IDENTIFYING DOMAIN REASONING TO SUPPORT COMPUTER MONITORING IN TYPED-CHAT PROBLEM SOLVING DIALOGUES

Angelica Willis, Ashana Evans, Jung Hee Kim, Kelvin Bryant
Dept. of Computer Science
North Carolina A&T State University
1605 E. Market Street, Greensboro, NC 27411
(336) 285-3695
jungkim@ncat.edu

Yesukhei Jagvaral, Michael Glass
Dept. of Computing and Information Sciences
Valparaiso University, Valparaiso, IN 46383
(219) 464-5161
michael.glass@valpo.edu

ABSTRACT

When students are working together solving a Java programming problem, can a computer gauge how often they show understanding? In the COMPS project, students in small groups engage in typed-chat problem-solving dialogues. This project applies topic modeling and text analytics toward computer assessment of the degree to which students are constructively discussing the problem. The aim is to provide a real-time assessment of the state of the conversation to an instructor overseeing the online conversations. Here we report on training machine classifiers to recognize parts of the dialogue where the students are reasoning about the Java problems.

INTRODUCTION

In COMPS exercises students work in small groups during their class lab time, solving exercises through typed-chat [4,5]. The instructors figuratively look over the shoulders of the students as they work, contributing to the typed-chat conversations when they think it will be useful. Students work in groups of three typically, a whole class has many groups for the instructors to check on. An example of such dialogue is shown in Figure 1.

This paper reports on work toward providing the instructors with a dashboard. The COMPS chat interface was designed by us for computer-supported collaborative instruction. Computer-monitoring will show the instructor indications of the state of each conversation. The dashboard indicator at the focus of this research will reflect whether student utterances show evidence they are discussing the topic. This paper explores the machine learning techniques to detect several aspects of the conversation that could, together, show discussion behavior:

a) The students are talking about the topic.

b) The students are agreeing or disagreeing with each other.

The intuition is that productive dialogues will contain both phenomena. For example, if the students are talking about the topic but not expressing agreement or disagreement, then possibly they are not addressing each other or engaging with each other's ideas.

The main method is to use text analytic techniques to identify these phenomena in individual student dialogue turns. Turns are machine-classified as containing the behavior or not. For the dashboard, a statistic will be derived based on the frequency of these two behaviors. Machine-identifying these behaviors in individual turns does not need high accuracy. For monitoring the state of the conversation, even a somewhat unreliable classifier could still provide enough correct data to tell the instructor which of the conversations are most in need of attention.

To train the machine classifier about 3000 lines of dialogue from 19 discussion groups were manually annotated as showing or not showing reasoning and agreement. Topic modeling was applied to the dialogues, discovering sets of related words that tended to occur together within the same turns. Each student utterance is then modeled as containing a mixture of the possible topics. Finally, machine classifiers were trained to try to predict the occurrence of the reasoning feature, based on the mixture of topics each turn was discovered to obtain.

The sections of this paper are: a description of the data and its annotation, a description of the text analytics process, the experimental results of classification tasks, and discussion and future work.

DIALOGUE DATA

Figure 1 contains an extract from a COMPS dialogue, extracted from the chat log file. Table 1 shows statistics on the dialogue turns used for this experiment.

In the Figure 1 dialogue, the students are answering questions about how to declare a Java method. The code and questions are visible to the students in a separate document outside the chat window. Three students labeled here A, B, and C are participating, as is a teaching assistant labeled TA. Their instructions are to come to agreement on small sub-problems, then have the TA judge their answers and possibly provide help. In addition to the dialogue text, Figure 1 shows "reason" and "agree" features. These were manually annotated.

The "reason" feature shows whether the turn evinced reasoning or understanding in the Java problem. It is marked "Y" or "N" for yes or no. Reasoning was shown if the student a) talked about constructs from Java or the program, and b) utilized some sort of reasoning. Saying "I understand" fails that test. So does simply saying some Java code, as in turn 14. The correctness of the student's statement and reasoning are not part of this judgment, it is enough that the student was talking about the material of the problem.

As a check on the manual annotation, the first 1000 'enter'-delimited chat turns were hand-annotated by two other people. Inter-rater agreement, measured by the kappa statistic [1], is kappa=0.47 for the reasoning annotation.

Turn	Stu	Dialogue Text	Reason	Agree
1-2	A	do this one work?	Y	--
		public double calculatePayment(double principleAmount, double interestRate, double totalCurrentMortgages)		
3	B	should t be void?	N	Dis
4	TA	Hm make sure you guys agree first and Ill come back in a sec		
5	C	no becuase void is if you are not returning a value	Y	Dis
6-7	B	yea i know.. So to my understanding we are returning something.	Y	--
		Ok i understand now		
8	C	I think what [student A] has works because it calls everything thta we are looking for	Y	Agr
9	A	wait we are not instanting a Morgage object correct, so doesn't that mean it's a no-arg constructor?	Y	Dis
10-12	B	See I was thinking about that.	Y	Agr
		but it does have parameters		
		lets try your original answer, agree?		
13	C	agree	N	Agr
14	B	public double calculatePayment(double principleAmount, double interestRate, double totalCurrentMortgages) {} @TA	N	--
15	TA	Ok you need one more modifier so it can be accessed without instanting the Mortgage class *instantiating		

Figure 1: Transcript of Discussion with Manually-Annotated Features

The “agree” feature is notated “Agr” here if the student shows evidence of agreeing with a previous student’s utterance, “Dis” for disagreeing, and “--” if neither is shown. Notice that the word “agree” in the dialogue is not always an indication of agreement. In turn 13 the student agrees with another. By contrast in turn 4 the TA tells the students to come to agreement, and in turn 12 student B is seeking agreement. Agreement and disagreement were annotated by hand separately as binary present/not-present features, but they are presented here as a single three-valued feature.

In the chat interface, students end turns by pressing 'enter.' In normal verbal conversation, several sentences in a row by one person are considered a single dialogue turn. Therefore when consecutive chat-turns are by the same person they are concatenated together and treated as a single dialogue turn. Turns 1 and 2, for example, were from the same student. They were annotated as providing reasoning because turn 1 contained some words that occur as part of reasoning and turn 2 contains Java. The COMPS chat interface permits students to overlap their typed responses to each other [4]. To sequence chat turns into a chronologically linear conversation, they were ordered by the timestamp of the 'enter' which ended a turn.

Table 1: Statistics of Dialogue Data

Number of enter-delimited chat-turns	3497
Number of dialogue turns (combining consecutive chat-turns from one person)	2394
Number of reasoning turns	888
Percent reasoning turns	37%
Number of dialogues	19
Average turns per dialogue	126
Average reasoning turns per dialogue	47

TEXT ANALYTICS PIPELINE

The main steps in the text analytics pipeline are: removing proper nouns and similar named entities, building a topic model from part of the transcript, applying the model to the whole transcript, identifying additional text features, and training and testing classifiers.

Topic Modeling

The primary text analysis method used in this experiment is topic modeling [6]. Texts (in this case, the dialogue turns) are converted to bags of words. A topic is a collection of words that likely occur together within the same dialogue turn. The experimenter specifies the number of topics to be included in the model. Based on training texts, the algorithm builds a probability distribution, where each word w_i is assigned a probability that it occurs within topic t_j. After a topic model is built, new dialogue turns can be analyzed according to the topics they contain. Every dialogue turn is assigned a vector of real-valued numbers in the range [0.0, 1.0] representing the participation of each topic within that turn.

Figure 2 shows three of the topics that the model discovered in the COMPS dialogues. The parentheses show the number of times each word occurs within each topic in the training data. In Figure 2 the topic names, e.g. "agreement and disagreement," were added manually for illustration. The topic model is constructed without supervision and

without reference to the meanings of the words. Nevertheless many topics that the model produces comprise recognizable semantic concepts.

Topic 1: *Agreement and Disagreement*	Topic 2: *Java Language*	Topic 3: *The Program Under Discussion*
answer (51), correct (38), yeah (30), yea (26), part (25), agree (23), move (18), makes (14), wrong (14), problem (12), cool (12), explanation (11), sense (11)	private (119), double (86), data (81), type(74), int (40), modifier (38)	rate (31), amount (27), interest (25), principal (23), variable (16), total (16), number (16), current (15), class (15), double (15), parameters (14)

Figure 2: Three Prominent Topics Produced by Topic Modeling of the COMPS Dialogues

We implemented our topic models using MALLET, a Java library [6] and replicated some of the results with gensim, a Python library [7]. For both libraries the topic modeling algorithm is rooted in Latent Dirichlet Allocation (LDA), a generative probabilistic model for collections of discrete data often applied to text analysis.

Named Entity Elimination

In the dialogues students often refer to people by name. For example in Figure 1 turn 8 one student addresses another. User names and other proper nouns are not relevant to the topic. Stanford's CoreNLP Named Entity Recognition library [2] was used to remove proper nouns from the text before subjecting it to topic modeling.

Additional Text Features

In addition to the topic models, a small number of other features in each dialogue turn were annotated by computer for use by classifier models [3]. These are:

a) Participation, a statistic based on the fraction of the conversational turns uttered so far that came from this person. Participation is adjusted according to the total number of people in the conversation, if everybody is participating equally the statistic is 0.5 for each person.

b) Discourse marker words at the beginning of a turn, such as "so" or "therefore," which may be indicative of reasoning. This is a 0 or 1 binary feature.

c) Question marks, emoticons, and pronouns such as "you," "we," or "us." These are also binary features taking the value 0 or 1. The presence of these items in a dialogue turn usually indicate that the speaker is conversationally addressing another person.

Classifiers

Weka (Java) and Scikit-Learn (Python) were used for training and testing classifiers. Trained classifiers can be saved for repeated use in testing on new data, and for eventual incorporation into the COMPS dashboard. This project tested J48 decision tree classifiers from Weka and Scikit, multiple linear regression from Scikit, and LogitBoost linear regression from Weka.

EXPERIMENTAL RESULTS

We trained decision tree and linear regression classifiers for the "reason" feature. The classifiers utilized only topic model, or the topic model augmented by the additional text features listed above. For the regression models the class variables were coded as numeric 0 or 1. The discrimination cutoff value for the regression results (the value of the regression result that was used to discriminate between the two cases) was picked to maximize F1 value. For the decision tree models we specified a minimum of 10 items per leaf node in the tree.

We tested with differing numbers of topics (number of clusters the topic modeler is set to generate) of 10, 20, 70 and 100 topics. Generally 10 topics worked best, thus the primary results we report used 10 topics.

Table 2 shows the results of testing J48 decision tree classifiers and linear regression classifiers, using a 10-topic model. Results are reported for classifiers that used the topic values only, and for the topics plus the other features. Table 3 shows the results for topics-only classifiers using larger topic models. The 10 topic classifiers were trained on 80% of the data and tested on the remainder, using 5-fold cross validation. The results for more than 10 topic classifiers were reported from training on a random 70% of the data and testing on the remaining 30%.

Table 2. Best Decision Tree and Regression Classifiers, Using 10 Topic Model

Other Feat?	J48 Decision Tree			Linear Regression			
	Precision	Recall	F1	Precision	Recall	F1	ROC-AUC
No	0.749	0.815	0.78	0.448	0.806	0.58	0.725
Yes	0.771	0.821	0.79	0.533	0.656	0.59	0.756

Table 3. Best Classifiers using More Than 10 Topic Models

	J48 Decision Tree			LogitBoost Linear Regression		
No. Topics	Precision	Recall	F1	Precision	Recall	F1
20	0.730	0.332	0.46	0.675	0.451	0.54
70	0.667	0.354	0.46	0.704	0.430	0.53
100	0.685	0.354	0.47	0.592	0.523	0.56

The primary results are: A) F1 values of 0.6 to 0.75 are achievable, B) modeling 10 topics is better than modeling larger numbers of topics, and C) adding the additional non-topic features provides very little improvement.

DISCUSSION AND FUTURE WORK

For a statistic for an instructor dashboard, F1 scores in the range of 0.6 to 0.75 might be adequate. Over half of the reasoning turns would be correctly identified by the classifiers tested above. With 47 reasoning turns per dialogue, this accuracy might sufficiently detect dialogues that are abnormally high or low in reasoning turns. The low interrater reliability between human annotaters hints that there may be little room for improvement in classifier accuracy given our current training dataset.

Providing more text to the classifier is potentially more accurate than classifying individual turns, many of which are quite short. A possible variation on the text processing would be to apply the classifiers to sliding windows in the text stream, for example the most recent hundred words taken from all speakers.

Manual inspection of the topic models, as in Figure 2, shows that they often seem to identify agreement and disagreement as a topic. This suggests that the same approach may be fruitful for the next task of identifying agreement/disagreement phenomena within a conversation.

A concern is that the topic models in this experiment contain many words that are specific to the problem domain. The dialogues in this study all came from a discussion of classes, objects, and reference types. A different Java exercise would contain a different mixture of Java concepts and vocabulary. A classifier model that is problem-independent would have considerably greater utility. There is evidence that a domain-independent judgment of student reasoning may be possible. Other experimenters, e.g., have gauged student understanding in tutoring dialogue, utilizing only domain-independent indications of student affect [8]. Another set of experiments is thus underway to apply topic modeling and train classifiers using only common English words, optionally augmented by a general Java concept vocabulary. Another experiment will be training on bigrams in addition to the individual words. This may create models sensitive to short phrases that are indicative of reasoning. For example, in the Figure 1 dialogue such bigrams include “let’s try” and “you need.”

ACKNOWLEDGMENT

Partial support for this work was provided by the National Science Foundation's Improving Undergraduate STEM Education (IUSE) program under Award No. 1504917. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Di Eugenio, B., Glass. M., The Kappa statistic: A second look, *Computational Linguistics* 30 (1) 95–101, 2004

[2] Finkel, J. R., Grenager, T., Manning, C., Incorporating non-local Information into information extraction systems by Gibbs sampling. Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), 363–370. http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf

[3] Glass, M., Kim, J. H., Bryant, K., Desjarlais, M. Indicators of Conversational Interactivity in COMPS Problem-Solving Dialogues. *Third Workshop on Intelligent Support for Learning in Groups (ISLG)*. Honolulu, 2014.

[4] Glass, M., Kim, J. H., Bryant, K., Desjarlais, M.,Come let us chat together: simultaneous typed-chat in computer-supported collaborative Dialogue. *Journal of Computing Sciences in Colleges*, 31 (2), 96–105, 2015.

[5] Kim, J. H., Kim, T., Glass, M., Early experience with computer-supported collaborative exercises for a 2nd semester Java class. *Journal of Computing Sciences in Colleges*, 32 (2) 68–76, 2016

[6] McCallum, A. K., MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu. 2002.

[7] Řehůřek, R., Sojka, P. , Software framework for topic modeling with large corpora, *Proc. LREC Workshop on New Challenges for NLP Frameworks*, 45–50, 2010.

[8] Williams, C., D'Mello, S. K., Predicting student knowledge level from domain-independent function and content words. In Aleven et al., *Intelligent Tutoring Systems 10th International Conf*. pp. 62–71, 2010.