

# A STATE TRANSITION MODEL FOR STUDENT ONLINE DISCUSSIONS

SOO WON SEO<sup>1</sup>, JEON-HYUNG KANG<sup>1</sup>, JOANNA DRUMMOND<sup>2</sup>,  
JIHIE KIM<sup>1</sup>

1. University of Southern California/Information Science Institute,  
U.S.A.

2. University of Toronto, Canada

---

Discussion boards, or online asynchronous discussions, have become increasing popular course tools for university-level engineering courses. As a step towards assessing student learning in online discussion, we are investigating whether it is possible to characterize successful versus unsuccessful question and answer (Q&A) type discussions. We use a four-state model in identifying different stages of the Q&A dialogue. In this paper, we examine whether it is possible to automatically classify patterns of interactions using a state transition model and identify successful versus unsuccessful student Q&A discussions. For four-state model classification, we apply Conditional Random Field and Hidden Markov Models to capture transitions among the states. The initial results indicate that such models are useful for modeling some of the student dialogue states. We also show the results of classifying threads as successful/unsuccessful using the state information.

Key Words and Phrases: Student online discussions, Q&A discussion classification

---

## 1. INTRODUCTION

After the emergence of the Internet in the 1990's, many web technologies have been utilized for educational purposes. Online discussion boards, one such web technology, have been a medium for students and instructors to share their ideas in web-enhanced traditional courses and web-based distance-learning courses. This work is gathered from the student discussion board that is used by a undergraduate computer science course at the University of Southern California. The course contains programming projects, where a student needs timely support from the instructor or other students to improve his or her performance.

As a step towards assessing student learning in online discussions and assisting instructors, we are investigating whether it is possible to characterize successful versus unsuccessful question and answer (Q&A) type discussions. A four-state model was generated based on an analysis of sample discussion threads and its dialogue status [3]. The states that we model are *initiation*, *understanding*, *solving* and *closing*. In this paper, we examine whether it is possible to automatically classify patterns of interactions using a state transition model and identify successful versus unsuccessful student discussions. In order to capture transitions among the states, we use graphical models, Conditional Random Fields (CRF) and Hidden Markov Model (HMM), that are often used for labeling sequential data.

---

Authors' addresses: Soo Won Seo, Department of Computer Science, University of Southern California, CA, U.S.A. E-mail:soowonse@usc.edu; Jeon-Hyung Kang, Department of Computer Science, University of Southern California, CA, U.S.A. E-mail:jeonhyuk@usc.edu; Joanna Drummond, Department of Computer Science, University of Toronto, Ontario, Canada. E-mail: jmd73@pitt.edu; Jihie Kim, Artificial Intelligence Group, Information Science Institute, CA, U.S.A. E-mail: jihie@isi.edu;

We use information sharing ‘speech acts’ and user dialogue roles as features for generating the classifiers. The overall classification of the threads as successful or unsuccessful relies on the state model. The initial results indicate that graphical models are useful for identifying some of the states. Using annotated state information, the system can classify the discussion successfulness with 96% accuracy.

## 2. CHARACTERIZING SUCCESSFUL VS. UNSUCCESSFUL THREADS WITH A STATE TRANSITION MODEL

In a Q&A type discussion, if an information seeker’s problems get resolved, we can say that the discussion reached a successful conclusion, or simply, that the discussion was successful. What, then, makes one discussion thread successful and another unsuccessful? Successful discussions can be defined in many ways: “Is the initial problem resolved?” “How many answers do we have?” “Is instructor involved in this discussion thread?” and so on. We define successful discussion as a discussion in which all of an information seeker’s questions get resolved, including initial questions, related questions, similar questions, and questions about derived problems. A four-state model was developed based on an analysis of sample discussion threads: An initiation state, an understanding state, a solving state and a closing state [4].

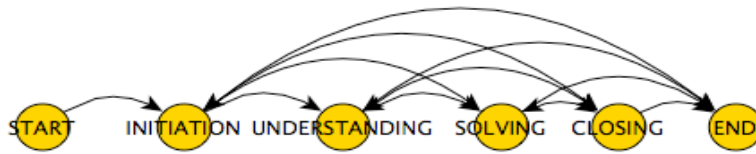


Fig. 1. State Transition Model

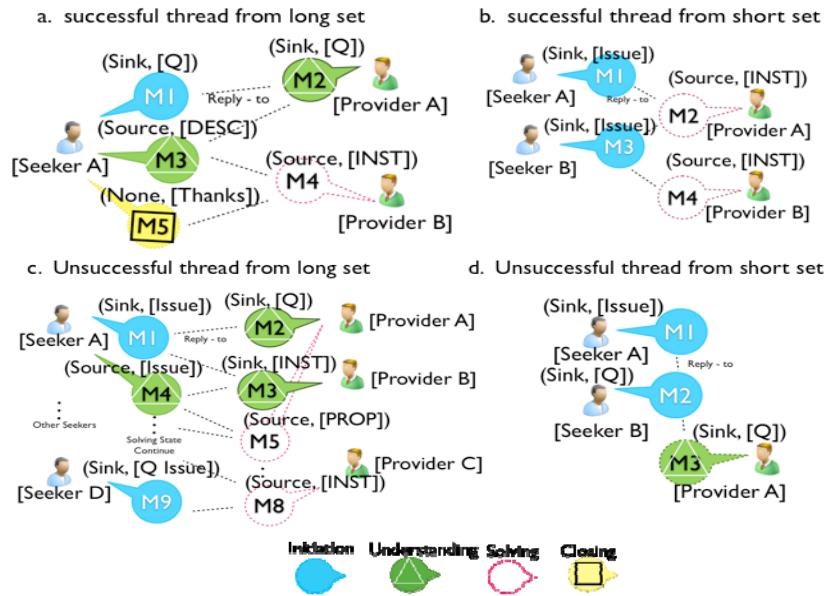


Fig. 2. Discussion thread examples (a: I-U-S-C | b: I-S-I-S | c: I-U-S-I | d: I-U)

In the first state (initiation), there must be a problem that exists, which is almost always proposed by the information seeker. A problem is an issue that makes it difficult to achieve a desired goal, and its context, which may include the author's unique situation, issue or proposition. In the second state (understanding), the problem is elaborated through communication with other users, who need to understand why this problem exists, and identify the obstacles that the user confronts. Multiple messages may be required in this state, depending upon the problem type, for example, whether the problem is common or unusual, and the author's ability to clearly describe the problem. In third state (solving), information providers give instructions, propositions, or hints that suggest solutions or actually solve the problem. The solving state may also require multiple messages if the information does not satisfy the seeker. Or, if the information provider misunderstands the root cause of the problem, the state machine might transition back to the understanding state.

In Figure 2, we describe four discussion thread examples with the transition model. Threads a. and c. are long, and threads b. and d. are short. We labeled user roles (seeker or provider), message roles (sink or source), and speech acts, such as question, instruction, description, done, issue, and proposition that can be automatically labeled by our classifiers [1], [2]. Thread a. has all four states in sequence, ending with a *closing*. Thread b. doesn't go through the *understanding* state and *closing* is missing, but it ends with a *solving* state without an additional issue. Threads c. and d. are both considered unsuccessful since thread c. ends at the seeker's *initiation* state and thread d. ends at the provider's *understanding* state. There are other possible patterns that can be captured with the transition model. For example, for difficult problems, students may stay in the understanding or solving state longer than average, making the thread longer. The model also provides clues about the type of student participation such as problem initiator, problem elaborator (in the understanding state), problem solver, etc. We expect that the state transition model will help us characterize the issues and qualitatively profile students. Table I describes circumstances for which state transitions occur with examples.

### 3. CLASSIFYING STATES WITH GRAPHICAL MODELS

This section describes how we model each state using graphical models: conditional random field (CRF) and hidden Markov model (HMM).

#### 3.1 FEATURES USED

The experimented data consists of a subset of one semester's forums, which is annotated for both state transitions and sink/source information. The agreement between two annotators for state transition had a final Kappa of 0.84. Sink/source information describes both the characteristics of the post, and the characteristics of the poster. There are four types of sink/source information—hasSink, hasSource, isProvider, and isSeeker. hasSink indicates if the post contains a request for information. while hasSource indicates if the post gives information. Note that these two categories are not mutually exclusive—one post can give and request information. isProvider and isSeeker are mutually exclusive and describe the poster's main intention—if the poster wants to provide information, or is actively seeking information within the thread. Inter-annotator agreement for hasSink had a Kappa of 0.93, hasSource had a Kappa of 0.96, and isProvider/isSeeker had a Kappa of 0.99. Initiation only occurs with information seekers.

All other states can occur with information providers or seekers, however, both understanding and solving states begin with information providers. Since both understanding and solving phases can contain multiple iterations of question-answer discussion, posts answering or further questioning those initial solving or understanding posts are also labeled solving or understanding, respectively. A total of 73 threads, containing 254 posts, were used to build a model for state transition. 151 of these posts were labeled solving, 93 were labeled initiation, 8 were labeled closing, and 2 were labeled understanding.

First, we decided to initially use all sink/source information as features for this classification problem. Upon inspection of the annotation manual, it seemed that most definitions could be written as a combination of sink/source information. Additionally, an automatic classifier for sink/source information already exists, with a F-measures (a score that combines recall and precision) of 0.88 for hasSink, 0.83 for hasSource, and 0.84 for isProvider/isSeeker [1]. However, since this is preliminary work, we chose to use the gold-standard human classified sink/source information.

Second, we added one more feature to classify state, which is THANK relation between posts. This relationship is usually much correlated to the closing state because people tend to appreciate when they got what they want in the thread.

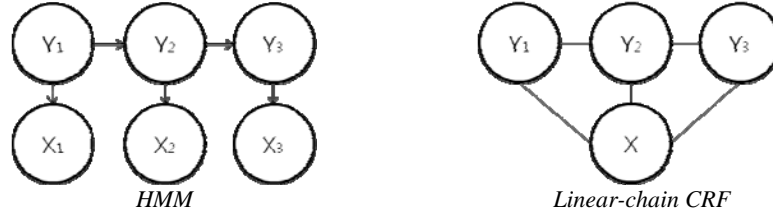
Table I. State transition matrix examples.

State	Initiation	Understanding	Solving	Closing
Initiation	I: 1. Other seekers' agreement on initial seeker's problem, e.g., "Me too, I 'm getting the same error."	U: 1) To understand seeker's problem, e.g., "How did you propagating your dirty bit?"	S: 1) Provider's answer with instruction, disagree/agree, proposition, or question, e.g., "Try putting them in a loop." 2) Seeker found answer by himself, e.g., "I just got this."	
U	Same as I.	Same as U.	Same as S.	
Solving	1) Same as I. 2) Same Seeker reports derived problems, e.g., "I found the reason of the problem and now I have another problem..."	1) Seeker explains details about his/her problem and Providers ask about problem detail again until they understand the seeker's problem correctly, e.g., "Then where did you set the flag?"	1) Same as S. 2) Seeker does not understand Provider's answer message, e.g., "Where exactly can I get semaphore?"	C: 1) Seeker thanks the Provider 2) Provider gives praise to Seeker for solving the problem.
Closing	1) Same as I. 2) Same Seeker reports derived problems, e.g., "I found the reason of the problem and now I have another problem..."		1) If closing is reached by non-initial problem provider and if initial problem was not resolved, he can bring state into Solving state, e.g., "But I am still getting bus error from ..."	Same as C.

### 3. 2 SELECTING CLASSIFIER MODELS

After looking at the data, we chose representative supervised machine learning algorithms to try to model these state. We chose to investigate using decision trees, which would classify each post individually, and hidden Markov model (HMM) and linear-chain Conditional Random Field (CRF) which would classify each thread as a whole. Decision trees were chosen for multiple reasons. First upon inspecting the annotation manual, it became apparent that our feature-space was highly partitionable if we used sink/source information as our features. Since decision trees iteratively partition the

feature-space, this seemed to be a natural fit. Also, since decision trees produce output that's easily human-readable, they are commonly chosen as a first-pass algorithm. We chose to investigate hidden Markov model[Figure.3] since it takes into account characteristics of the thread as a whole in generative way. However, unlike decision tree, which trains at the post-level, a HMM trains at the thread level, so we have "more" training data for the decision trees, at the cost of the thread's characteristics. Finally, we chose to use linear-chain CRF[Figure.3] to improve learning by considering arbitrary dependencies on the feature sequences in discriminative way[5].



**Fig. 3.** Factor Graphs of HMM and linear-chain CRF

$Y_i \in \{ \text{Initiation, Understanding, Solving, Closing} \}$

$X_i \in \{ \text{hasSink, HasSource, isProvider, isSeeker, isPosAck} \}$

To test supervised learning classifiers, we compare these classifiers with the same distribution as the training set. For each classifier, we performed 10-fold cross-validation. k-fold cross-validation is a process where the data is randomly partitioned into k complementary subsamples, then k different models are built, each one testing on its own k-th portion of the data, and training on the remaining data. We then analyze the data based on Kappa, accuracy (i.e. percent correct overall), precision (i.e. correctly classified in a category over total number of that category classified), and recall (i.e. percentage correctly classified per category). To calculate these measures for the 10-fold cross-validation, we use a weighted average for precision, recall, and accuracy, with the weight controlling for the number of posts each model classifies.

### 3.3 CLASSIFICATION RESULTS

#### State Classification

Table II shows kappa values with the human annotation. Table III shows precision and recall scores for the three classifiers. The three classifiers show different strengths in this experiment. Linear-chain CRF shows highest kappa and accuracy although it cannot recognize understanding state, which mainly comes from the fact that only two out of 254 posts are in understanding state. HMM is the only model to correctly classify any understanding instances. We expect that as we use more data, we can improve the accuracy of the classifiers. For implementation, we used Jahmm[6] for HMM, mallet[7] for linear-chain CRF and Weka[8] for decision tree

Table II. Kappa and Accuracy for tree, HMM and linear-chain CRF models

Model	Kappa	Accuracy
Tree (J48)	0.6964	0.8386
HMM	0.6493	0.8071
LCCRF	0.7824	0.8937

Table III. Precision and Recall for Rand, Decision Tree, HMM and linear-chain CRF models

Model	Precision				Recall			
	I	U	S	C	I	U	S	C
Tree (J48)	0.7317	0.0000	0.9516	0.7143	0.9677	0.0000	0.7815	0.6250
HMM	0.6691	0.5000	1.0000	0.6250	0.9785	0.5000	0.7152	0.6250
LCCRF	0.9733	0.0000	0.8721	0.5714	0.7849	0.0000	0.9934	0.5000

### Discussion Thread Classification

We used the above state information and the final post sink/source labels for classifying successful versus unsuccessful discussion threads. We have the same accuracy of 95.83% in three supervised learning algorithms which are decision tree, support vector machine and logistic regression. The results indicate two things: first, state information and the final post sink/source labels are worthwhile to be used in classifying successful threads in online discussion boards. Second, this data used in the feature space is highly separable with some exceptions, which come from unanswered original question and partially answered question. All the three classifiers failed to differentiate these exceptions because they reside virtually in the same properties as successful threads usually are. For this reason, the features related to an existence of an unanswered question may improve performance.

Table IV. Precision, Recall and Accuracy of classifying Successful/Unsuccessful Threads

Model	Accu(%)	Accu(%) (Short)	Precision	Recall	Accu(%) (Long)	Prec	Recall
Tree (J48)	95.83	95.65	0.97	0.90	96.30	0.98	0.88
SVM	95.83	95.65	0.97	0.90	92.56	0.85	0.85
Logistic Regression	95.83	94.48	0.92	0.90	92.56	0.85	0.85

Additionally, we separate our threads into two groups by the length of threads. One is short thread group and the other is long thread group. We define short thread as a thread with less than three posts. Others are long threads. Accuracy rates and precision/recall in classifying with short and long threads are also presented in Table IV. The three classifiers show similarly good performance. All the classification was implemented by using Weka[8] with ten-fold validation .

## 4. RELATED WORK

There have been various approaches to assessing collaborative activities. Various approaches of computer supported collaborative argumentation have been discussed.

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 on discourse processing 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 both a thread level and a message level.

There has been prior work on dialogue act analysis and associated surface cue words [13], [14]. Although they are closely related to our speech act (sink/source) 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.

There has been effort to analyze interactions in on-line communities. For example, Talk-to-me [15] can predict the likelihood of that a message will receive a reply based on the content of the message and the message sender. Our work provides complementary capability by providing Speech Act 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 related problems including characterizing dialogue with tutors [16]. Our work extends the use and demonstrates how we can profile or find patterns in online discussion threads, where threads are represented by graphs and messages within a thread are represent nodes in the graph.

## 5. DISCUSSION AND FUTURE WORK

We have presented a model for automatically analyzing patterns of student interactions within discussion threads. With respect to modeling state transitions, our preliminary models show Kappa scores within the range of our human annotators. Although the current state classifiers were created from sink/source annotations, as we have automatic classifiers sink/source, we plan to generate end-to-end automatic classifiers. By combining these automatic classifiers, we hope that we can create assessment tools for instructors. As the number of participants increases, some over hundreds students, instructors need assistance in efficiently processing a large amount of Q&A threads and figuring out which threads need close follow-up.

We also plan to relate discussion topic models to the discussions state model so we can identify topics of unresolved discussions. Although the presented classifiers are only tested with a sample data from computer science online discussions, the approach can be adopted in analyzing large online discussion system in real-time.

## 6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0917328.

## 7. REFERENCES

- [1] Kang, J. and Kim, J. 2010. Profiling Message Roles in Threaded Discussions using an Influence Network Model, internal project report.

- [2] Drummond, J., and Kim, J. (2010). Role of Elaborated Answers on Degrees of Student Participation in an Online Question-Answer. Under review.
- [3] Kang, J.H., Kim, J., and Shaw, E. (2010). Modeling Successful versus Unsuccessful Threaded Discussions. Workshop on Opportunities for intelligent and adaptive behavior in collaborative learning systems, 13.
- [4] Kim, J., Chem, G., Feng, D., Shaw, E., and Hovy, E. (2006). Mining and assessing discussions on the web through speech act analysis. Proceedings of the Workshop on Web Content Mining with Human Language Technologies at the 5th International Semantic Web Conference.
- [5] J. Lafferty, A. McCallum, and F. Pereira. Conditional random Fields: Probabilistic models for segmenting and labeling sequence, data. In Proc. ICML-01, pages 282-289, 2001.
- [6] Jean-Marc François. Jahmm, 2009.<http://code.google.com/p/jahmm/>.
- [7] Andrew Kachites McCallum. 2002. MALLET: a machine learning for language toolkit. <http://mallet.cs.umass.edu>
- [8] Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. 2nd edn. Morgan Kaufmann, San Francisco (2005).
- [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, 8 (3), pp. 243-281.
- [11] Soricut, R. and Marcu, D. 2003. Sentence level discourse parsing using syntactic and lexical information. In Proceedings of HLT/NAACL-2003.
- [12] Sporleder, C. and Lapata, M. 2005. Discourse chunking and its application to sentence compression. In Proceedings of HLT/EMNLP 2005.
- [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] Arguello, J., Butler, B. S., Joyce, L., Kraut, R., Ling, K. S., & Wang, X. (In press) Talk to me: Foundations for successful individual-group interactions in online communities. In CHI 2006: Proceedings of the ACM conference on human-factors in computing systems.
- [16] Boyer, E., Phillips R., Ingram, A., Ha E., Wallis, M., Vouk, M. & Leste J. (2010). Characterizing the Effectiveness of Tutorial Dialogue with Hidden Markov Models, Proceedings of the Intelligent Tutoring Systems conference