements with uncommon words, answers in form of question, are difficult to classify. We are looking at additional features such as features of neighboring messages and topic features [21] as candidates for improving the performance of the system. Our informal interview with human annotators indicates that features from previous messages may provide useful hints for the classifier. For example, if the previous message was a question, 84% of the reply messages present an answer [5]. We are currently exploring possible extensions to our classifiers based on these ideas. We are also working on classifiers for the remaining SA categories such as objection or support so that we can analyze various types of student interactions.

## Acknowledgement

This work is supported by the NSF, CCLI-Phase 2 grant (Award No. 0618859). We would like to thank Bernadette Aurelio and Cheng-Wei Lin for their data annotation work. We also thank Erin Shaw for her comments on an earlier draft.

## References

[1] Feng, D., Shaw, E., Kim, J., and Hovy, E.H., 2006. An Intelligent Discussion-bot for answering student queries in threaded discussions. In Proceedings of Intelligent User Interface Conference, pp. 171-177

[2] Austin, J., 1962. How to do things with words. Cambridge, Massachusetts: Harvard Univ. Press.

[3] Searle, J., 1969. Speech Acts. Cambridge: Cambridge Univ. Press.

[4] Chang, C.-C. and C.-J. Lin, 2001. LIBSVM: a library for support vector machines.

[5] Kim, J., Chern, G., Feng, D., Shaw, E., and Hovy, E., 2006. Mining and Assessing Discussions on the Web through Speech Act Analysis, Proceedings of the ISWC'06 Workshop on Web Content Mining with Human Language Technologies.

[6] Kim, J. and Beal, C., 2006. Turning quantity into quality: Supporting automatic assessment of on-line discussion contributions, American Educational Research Association (AERA) Annual Meeting.

[7] Yang, Y., Pedersen, J.O., 1997. A Comparative Study on Feature Selection in Text Categorization, Proc. Of the 14th International Conference on Machine Learning, pp.412---420.

[8] Cohen, J., 1960. A coefficient of agreement for nominal scales, Educational and Psychological Measurement, 20, 37-46.

[9] Kolodner, J. L., & Nagel, K., 1999. The Design Discussion Area: A collaborative learning tool in support of learning from problem solving and design activities. Proceedings of CSCL'99, pp. 300-307.

[10] Mann, W.C. and Thompson, S.A., 1988. Rhetorical structure theory: towards a functional theory of text organization, Text: An Interdisciplinary Journal for the Study of Text, 8 (3), pp. 243-281.

[11] Soricut, R. and Marcu, D., 2003. Sentence level discourse parsing using syntactic and lexical information. In Proceedings of Human Language Technology conference - NAACL.

[12] Sporleder, C. and Lapata, M., 2005. Discourse chunking and its application to sentence compression. In Proceedings of Human Language Technology conference - EMNLP.

[13] Samuel, K., 2000. An Investigation of Dialogue Act Tagging using Transformation-Based Learning, PhD Thesis, University of Delaware.

[14] Hirschberg, J. and Litman, D., 1993 Empirical Studies on the Disambiguation of Cue Phrases, Computational Linguistics, 19 (3).

[15] Stolcke, A. , Coccaro, N. , Bates, R. , Taylor, P. , Van Ess-Dykema, C. , Ries, K., Shriberg, E. , Jurafsky, D. , Martin, R. , Meteer, M., 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. Computational Linguistics, v.26 n.3, p.339-373.

[16] Shawar, B. A.; Atwell, E. 2005. Using corpora in machine-learning chatbot systems. International Journal of Corpus Linguistics, vol. 10, pp. 489-516.

[17] Carvalho, V.R. and Cohen, W.W., 2005. On the collective classification of email speech acts. In Proceedings of SIGIR, pp. 345-352.

[18] Carvalho V., and Cohen, W., 2006. Improving Email Speech Act Analysis via N-gram Selection, HLT-NAACL, ACTS Workshop

[19] Arguello, J., Butler, B. S., Joyce, L., Kraut, R., Ling, K. S., & Wang, X., 2006. Talk to me: Foundations for successful individual-group interactions in online communities. In Proceedings of the ACM Conference on Human Factors in Computing Systems.

[20] Mollá, D., 2006. Learning of Graph-based Question Answering Rules. In Proceedings HLT/NAACL Workshop on Graph Algorithms for Natural Language Processing, 37-44.

[21] Feng, D., Kim, J., Shaw, E., and Hovy E., 2006, Towards Modeling Threaded Discussions through Ontology-based Analysis, In Proceedings of National Conference on Artificial Intelligence.

[22] Soller, A., and Lesgold, A., 2003. Computational Approach to Analyzing Online Knowledge Sharing Interaction, In Proceedings of Artificial Intelligence in Education.