

# An Intelligent Discussion-Bot for Answering Student Queries in Threaded Discussions

Donghui Feng, Erin Shaw, Jihie Kim, Eduard Hovy  
Information Sciences Institute  
University of Southern California  
4676 Admiralty Way  
Marina del Rey, CA, 90292 USA  
{donghui, shaw, jihie, hovy}@isi.edu

## ABSTRACT

This paper describes a discussion-bot that provides answers to students' discussion board questions in an unobtrusive and human-like way. Using information retrieval and natural language processing techniques, the discussion-bot identifies the questioner's interest, mines suitable answers from an annotated corpus of 1236 archived threaded discussions and 279 course documents and chooses an appropriate response. A novel modeling approach was designed for the analysis of archived threaded discussions to facilitate answer extraction. We compare a self-out and an all-in evaluation of the mined answers. The results show that the discussion-bot can begin to meet students' learning requests. We discuss directions that might be taken to increase the effectiveness of the question matching and answer extraction algorithms. The research takes place in the context of an undergraduate computer science course.

## Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces: *Theory and methods, Natural language.* H.1.2 [Models and Principles]: User/Machine Systems: *Human information processing*

## General Terms

Algorithms, Design, Experimentation, Human Factors.

**Keywords:** Natural language processing, discussion-bot, threaded discussion, online learning environment.

## 1. INTRODUCTION

Online learning is ubiquitous and intelligent computer mediated communication (CMC) interfaces play an increasingly important role in the acquisition of knowledge during learning. Discussion boards, also referred to as bulletin boards and conferencing systems, are integral components of most online learning environments, and used in both on-campus and remote courses to facilitate instructor-student and student-student communication. The Distant Education

Network (DEN) at the University of Southern California's Viterbi School of Engineering broadcasts on-campus courses to remote students via microwave and internet [20]. DEN provides its own course management system (CMS) to participants of these courses. In place of the native discussion board of the CMS, several courses use an instrumented open source discussion board based on phpBB [15], known as the ISI Discussion Board (ISI DB). The ISI DB, shown in Figure 1, was developed as a platform for developing and evaluating new teaching and learning technologies. This paper describes our efforts to develop new information retrieval (IR) and natural language processing (NLP) techniques for the educational domain.

This research takes place in the context of an undergraduate computer science course on operating systems. The course is offered every semester and is fairly consistent with respect to the topics discussed and projects required each term. Students are encouraged to participate in discussions on theoretical or practical problems during their studies and generally use the discussion board to seek answers from instructors or peers. Their questions are often time critical, and responding to multiple posts in a timely manner can be a challenge for instructors and assistants. It follows that an intelligent agent able to monitor the discussions, identify query interests, and select matching answers from previous discussions and course materials would be desirable, especially in cases where similar questions are asked repeatedly, either per or across semesters, or when answers to questions have already been provided in available course documents. However, while we wish to relieve the burden of instructors and assistants, we do not wish to turn the discussion board into a search engine, which we believe would discourage interchanges; thus, it is vital to provide an effective yet unobtrusive agent interface.

In this paper, we describe an intelligent agent, or discussion-bot, that has been implemented within the ISI Discussion Board, which automatically answers student questions. Using information retrieval and natural language processing techniques, the discussion-bot identifies users' interests, mines suitable answers from an annotated corpus and displays an appropriate response. A novel approach to modeling threaded discussions is used to analyze past discussions to facilitate answer extraction.

The rest of the paper is organized as follows: We describe related work in Section 2, and the processing of archived data in Section 3. The details of the discussion-bot system are described in Section 4, including the system architecture and the interest-matching and answer generation algorithms.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*IUI'06*, January 29–February 1, 2006, Sydney, Australia.  
Copyright 2006 ACM 1-59593-287-9/06/0001...\$5.00.

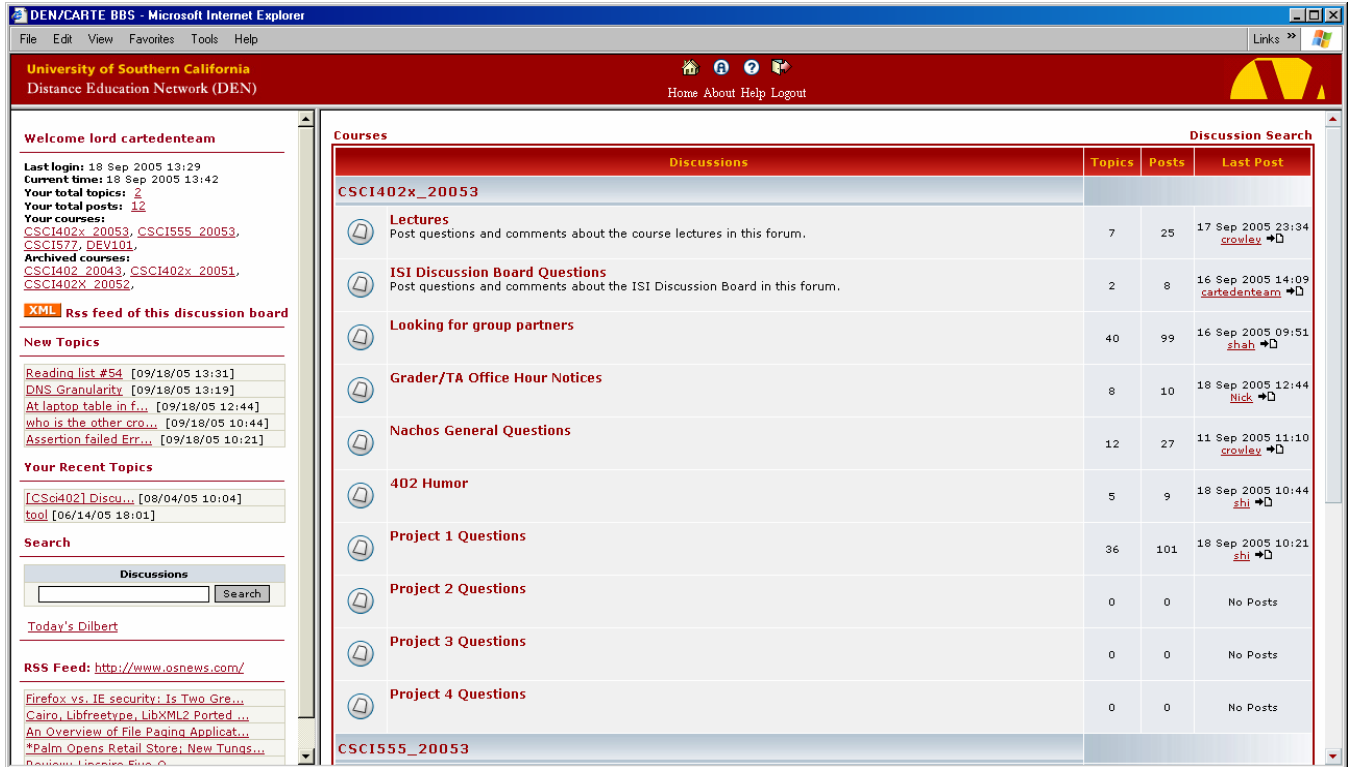


Figure 1. USC/DEN ISI DB online learning environment

Experimental results are presented in Section 5. We conclude in Section 6 with discussions on related issues and future work.

## 2. RELATED WORK

Research in natural language understanding and information retrieval is not new [17], but automatic question answering in an open domain is still an unsolved problem. Much of the current research in the natural language community focuses on developing natural language interfaces for understanding and answering user queries. Successful question answering systems tend to have similar underlying pipeline structures [e.g. 7, 8, 11, 14, 16, 22]. The general framework involves parsing questions, searching documents, and pinpointing answers. Most research focuses on factoid questions that can be answered with a short phrase (e.g., ‘Who is the Prime Minister of Australia?’). This makes it feasible to classify the answer type and target when a question is parsed. Typical queries to a discussion board domain are more complex. Questions often run over multiple lines and include lengthy contextual or procedural descriptions; often, the text is incoherent. This special characteristic makes it almost impossible to represent and identify answer types in students’ posts, and also more difficult to computationally discern true questions and answers in student messages.

The artificial intelligence community has developed various types of chat-bot to simulate a human interaction [1], however, most of them focus on dialogue via single sentences or single phrases only. They generate answers based only on the immediate past input from the user and have limited ability to learn from a corpus. Complex or contextual questions are difficult for a typical chat-bot to converse about.

Related work on dialogue modeling investigates different ways to manage continuous dialogue for personal assistants [e.g. 13] or scaffold conversations in a virtual meeting space [9, 12]. In contrast, while we focus on optimizing a one-step question response by mining knowledge from archived materials asynchronously. More similarly, the work in [10] generates help-desk responses using clustering techniques, but their corpus is composed of two-party, two-turn, conversation pairs, which makes it easier to bypass the complex analysis of discussions among multiple parties.

In the area of online learning, much attention has been paid to the analysis of student learning behaviors in online communications. Various frameworks have been proposed for characterizing and analyzing computer mediated communication in the context of collaborative discussions [18], knowledge sharing [19], email and chat exchanges [4, 10], and general argumentation [2, 3], but none are sufficiently relevant or fine-grained to facilitate data mining and answer extraction in threaded discussions.

## 3. PROCESSING COURSE MATERIALS AND ARCHIVED DISCUSSIONS

For the operating systems course, we had two resources available for data mining suitable answers to student queries: the supplementary course documents and threaded discussions from past semesters. Table 1 gives the numbers of message threads and individual posts, and the number of documents.

**Table 1. Numbers of threads and documents**

Msg. Threads	Msg. Posts	Documents	Doc. Tiles	Avg. Tiles per Doc.
1236	3093	279	2826	10.13

### 3.1 Document Processing

Course documents include reading assignments, homework and solutions, project descriptions, and instructions. Instead of matching a whole document, which is common in regular keyword searches, we aim to match and retrieve only part of a document. We applied a natural language processing tool, TextTiling [6], to segment every whole document into semantically-related segments (tiles) and subsequently process tile units (versus document units). As shown in Table 1, the total number of document tiles is 2826 and the average number of tiles per document is 10.13.

### 3.2 Modeling Threaded Discussions

Archived threaded discussions from previous semesters are also included in the corpus. A threaded discussion includes an initial message together with all messages posted in response to it. All responses are sequentially linked to the original message in chronological order. Participants can read or respond to any of the messages in a thread. We used the last three semesters of discussions, resulting in 1236 threads.

**Table 2. Definitions of post speech acts**

Code	Name	Description
QUES	Question	Question on specific problems
COMM	Command	Command or announcement
DESC	Describe	Describe a fact or situation
CANS	Complex Answer	An answer requiring a full description of procedures, reasons, etc.
SANS	Simple Answer	An answer with short phrase or words, e.g. factoid, Yes/No
ELAB	Elaborate	Elaborate on a previous arguments or questions
CORR	Correct	Correct a wrong answer/solution with new one
OBJ	Object	Object to some argument/suggestions/solutions
SUG	Suggest	Give advices/suggestions for some problems/solutions
SUP	Support	Support others' arguments/solutions
ACK	Acknowledge	Confirm or acknowledgement

Unlike in a flat document set, in a threaded discussion, each post plays a different role in the discussion. For example, people may make arguments, support or object to points, or give suggestions. Some previous work studied collaborative argumentation

structure [3] to capture design rationale [e.g. 2] or to help the student develop augmentation skills [5]. However, unlike the scenarios in 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 structure in conversation possibly with multiple anonymous people involved.

To better extract useful information from the discussion, we defined a set of post speech acts based on [21]. Every message in the corpus was manually analyzed and assigned to a speech act category based on its role in the thread. Table 2 gives the specific definitions of each type of message post speech act.

## 4. THE DISCUSSION-BOT: GENERATING RESPONSES TO STUDENT QUERIES

Instructors, students, teaching assistants, and graders all play different roles on the discussion board. Students typically ask administrative or technical questions about assignments, while teachers typically answer questions and post announcements. Students' questions are often time critical, and responding to multiple posts in a timely manner can be a challenge for instructors and assistants. As a virtual agent, the discussion-bot will mine answers and display them automatically.



**Figure 2. A student query and its BB Bot response.**

The discussion-bot runs as a background server, monitoring new posts to the discussion board. We have chosen to process first-messages only, i.e., messages posted when a student starts a new thread. In an analysis of 640 threads, we found that almost 80% of these first-messages were queries of some sort, and over 60% of the replies to the queries were answers or suggestions. Though questions can arise mid-thread, lengthy contexts and complex descriptions make it difficult to computationally discern an exact question. Furthermore, we wish the response interface to be subtle and want to avoid stifling interchanges with a 'search engine' response effect. Figure 2 shows a student query and a matched response from the discussion archives. The response also includes a hyperlink to the discussion thread from which the answer was extracted.

## 4.1 System Architecture

We have implemented the discussion-bot within the ISI discussion board. Figure 3 depicts the system architecture. When a query is posted to the discussion board, the discussion-bot system extracts features from the post first, e.g., the words and word frequencies. Following that, the system tries to match the student's interest in all archived data, both course documents and past discussions. A list of document tiles or message posts related to the question is ranked based on predefined metrics. The answer extraction module processes the top 1 candidate in the list based on whether it is a segment of a document or a post. Different strategies are then applied to generate the answer which is automatically presented on the discussion board.

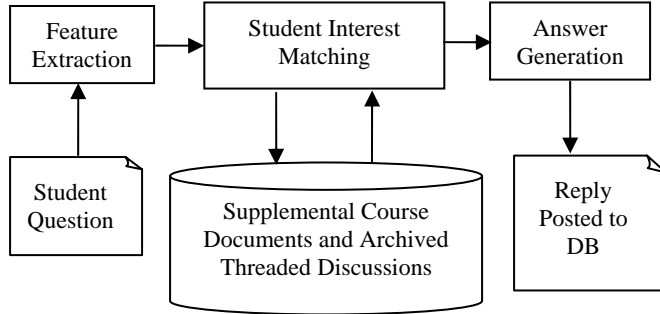


Figure 3. System architecture

## 4.2 Matching Students' Interests

After a question is posted, the system determines which document or message best matches the student's interest based on features extracted from the post.

Traditional question answering systems in the natural language community usually apply a question-processing module to determine an answer type and extract query terms for the search engine [e.g. 7, 8, 11, 14]. The process is not feasible in our case because it is difficult to discern an exact question and thus identify a single answer type with enumerated query terms. Instead, we retrieve a set of semantically-related passages that match a student's interest by directly computing the cosine similarity between question post and archived data using the TF\*IDF technique [17]. Here a passage refers to a document tile or a message post.

Intuitively, document tiles and posts with similar words are more likely to be semantically-related. This information is represented by term frequency (TF). However, those with more general terms may be unintentionally biased when only TF is considered, so inverse document frequency (IDF) is introduced to fix the bias. This results in a more general term appearing in more data units with a smaller weight.

Mathematically, suppose we have a total of  $n$  passages in the corpus, a student query  $q$ , and corpus passages  $p_1, p_2, \dots, p_n$ . Also suppose there are a total of  $m$  different unique words,  $w_1, w_2, \dots, w_m$ , found in all documents and posts. Let the number of occurrences of word  $w_j$  in passage  $p_i$  be  $tf_{ij}$  and the number of passages in which word  $w_j$  is found  $c_j$ . The

student's post and each passage in the corpus can be represented with a vector in the following format:

$$q = \langle w_{q1}, w_{q2}, \dots, w_{qm} \rangle$$

$$p_i = \langle w_{p_i1}, w_{p_i2}, \dots, w_{p_im} \rangle$$

where  $m$  is the total number of words in this domain, and  $w_{ij} = 0$  if a word is missing in that passage. Each element then, after normalization in the vectors, can be computed by

$$w_{ij} = \frac{tf_{ij} \log(n/c_j)}{\sqrt{\sum_{j=1}^m (tf_{ij})^2 [\log(N/c_j)]^2}}$$

When a question is posted, we can retrieve a list of semantically-related passages (post or document tile) using the cosine similarities between the query post and the passages using

$$\cos\_sim(q, p_i) = \frac{\sum_{j=1}^m w_{qj} * w_{p_ij}}{\sqrt{\sum_{j=1}^m (w_{qj})^2 * \sum_{j=1}^m (w_{p_ij})^2}}$$

## 4.3 Answer Generation

The resultant list of passages is ranked in the descending order of the cosine similarities. Typically, we take the top 1 result to generate the reply post. Depending on the type of the passage, we generate the answers in different ways. If the passage is a document tile from a course document as described in Section 3, we take all the tile text as the answer and include a reference link to the original whole document. If the passage is a message post, we apply an extraction procedure to generate the answer based on an analysis of the threaded discussions using the post speech acts defined in the previous section. Typically, we take the post that appears in the top position and search through the thread it belongs to, applying our forward-backward answer generation algorithm.

To delimit the extent of the answer material harvesting, we argue that the extracted posts should be limited to posts discussing only single questions, and therefore assign posts annotated with "QUES", "CANS/SANS" as boundary nodes for the extraction procedure. Also, any post annotated with "ACK" is viewed as a sign of the end of the discussion of one question and is included in the boundary set. The extraction procedure does not go beyond the posts defined in the boundary set. On the other hand, the relations between consecutive posts make it possible that some posts contain important information while others do not. We define relation rules to distinguish those posts. For example, a post annotated with "CORR" is actually a correction of a previous one, and will void the previous post. In this case, only the latter node will be included during the extraction. A relation of "SUP" between two posts will emphasize that both of them are correct and desired. On the contrary, a relation of "OBJ" will make the first post less useful.

We next take as the starting point the top 1 post returned by matching the student interest. The forward-backward algorithm

first traverses from the starting point the partial thread graph forward with a Depth First Search (DFS) followed by a backward traverse with DFS. If it meets the boundary test, it will stop at that level. During the DFS traversal, extraction rules as described above are applied (See Figure 4). This builds a list of extracted posts whose contents will be combined in the reply post. The reply post is finally assembled with a reference to the original source, either a full document or a whole discussion.

1. Define boundary node set and relation rule set;
2. Start from the starting post, and go forward with DFS to construct a node list;
3. Start from the starting post, and go backward with DFS to construct a node list;
4. Combine result lists from Step 2 and Step 3;
5. Generate text answers from the list in Step 4 and attach a reference to the original thread discussion.

Figure 4. Forward-backward answer generation algorithm.

## 5. EXPERIMENTS

### 5.1 Experimental Setup

For evaluation purposes, we used the operating systems course corpus, which includes 279 documents and 3093 posts. The archived discussions contain 1236 threads. We first looked at the distribution of the length of each thread, that is, how many posts were included in each thread. Figure 5 gives the distribution of the length of each thread: 524 threads (over 40%) consist of only one post while most of the threads consist of from two to ten posts. There are very few threads that contain more than 10 posts.

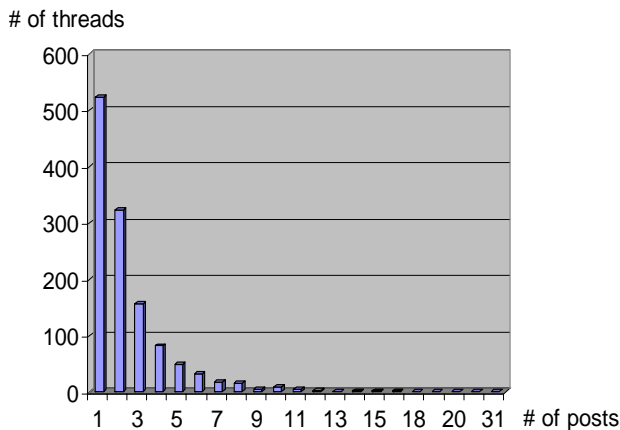


Figure 5. Statistics of thread length.

With three semesters of data it was easy to carry out a realistic experiment. We chose to evaluate the most recent semester of archived discussion data against the previous two semesters of archived data. The test set contains 66 questions, but there is no guarantee that the questions are answered or even discussed in the corpus. The testing performance for this situation was expected to be considerably worse than for the situation where the condition has been met (i.e., an answer exists).

We wrote a script to simulate a user posting new questions on the discussion board. The posting automatically triggers the discussion-bot and sends the message text on for processing. The selected reply is posted to the discussion board immediately following the original post.

### 5.2 Results and Analysis

To better search the threaded discussions and extract the most useful information, all archived posts were manually classified as one of eleven speech acts described in Section 3. Our corpus includes a total of 2173 speech acts. Table 3 gives the percentage of speech acts found in all posts of the annotated corpus.

Table 3. Statistics of post speech acts in archived discussions.

Code	Frequency	Percentage (%)
QUES	794	36.54
COMM	11	0.51
DESC	133	6.12
CANS	372	17.12
SANS	59	2.72
ELAB	149	6.86
CORR	25	1.15
OBJ	37	1.70
SUG	417	19.19
SUP	105	4.83
ACK	71	3.27

We found that questions comprised the biggest portion of the corpus. This is consistent with the use of the board as a technical question and answer platform. Correspondingly, answers (CANS and SANS) and suggestions comprise 39.03% of total posts. The reason we consider suggestions together with answers is that for some of the questions, it is difficult to give an exact answer and in most cases, the replies are presented as suggestions. The ratio of complex answers to simple answers is 6.3. This matches our expectation that students ask lengthy context and procedural questions instead of simple factoid or Yes/No questions.

We also investigated the relations between two consecutive posts. As each post is classified as a speech act, the relations are represented by the consecutive relations between post speech acts. Table 4 gives the probabilities of transitions between all speech acts. To make it easier to understand, we add “START” and “END” states that refer to the start and the end of a thread discussion, respectively. Each represents the probability of going from the previous speech act (prev\_SA in left column) to the next speech act (SA in top row). The information shows us how a discussion is conducted within a group of students. For example, there is a probability of 78.8% that any given discussion will start with a question (QUES), and a probability of 18.4% that it will start with a description of a situation (DESC).

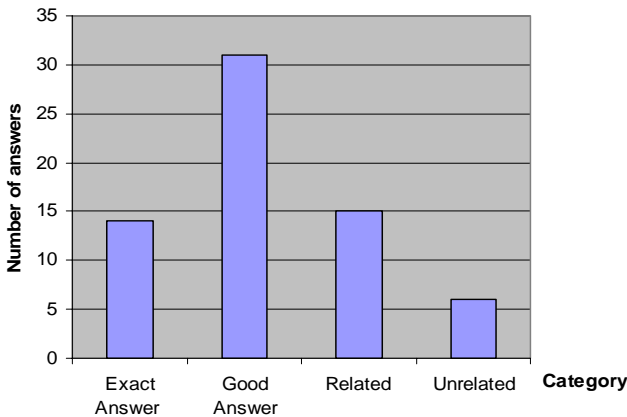
To evaluate the quality of the automatically-generated reply, human judges were asked to manually classify each of the 66 replies, as *Exact Answer*, *Good Answer*, *Related Answer*, or *Unrelated Answer*. Based on these classifications, we tested both

**Table 4. Probabilities of speech act transitions.**

P(SA prev_SA)	ACK	CANS	COMM	CORR	DESC	ELAB	OBJ	QUES	SANS	SUG	SUP	END
START	0	0	0.018	0	0.184	0	0	0.788	0	0.01	0	0
ACK	0.029	0	0	0	0.014	0.014	0	0.043	0	0	0.043	0.857
CANS	0.044	0.008	0	0.013	0.005	0.076	0.021	0.154	0	0.016	0.029	0.634
COMM	0.063	0	0	0	0	0.188	0	0.438	0	0.25	0.063	0
CORR	0.038	0	0	0.038	0.038	0	0	0.077	0	0	0.038	0.769
DESC	0.059	0.036	0	0.036	0.024	0.083	0.036	0.249	0	0.284	0.136	0.059
ELAB	0.052	0.091	0	0.006	0	0.143	0	0.117	0.013	0.071	0.013	0.494
OBJ	0	0.027	0	0.054	0.027	0	0.081	0.054	0	0.054	0.108	0.595
QUES	0.01	0.349	0	0.005	0.003	0.057	0.007	0.072	0.057	0.317	0.032	0.089
SANS	0.034	0	0	0	0	0	0.017	0.085	0	0.017	0	0.847
SUG	0.04	0.007	0	0.011	0.004	0.052	0.022	0.168	0.002	0.045	0.04	0.608
SUP	0.009	0.037	0	0	0.009	0	0.019	0.056	0	0.065	0.083	0.722

the student interest matching algorithm and the answer generation algorithm using two different strategies. The first strategy is all-in test, which exploits the whole corpus including the thread from which the question originated. This is designed mainly for testing the answer generation module. The second strategy is self-out test, which excludes the thread that the original question came from. In this test, we tried to simulate the ideal test situation with different configurations by tuning threshold values.

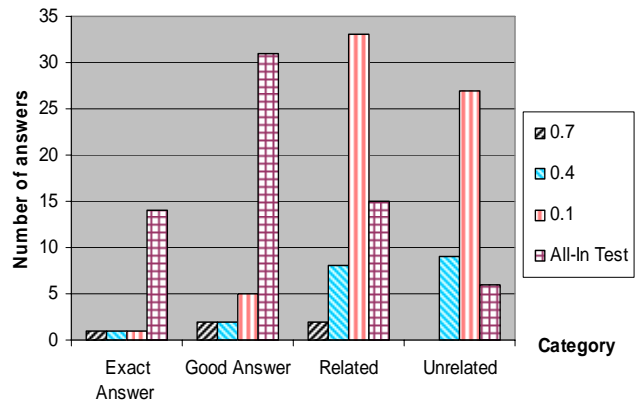
Figure 6 gives the evaluation results for the all-in test, which tests the efficiency of the answer generation algorithm given the assumption that the student’s interest is correctly matched. Even in this case, though, we are not guaranteed a response (in the case that no one replied to the question), and if there was an original response, it may not have been an exact answer (it may have been classified as a suggestion, for example.) It is also possible that the target answer was not retrieved by the extraction rules due to the complexity of post speech act analysis. The results are as expected.



**Figure 6. All-in test performance.**

Figure 7 gives the evaluation results for a series of self-out tests, and compares these results to the all-in results. Each self-out test corresponds to an adjustment of a threshold value on the cosine similarity score. In this case, there is no guarantee of an interest

match or that the topic was ever discussed previously (recall that over 40% of threads consist of a single post), and we see the results shift from Exact/Good Answer categories to Related/Unrelated categories. As we increase the threshold value from 0.1 to 0.7, the results falling into Related/Unrelated categories decrease greatly and the system returns less noise in the result set. These results are based on two semester’s of data; assuming that the course is similarly taught, we would expect to see them improve naturally over time. Although the results are not ideal, the matching, modeling, extraction, and evaluation processes show promise and merit iteration, as the understanding of complex questions is very much an unsolved problem.



**Figure 7. Self-out vs. All-in evaluation.**

## 6. DISCUSSION AND FUTURE WORK

In this paper, we addressed one of the challenges of natural language understanding in the context of learning, to automatically reply to students’ questions on a course discussion board. We implemented an intelligent discussion-bot, which can monitor students’ posts, match students’ interests, and display appropriate replies automatically. Evaluations of the resulting answers show that the technology can begin to meet students’ learning requests.

We are currently working to acquire data from other sources, including classification opinions from the students themselves, and web cast and camera document transcriptions from lectures. We are planning to improve several aspects of the system's performance. In the current discussion-bot system, the relations between posts are identified and analyzed manually; automatically labeling their relations based on language features is highly desirable. Some ideas from argumentation analysis to improve performance, and more complex reasoning-based strategies to provide more accurate and compact answers from archived threads, are also being considered.

## 7. ACKNOWLEDGMENTS

The work was supported in part by a grant from the Lord Corporation Foundation to the USC Distance Education Network and in part by *Learning by Reading*, DARPA grant DOI-NBC Contract No. NBCHC050051. We thank USC Professor Michael Crowley for providing the discussion archives, Lei Ding, Feng Pan, Patrick Pantel, Deepak Ravichandran, and Mei Si for their insightful contributions and Swapnil Patel for his effective support in integrating the discussion-bot to the ISI discussion board.

## 8. REFERENCES

- [1] ALICE. <http://www.alicebot.org/>.
- [2] Buckingham Shum, S., MacLean, A., Bellotti, V. M. E., and Hammond, N. V. 1997. Graphical argumentation and design cognition. *Human-Computer Interaction*, 12(3), 1997, 267-300.
- [3] Buckingham Shum, S. 2000. Workshop report: computer-supported collaborative argumentation for learning communities, *SIGWEB Newsl.* 2000. ACM Press, New York, NY, 27-30.
- [4] Cakir, M., Xhafa, F., Zhou, N., and Stahl, G. Thread-based analysis of patterns of collaborative interaction in chat, *AIED* 2005.
- [5] Carr, C. S. 2001. Computer-Supported Collaborative Argumentation: Supporting Problem-based Learning in Legal Education. *Proceedings of Euro-CSCL 2001*.
- [6] Hearst, M. A. Multi-paragraph segmentation of expository text. *Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics(ACL-94)*, 9-16, Las Cruces, New Mexico, 1994.
- [7] Hermjakob, U., Hovy, E. H., and Lin, C. 2000. Knowledge-based question answering. *TREC-2000*.
- [8] Hovy, E.H., Gerber, L., Hermjakob, U., Junk, M., and Lin, C. 2000. Question answering in Webclopedia. *Proceedings of TREC-2000*.
- [9] Isbister, K. Nakanishi, H., Ishida, T., Nass, C. (2000) Helper agent: Designing an assistant for human-human interaction in a virtual meeting space. *Proceeding of CHI'2000*, pp. 57-64.
- [10] Marom, Y. and Zukerman, I. 2005. Corpus-based Generation of Easy Help-desk Responses. *Technical Report*, School of Computer Science and Software Engineering, Monash University. Available at: <http://www.csse.monash.edu.au/publications/2005/tr-2005-166-full.pdf>.
- [11] Moldovan, D., Harabagiu, S., Pasca, M., Mihalcea, R., Girju, R., Goodrum, R., and Rus, V. 2000. The structure and performance of an open-domain question answering system. *Proceedings of ACL-2000*.
- [12] Nakanishi, H., Isbister, K., Ishida, T. and Nass, C. Designing a Social Agent for Virtual Meeting Space. Robert Trappl and Sabine Payr Ed., *Agent Culture: Designing Virtual Characters for a Multi-cultural World*, Lawrence Erlbaum Associates, pp. 245-266, 2004.
- [13] Nguyen, A. and Wobcke, W. 2005. An Agent-Based Approach to Dialogue Management in Personal Assistants. *Proceedings of the 2005 International Conference on Intelligent User Interfaces*, 137-144.
- [14] Pasca, M. and Harabagiu, S. High Performance Question/ Answering, in *Proceedings of SIGIR-2001*, pp. 366-374.
- [15] phpBB. <http://www.phpbb.com/>.
- [16] Prager, J. M., Chu-Carroll, J., and Czuba, K.W.. 2004. Question answering using constraint satisfaction. *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, 2004.
- [17] Salton, G. *Automatic Text Processing, The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, Reading, MA, 1989.
- [18] Shaw, E. Assessing and Scaffolding Collaborative Learning in Online Discussions. In *Proceedings of the 12th International Conference on AI in Education (AIED '05)*.
- [19] Soller, A., and Lesgold, A. 2003. A computational approach to analyzing online knowledge sharing interaction, *Proceedings of AI in Education 2003*, 253-260
- [20] USC Distance Education Network. <http://den.usc.edu/>.
- [21] Winograd, T. 1987. A Language/Action Perspective on the Design of Cooperative Work. *Human-Computer Interactions*, 3:1, pp. 3-30.
- [22] Xu, J., Licuanan, A., Weischedel, R. 2003. TREC 2003 QA at BBN: Answering Definitional Questions. *Proceedings of TREC 2003*.