

## ABSTRACT

EZEN CAN, AYSU. Unsupervised Dialogue Act Modeling for Tutorial Dialogue Systems. (Under the direction of Dr. Kristy Elizabeth Boyer.)

Tutorial dialogue systems interact with students using rich natural language, in support of a learning task. These systems have been shown to be effective at supporting student learning, in some cases rivaling the effectiveness of human tutors. However, due to the engineering effort required within their various components, building and developing tutorial dialogue systems is a challenging task. To address this challenge, machine learning approaches that automatically learn from human-human dialogue are the focus of increasing attention. One of the most important tasks for a tutorial dialogue system is to understand students' natural language input in order to respond accordingly. This dissertation research focuses on techniques for understanding dialogue acts, which are used to represent the intention of an utterance.

The approach presented here leverages unsupervised machine learning to model dialogue acts. The goal of this approach is to dramatically decrease the manual effort required for engineering the dialogue management module of tutorial dialogue systems, including the work required to manually label large corpora for training dialogue act classifiers. Unsupervised models represent directly data-driven groupings of student utterances into the most natural clusters, rather than manually labeled tags that were created based upon dialogue taxonomies. The work presented in this dissertation shows that unsupervised models can be effectively used for modeling student dialogue acts. As part of this dissertation research, several types of unsupervised dialogue act classifiers have been built and investigated, including two models that adapt information retrieval approaches for the task of dialogue act clustering, and another model that explicitly models task events and learner characteristics. In addition, an extended set of features including multimodal features were explored and shown helpful for particular dialogue acts. Initial evaluations of the dialogue act classifiers utilized manually labeled data for computing accuracy.

Based on the encouraging results obtained by comparing proposed unsupervised models to manual labels, in-context evaluation was conducted in a simulated environment to investigate the performance of unsupervised models in a tutorial dialogue system. The results indicated that this unsupervised model holds promise for incorporation into a deployed tutorial dialogue system. Motivated by the results of the simulated in-context evaluation, a tutorial dialogue system was built that relies on an unsupervised model for dialogue act classification in real time. We measured how well the unsupervised dialogue act classifier could support student learning and engagement with a tutorial dialogue system compared to a supervised model. The results of user studies indicate that unsupervised techniques support significant student learning while achieving positive user satisfaction scores and tutor feedback evaluations.

The results of this dissertation research suggest that in the future, unsupervised dialogue act models may dramatically change the way tutorial dialogue systems are built. It presents an important step toward creating fully data-driven tutorial dialogue understanding models utilizing unsupervised methodologies that address the labor-intensive efforts associated with the current generation of dialogue systems.

© Copyright 2015 by Aysu Ezen Can

All Rights Reserved

Unsupervised Dialogue Act Modeling for Tutorial Dialogue Systems

by  
Aysu Ezen Can

A dissertation submitted to the Graduate Faculty of  
North Carolina State University  
in partial fulfillment of the  
requirements for the Degree of  
Doctor of Philosophy

Computer Science

Raleigh, North Carolina

2015

APPROVED BY:

---

Dr. Tiffany Barnes

---

Dr. Munindar Singh

---

Dr. Jimmy Scherrer

---

Dr. Kristy Elizabeth Boyer  
Chair of Advisory Committee

## DEDICATION

To my loving family, my mom, dad, mother-in-law, father-in-law, sister, brothers, Leyla Gun, and mostly to my dear husband, whose unconditional support and encouragement made it possible.

## BIOGRAPHY

Aysu Ezen Can was born in Ankara, Turkey where she grew up. She obtained a Bachelor of Science degree in Computer Engineering from Bilkent University in 2011. She received several awards and honors in her B.S. study. She was a research assistant in the LearnDialogue group, and a member of the Center for Educational Informatics during her Ph.D. studies, and also served as a teaching assistant. She held a summer research internship at IBM Watson Research Center in the summer of 2013. Aysu married in 2013.

## ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest appreciation to my advisor Dr. Kristy Elizabeth Boyer for her help. She has been a great advisor throughout the years of collaboration. I am deeply grateful to the members of my committee, Tiffany Barnes, Munindar Singh and Jimmy Scherrer for their insightful comments and suggestions. My colleagues in the LearnDialogue Group have provided vital support and collaboration.

Last but not the least, I would like to thank my husband and my parents, for their patience and encouragements during my pursuit of the Ph.D. study. It wouldn't have been possible without their emotional support and loving care.

# TABLE OF CONTENTS

<b>LIST OF TABLES</b> . . . . .	<b>viii</b>
<b>LIST OF FIGURES</b> . . . . .	<b>x</b>
<b>Chapter 1 Introduction</b> . . . . .	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Research Questions and Hypotheses . . . . .	2
1.3 Approach . . . . .	4
1.4 Contributions . . . . .	4
1.5 Organization . . . . .	6
<b>Chapter 2 Background and Related Work</b> . . . . .	<b>7</b>
2.1 Tutorial Dialogue Systems . . . . .	7
2.2 Dialogue Act Classification . . . . .	12
2.2.1 Supervised Dialogue Act Classification . . . . .	13
2.2.2 Unsupervised Dialogue Act Classification . . . . .	13
<b>Chapter 3 Corpora</b> . . . . .	<b>16</b>
3.1 JavaTutor 2007 Corpus . . . . .	16
3.2 JavaTutor 2011 Corpus . . . . .	17
<b>Chapter 4 Dialogue Act Classification: Query-Likelihood Clustering</b> . . . . .	<b>22</b>
4.1 Query-Likelihood Clustering Methodology . . . . .	23
4.1.1 Natural Language Preprocessing . . . . .	23
4.1.2 Query-Likelihood Representation . . . . .	24
4.1.3 Clustering . . . . .	24
4.2 Experiments with Query-Likelihood Clustering . . . . .	25
4.2.1 Parameter Tuning . . . . .	26
4.2.2 Accuracy in Identifying Manually Labeled Dialogue Acts . . . . .	29
4.2.3 Analysis of Clusters . . . . .	31
4.3 Query-Likelihood Clustering: Discussion . . . . .	32
<b>Chapter 5 Dialogue Act Classification: Markov Random Field Based Approach</b> <b>35</b>	
5.1 Unsupervised Dialogue Act Modeling . . . . .	35
5.1.1 Utterance Similarity Calculation . . . . .	36
5.1.2 Clustering . . . . .	40
5.2 Experiments with Comparisons . . . . .	41
5.3 Evaluation of MRF-Based Clustering . . . . .	41
5.3.1 Test Set Accuracy, Precision and Recall . . . . .	42
5.3.2 Qualitative Evaluation . . . . .	44
5.4 Discussion: MRF-Based Clustering . . . . .	46
<b>Chapter 6 Enriching Feature Sets for Task-Oriented Dialogue Act Modeling</b> . <b>48</b>	

6.1	Features . . . . .	49
6.1.1	Lexical Features . . . . .	49
6.1.2	Dialogue Context Features . . . . .	50
6.1.3	Task Features . . . . .	50
6.2	Enriched Feature Set: Methodology . . . . .	51
6.2.1	Dialogue Act Modeling With $k$ -medoids Clustering . . . . .	51
6.2.2	Experimental Results . . . . .	53
6.2.3	Utilizing Dialogue History . . . . .	53
6.3	Results . . . . .	55
<b>Chapter 7 Incorporating Learner Characteristics into Dialogue Act Models . .</b>		<b>59</b>
7.1	Dialogue Act Modeling with Learner Characteristics: Methodology . . . . .	60
7.1.1	Experimental Design . . . . .	63
7.1.2	Unsupervised Dialogue Act Modeling with Learner Characteristics . . . . .	64
7.2	Results of Dialogue Act Classification by Learner Characteristics . . . . .	66
7.3	Discussion: Learner Characteristics and Dialogue Act Classification . . . . .	69
<b>Chapter 8 Incorporating Multimodal Features into Dialogue Act Models . . .</b>		<b>73</b>
8.1	Features . . . . .	74
8.2	Methodology: Clustering with Multimodal Features . . . . .	76
8.3	Experiments: Clustering with Multimodal Features . . . . .	79
8.4	Results: Clustering with Multimodal Features . . . . .	80
<b>Chapter 9 Preliminary In-Context Evaluation of Unsupervised Dialogue Act Classifiers . . . . .</b>		<b>86</b>
9.1	Preliminary Evaluation: Methodology . . . . .	86
9.1.1	Unsupervised Dialogue Act Classifier Being Evaluated . . . . .	87
9.1.2	Evaluation Framework . . . . .	88
9.2	Preliminary In-Context Evaluation: Results . . . . .	91
9.3	Preliminary In-Context Evaluation: Discussion . . . . .	91
<b>Chapter 10 A Tutorial Dialogue System for Real-Time Evaluation of Unsu- pervised Dialogue Act Classifiers . . . . .</b>		<b>93</b>
10.1	System Design . . . . .	94
10.1.1	Tutorial Policies and Utterance Generation . . . . .	96
<b>Chapter 11 Evaluation Study I: In-Class Participants . . . . .</b>		<b>102</b>
11.1	Evaluating System Outcomes . . . . .	103
11.2	Exploring the Relationship Between Learner Personality, Attitudes, and Tutorial Dialogue Participation . . . . .	105
11.2.1	Analysis . . . . .	106
11.2.2	Modeling Level of Participation . . . . .	107
<b>Chapter 12 Evaluation Study II: Volunteer Participants . . . . .</b>		<b>112</b>
12.1	Learning Gains in JavaTutor-U and JavaTutor-S . . . . .	113
12.2	User Satisfaction in JavaTutor-U and JavaTutor-S . . . . .	114

12.3 Student Participation in JavaTutor-U and JavaTutor-S . . . . .	115
12.4 Incoming Student Characteristics . . . . .	117
12.5 Discussion . . . . .	118
<b>Chapter 13 Conclusion . . . . .</b>	<b>121</b>
13.1 Hypotheses Revisited . . . . .	121
13.2 Summary . . . . .	122
13.3 Threats to Validity . . . . .	123
13.4 Future Work . . . . .	123
13.5 Concluding Remarks . . . . .	124
<b>References . . . . .</b>	<b>125</b>
<b>Appendices . . . . .</b>	<b>137</b>
Appendix A Tutoring Tasks . . . . .	138
Appendix B Main Pre-survey . . . . .	141
Appendix C Lesson 1 Student Pre-test . . . . .	149
Appendix D Lesson 1 Student Post-test and Post-survey . . . . .	153
Appendix E User Satisfaction Survey . . . . .	156

## LIST OF TABLES

Table 3.1	Dialogue act tags from JavaTutor 2007 corpus with examples and student frequencies. . . . .	18
Table 3.2	Excerpt from the JavaTutor 2007 corpus. . . . .	19
Table 3.3	Student dialogue act tags and their frequencies for JavaTutor 2011 corpus. . .	19
Table 3.4	Excerpt of dialogue from the JavaTutor 2011 corpus and the task action that follows utterances. . . . .	21
Table 4.1	Original utterances, their processed versions and query combinations. . . . .	25
Table 4.2	Sample queries and their top three query-likelihood results. . . . .	26
Table 4.3	Query-likelihood clustering algorithm. . . . .	27
Table 4.4	Mean average precision (MAP) results for weighting interrogative parts of speech and punctuation. . . . .	27
Table 4.5	<i>b</i> -value MAP results. . . . .	28
Table 4.6	<i>n</i> -value MAP results with set <i>b</i> -value. . . . .	28
Table 4.7	Student utterance distributions over clusters using manual tags. . . . .	31
Table 4.8	Utterances from selected clusters with manual tags. . . . .	34
Table 5.1	Markov Random Field-based clustering algorithm. . . . .	40
Table 5.2	Example utterances from each cluster. . . . .	47
Table 6.1	Dialogue context features and their descriptions. . . . .	50
Table 6.2	Task features extracted from student computer programming activities. . . . .	51
Table 6.3	Test set accuracies and Kappa for the flat clustering model. . . . .	54
Table 6.4	The number of student utterances after branching on the previous tutor dialogue act. . . . .	55
Table 6.5	Test set accuracies and Kappa for branching on previous tutor dialogue act. .	56
Table 6.6	Accuracies for individual dialogue acts. Acts with fewer than 10 utterances after branching are omitted from the table. . . . .	58
Table 7.1	Excerpt of dialogue with a female student in the low self-efficacy and low skill perception groups. . . . .	60
Table 7.2	Excerpt of dialogue with a male student in the low self-efficacy and low skill perception groups. . . . .	61
Table 7.3	Average percentage of utterance length. . . . .	63
Table 7.4	Average test set accuracies for each learner characteristic. . . . .	68
Table 7.5	Selected utterances from clusters tailored to gender. . . . .	70
Table 7.6	Selected utterances from clusters tailored to skill perception. . . . .	71
Table 7.7	Selected utterances from clusters tailored to self-efficacy. . . . .	72
Table 8.1	Excerpt of dialogue from the corpus and the corresponding dialogue act tags. .	74
Table 8.2	Sample utterances that were correctly classified with the help of multimodal features and their incorrect classifications by the model that did not utilize posture and gesture. . . . .	84

Table 8.3	Sample utterances that were incorrectly classified when multimodal features were used but were correctly classified by the model that did not use posture and gesture features. . . . .	85
Table 9.1	Example student utterances and tutor responses. . . . .	90
Table 10.1	Sample tutor moves from the dialogue policy in the JavaTutor-U condition. . .	98
Table 10.2	Sample tutor moves for each dialogue act in the JavaTutor-S condition. . . . .	99
Table 10.3	Sample tutor moves for each hand-written rule. The rules are the same for both JavaTutor-U and JavaTutor-S. . . . .	100
Table 10.4	Regular expressions for capturing errors in student’s code and their corresponding tutor moves. . . . .	101
Table 11.1	Predictor variables from pre-survey and pre-test. . . . .	110
Table 11.2	Stepwise linear regression model for the number of utterances. . . . .	110
Table 11.3	Stepwise linear regression model for number of compile/run events. . . . .	111
Table 11.4	Stepwise linear regression model for number of tutor messages received. . . . .	111
Table 12.1	Results of the voluntary study for both versions of the system considering various metrics. . . . .	120

## LIST OF FIGURES

Figure 3.1	The tutorial dialogue interface of JavaTutor 2007 study. . . . .	17
Figure 3.2	The tutorial dialogue interface with four windows. . . . .	20
Figure 4.1	The query-likelihood clustering framework. . . . .	25
Figure 4.2	Accuracy results (%) for query-likelihood clustering, Rus et al. approach with 5 leading tokens and majority baseline. . . . .	30
Figure 5.1	The MRF-based clustering framework. . . . .	38
Figure 5.2	Comparison of query-likelihood clustering, MRF-based clustering, Rus et al. approach. . . . .	42
Figure 5.3	BIC values for varying number of clusters. . . . .	43
Figure 5.4	Confusion matrix for leave-one-student-out cross validation. . . . .	44
Figure 5.5	Distribution of dialogue acts over clusters in one of the folds. Bold face indicates the majority dialogue act in each cluster. . . . .	45
Figure 6.1	Intra-cluster distances with varying number of clusters. . . . .	52
Figure 6.2	Branching student utterances according to previous tutor dialogue act. . . . .	55
Figure 6.3	Confusion matrix for hierarchical model utilizing all features: T+D+L. . . . .	57
Figure 7.1	Distribution of students in the corpus. . . . .	61
Figure 7.2	Distributions of students with respect to learner characteristics (data points jittered to reveal overlap). . . . .	64
Figure 7.3	Leave-one-student-out test set accuracies for models by gender. . . . .	67
Figure 7.4	Leave-one-student-out test set accuracies for models by self-efficacy. . . . .	68
Figure 7.5	Leave-one-student-out test set accuracies for models by skill perception. . . . .	69
Figure 8.1	Output of the posture algorithm. . . . .	76
Figure 8.2	Output of the gesture algorithm showing the one-hand-to-face feature. . . . .	77
Figure 8.3	Output of the gesture algorithm showing the two-hands-to-face feature. . . . .	78
Figure 8.4	Branching student utterances according to previous tutor dialogue act and choosing which clustering group to use for unseen utterances. . . . .	79
Figure 8.5	Confusion matrix for the model <i>without</i> posture and gesture (61.8% accuracy). . . . .	81
Figure 8.6	Confusion matrix for the model <i>with</i> posture and gesture (67.05% accuracy). . . . .	82
Figure 9.1	Sum of squared errors graph. . . . .	87
Figure 9.2	Clusters from unsupervised dialogue act modeling and corresponding dialogue policy. . . . .	89
Figure 9.3	Evaluation framework structure. . . . .	89
Figure 10.1	Screenshot from the tutorial dialogue system. . . . .	94
Figure 10.2	System architecture diagram. . . . .	95
Figure 10.3	Sample tutor moves with sample code and utterance. [U] indicates the unsupervised dialogue act classifier move and [S] indicates the supervised classifier move. . . . .	97

Figure 11.1	Multivariate linear regression analyses for describing the outcomes of the system using measures from pre-survey and from tutorial interaction. . . . .	105
Figure 11.2	Scatter plots of various predictors and response variables. . . . .	108
Figure 12.1	Bar charts visualizing students' pre and post-test scores in the unsupervised condition. . . . .	114
Figure 12.2	Bar charts visualizing students' pre and post-test scores in the supervised condition. . . . .	115
Figure 12.3	Box plots showing learning gains for both conditions . . . . .	116
Figure 12.4	Box plots showing learning gains for both conditions considering the students who scored less than 75% of the questions correctly in the pre-test. . . . .	117
Figure 12.5	Box plots showing system usability scores for both conditions. . . . .	118
Figure 12.6	Box plots showing tutor feedback evaluation scores for both conditions. . . . .	119
Figure 12.7	Percentage of utterances binned with 5 utterance increments. . . . .	120
Figure 12.8	Percentage of compile/run events binned with 50 event increments. . . . .	120
Figure B.1	Computer Science Attitudes Instrument . . . . .	142
Figure B.2	New General Self-Efficacy Instrument (8 items) followed by Goal Orientation Instrument (12 items) . . . . .	143
Figure B.3	Big Five Inventory . . . . .	145
Figure B.4	Achievement Goals Questionnaire . . . . .	146
Figure B.5	Demographics Questionnaire . . . . .	148
Figure C.1	JavaTutor Lesson 1 pre-test. . . . .	152
Figure D.1	User Engagement Scale . . . . .	154
Figure D.2	User Engagement Scale continued . . . . .	155
Figure D.3	NASA-TLX workload index . . . . .	155
Figure E.1	Usability questions for the proposed tutorial dialogue system. . . . .	157

# Chapter 1

## Introduction

### 1.1 Motivation

One-on-one tutoring is highly effective for supporting student learning and generally results in higher learning gains than classroom instruction (Bloom, 1984; Chi, Siler, Jeong, Yamauchi, & Hausmann, 2001; P. A. Cohen, Kulik, & Kulik, 1982; VanLehn et al., 2007). In order to bring this high level of effectiveness in learning to a broader range of students, intelligent tutoring systems, adaptive systems that teach a predefined curriculum without requiring human tutors, have been an active area of investigation for several decades (Wenger, 1987). However, today's intelligent tutoring systems are not yet as effective as expert one-on-one tutoring (VanLehn, 2008). In order to enhance the performance of ITSs, one of the hypotheses under investigation involves enriching the interaction from questions and menu-based interaction to natural language communication (Graesser, Person, & Magliano, 1995). According to the natural language tutoring hypothesis, in order for intelligent systems to be as successful as human tutoring, it is necessary for ITSs to provide rich natural language dialogue with students. To this end, improving natural language interaction is crucial for development of intelligent tutoring systems.

Inspired by this hypothesis, tutorial dialogue systems, a subset of intelligent tutoring systems which provide rich natural language interaction, have become an active area of investigation in the artificial intelligence and education research communities. These systems mainly face two challenges in terms of improving natural language interactions: interpreting (understanding) user utterances and selecting system dialogue moves accordingly. This dissertation research focuses on interpreting user utterances so that the input provided to a system move selection module will be more accurate. Among several natural language understanding steps, identifying dialogue acts, which represent the communicative intentions of each utterance, constitutes the first and foremost representation (Allen et al., 1995; Core & Allen, 1997; Serafin & Di Eugenio, 2004; Traum, 1999; Stolcke et al., 2000). This level of representation is important for an automated

system to understand users and generate responses (Austin, 1975). For example, in a tutorial dialogue system, distinguishing whether the student is asking a question, requesting feedback or approval helps subsequent tutorial move selections which motivates the research on improving dialogue act classification.

Dialogue act classification has been studied extensively for decades. The approaches can be considered as two main groups: supervised and unsupervised modeling techniques. Supervised modeling builds on the pipeline of engineering a dialogue act annotation scheme, manually labeling all utterances in the training set according to the dialogue act taxonomy, building a classifier and using the classifier to label unseen utterances. In contrast, unsupervised modeling investigates groupings in the corpus without the dialogue act taxonomy engineering and labeling efforts. While supervised methods require long hours of engineering effort for designing and labeling corpora, unsupervised methods directly build on the corpus. Because dialogue act tags are not yet domain-independent, the need for engineering dialogue act taxonomies and tagging utterances rises each time a new corpus is to be studied. For fast development of tutorial dialogue systems, it is important to be able to process and model corpora quickly, which is the overarching goal of unsupervised dialogue act modeling.

In spite of the critical need for development of unsupervised techniques, these techniques have not yet been explored within today’s tutorial dialogue systems. This dissertation project is the first investigation of unsupervised dialogue act modeling frameworks and their evaluation within a fielded tutorial dialogue system. The overarching goal is to build effective unsupervised dialogue act modeling techniques that model student utterances successfully without needing manual annotations, which will help to develop tutorial dialogue systems more quickly and easily. Accomplishing this goal requires addressing numerous technical challenges, many of which have been addressed in this dissertation project as described later in this document. For example, a challenge arises regarding how to evaluate unsupervised models. Typically, manual tags are used to evaluate unsupervised models: by this metric unsupervised approaches achieve accuracy well below that achieved by supervised techniques. In order to address this weakness, several techniques and different feature sets have been utilized recently as presented in Chapters 4, 5, 6, 7, 8 of this dissertation; however, it is not clear that comparing against manual tags is the optimal approach for evaluating these models. Rather, an end-to-end evaluation such as the one presented here is needed within a fielded system.

## 1.2 Research Questions and Hypotheses

This project contributes to the goal of creating a data-driven tutorial dialogue system that relies on an unsupervised classifier for modeling student utterances. This dissertation’s central research question is: “**How can we derive automatic dialogue act classification models**

**that do not rely on manually labeled corpora, but that facilitate effective tutorial dialogue?”** To this end, the primary goal of this work is to investigate approaches to improve unsupervised dialogue act classifiers and to implement an unsupervised model in its intended usage environment (i.e., tutorial dialogue system). The first phase consists of modeling human-human tutoring data and building unsupervised dialogue act classification models. The second phase, which involves implementation of the tutorial dialogue system, data collection and analysis, evaluated an unsupervised dialogue act classifier in real-time and compared it to a supervised classifier. Data collection includes two versions of the system: one that relies on an unsupervised dialogue act classifier utilizing hierarchical clustering (JavaTutor-U) and one that uses a supervised dialogue act classifier utilizing a decision tree classifier (JavaTutor-S). These phases correspond to the hypotheses below. The hypotheses center around the notion that unsupervised dialogue act modeling will support learning and user satisfaction, while being substantially less time-consuming to build than supervised dialogue act models.

**Hypothesis 1:** A tutorial dialogue system that relies on unsupervised dialogue act models will support significant student learning.

**Hypothesis 1.1:** The learning gain, computed by the improvement of student’s knowledge from pre-test to a post-test, will be positive with JavaTutor-U.

**Hypothesis 1.2:** The learning gains of students who interact with JavaTutor-U will be higher than the learning gains of students who interact with JavaTutor-S. This hypothesis is motivated by the fact that the unsupervised models represent directly data-driven groupings of student utterances into the most natural clusters, rather than manually labeled tags that were created based upon dialogue taxonomies. Therefore, the tutor responses based on the clusters may be more contextually appropriate, resulting in higher learning gains than responses based on manual labels.

**Hypothesis 2:** A tutorial dialogue system that relies on unsupervised dialogue act models can provide students with a satisfying user experience.

**Hypothesis 2.1:** The user satisfaction obtained by post-survey will be positive with JavaTutor-U.

**Hypothesis 2.2:** The user satisfaction of students who interact with JavaTutor-U will be higher than the user satisfaction of students who interact with JavaTutor-S. This hypothesis is motivated by the same reasoning as Hypothesis 1.2 above.

## 1.3 Approach

This project utilizes a data-driven approach for facilitating improvement of tutorial dialogue systems. To this end, the following research phases were conducted to address the hypotheses above.

1. Extracted unsupervised dialogue act classifiers from human-human corpora of tutoring (Chapters 4, 5, 6, 7, 8)
2. Implemented two versions of a tutorial dialogue system that leverage two different dialogue act classifiers: the unsupervised dialogue act classifier and a baseline supervised classifier
3. Conducted an evaluation study with students using the implemented tutorial dialogue systems
4. Analyzed the resulting data to determine whether the evidence supports the hypotheses.

## 1.4 Contributions

The work reported in this dissertation has made the following novel contributions to the natural language processing community and the results are also of importance to the intelligent tutoring systems community<sup>1</sup>.

- **A novel dialogue act classification technique, *query-likelihood clustering***, showing the applicability of information retrieval methodologies to the dialogue act classification task (Ezen-Can & Boyer, 2013b). This approach utilizes query-likelihood modeling technique to retrieve similar utterances to a target utterance and utilizes the similarities for clustering purposes.
- **A novel dialogue act classification technique, *Markov random field-based clustering***, improving upon query-likelihood clustering by taking word orderings into account and further demonstrating the applicability of information retrieval methodologies to the dialogue act classification task (Ezen-Can & Boyer, 2015c).
- **Incorporation of task-context features** derived from complex task-oriented tutorial dialogue into unsupervised dialogue act classification (Ezen-Can & Boyer, 2014a). The model produced by this work is designed to take the complex task-oriented domain into account to more accurately model student utterances.

---

<sup>1</sup>This dissertation work builds upon the JavaTutor Project (NSF DRL-1007962) and leverages the JavaTutor learning environment built by a team including Bradford Mott, Eunyoung Ha, Christopher Mitchell, Joseph Grafsgaard, Alok Baikadi, Joseph Wiggins, and Megan Hardy Frankosky.

- **Novel representation of dialogue history** for dialogue act classification for incorporating dialogue-context features into modeling phase (Ezen-Can & Boyer, 2014a). This model groups student utterances in a more granular way according to the dialogue history so that the produced clusters are purer.
- **Results showing the role of learner characteristics** such as gender, self-efficacy and skill perception for adaptive dialogue act modeling (Ezen-Can & Boyer, 2014b, 2014c). The results demonstrate that learner characteristics are important factors affecting students' language choice.
- **Incorporation of multimodal features** derived from student postures and gestures during tutorial interaction into unsupervised dialogue act classification (Ezen-Can, Grafsgaard, Lester, & Boyer, 2015). This model is designed to utilize as many information sources as possible for the dialogue act classification task, including visual cues obtained from the tutoring session. The results show that posture and gesture features are strong predictors of dialogue acts.
- **Simulated evaluation** of unsupervised dialogue act models (Ezen-Can & Boyer, 2013a). The results of this study, showing that unsupervised dialogue act classifiers can be effective for tutor move selection, motivate the implementation of a tutorial dialogue system that relies on an unsupervised model.
- **First system to implement unsupervised dialogue act classification** within tutorial dialogue systems (Ezen-Can & Boyer, 2015b). The system built for evaluating an unsupervised dialogue act classifier within its intended usage environment utilizes the cluster choices of the model to select tutor moves. Two versions of the system, one relying on an unsupervised model and the other on a supervised model, allows us to compare different approaches in terms of their applicability and practicality for dialogue systems.
- **The relationship between student personalities/attitudes and the level of participation with the tutorial dialogue system** (Ezen-Can & Boyer, 2015a). The results of this work shows that incoming attitudes of students as well as their personalities are significantly predictive of their participation with a tutorial dialogue system.

In addition to these contributions, this dissertation research has produced results addressing its hypotheses. As detailed in Chapter 10, the two versions of the system were not statistically significant from each other in terms of student learning gain and satisfaction but that the version utilizing the unsupervised dialogue act classifier required substantially less manual effort to construct its dialogue management module. The results demonstrated that a tutorial dialogue system that relies on an unsupervised dialogue act classifier helps students learn significantly.

From a broader perspective, this dissertation has made significant improvements on both unsupervised dialogue act models and on evaluation of unsupervised techniques in general. The techniques presented in this dissertation improved the state-of-the-art in unsupervised dialogue act models up to  $\sim 70\%$  test-set accuracy. In addition, the system presented in this dissertation for evaluation of unsupervised dialogue act models is not only the first tutorial dialogue system relying on an unsupervised dialogue act classifier, but also the first dialogue system utilizing an unsupervised dialogue act model considering other domains. The results motivate further research on evaluation of unsupervised models in real-time and comparing them to their counterparts for the task that they are intended for.

## 1.5 Organization

The remainder of this document is structured as follows. Chapter 2 presents the background and related work in tutorial dialogue systems as well as dialogue act classification research within a broader set of dialogue systems. Chapter 3 describes the corpora utilized for learning and evaluating the dialogue act classifiers in this work. Chapters 4, 5, 6, 7 and 8 present unsupervised dialogue act classifiers built from task-oriented tutorial dialogue. Chapters 4 and 5 propose two novel clustering frameworks for grouping student utterances. Chapter 5 improves the technique presented in Chapter 4 by taking token orderings into account. Work presented in Chapter 6 enriches the feature sets used in dialogue act modeling with rich attributes derived from the task-oriented dialogue. To this end, dialogue history is represented in a novel structure and task-oriented features are incorporated into the unsupervised dialogue act modeling task. Chapter 6 utilizes longest-common-subsequence metric to calculate similarities between utterances, which proves to be successful and constitutes background for the other dialogue act classifiers presented in this dissertation. Chapter 7 incorporates learner characteristics such as gender, self-efficacy and skill perception for adapting dialogue act classification to users. Chapter 8 extends the work presented in Chapter 6 by incorporating multimodal features for dialogue act classification. Chapter 8 presents the best dialogue act classifier in this dissertation in terms of utilizing rich set of features and calculating utterance similarities efficiently. Chapter 9 describes preliminary experiments for evaluation of unsupervised dialogue act classifiers within their context. Chapter 10 details the tutorial dialogue system implemented for evaluating an unsupervised dialogue act classifier in real-time. Chapters 11 and 12 present the results of the two user studies for evaluation of the system, where Chapter 12 details the results of the most successful evaluation study. Finally, Chapter 13 revisits the hypotheses and presents conclusions and directions for future work. For a reader who wishes to focus on the most important pieces of this dissertation in terms of its scientific contribution, reading Chapters 3, 8, and 12 will provide a strong view of the data collection, unsupervised model building, and system evaluation work, respectively.

## Chapter 2

# Background and Related Work

This dissertation project falls at the intersection of two different fields: natural language dialogue and tutorial dialogue systems. Tutorial dialogue system research focuses on improving natural language interaction in intelligent tutoring systems so as to support learners more effectively. Section 2.1 presents background from this literature by providing an overview of existing tutorial dialogue systems to highlight that none of the tutorial dialogue systems currently rely on an unsupervised dialogue act classifier.

The second field this dissertation draws from is natural language dialogue, specifically dialogue act classification which is concerned with modeling user utterances according to their actions and intentions. Background from this literature is presented in Section 2.2, where the techniques are grouped as supervised (Subsection 2.2.1) and unsupervised (Subsection 2.2.2). These sections highlight existing techniques that can be further improved to increase unsupervised dialogue act classifiers' performance.

### 2.1 Tutorial Dialogue Systems

One-on-one tutoring is shown to be more effective than classroom instruction (Bloom, 1984; P. A. Cohen et al., 1982). Researchers have been working on understanding the factors that make one-on-one tutoring so effective. It is known that there are several factors affecting the effectiveness of tutoring. One factor that is believed to be particularly influential is interactivity (VanLehn, 2008; Chi et al., 2001). Tutorial interaction is highly valuable, and in particular *tutorial dialogue* has been found to affect tutorial effectiveness in many ways. It allows students to engage in dialogue during tutoring and to show uncertainty, which is found to be related to tutorial effectiveness (Litman, Moore, Dzikovska, & Farrow, 2010). Additionally, considering student moves such as disengagement (Forbes-Riley & Litman, 2013) and affective states (Forbes-Riley & Litman, 2012) as well as student utterances demonstrating reasoning (Forbes-Riley

& Litman, 2006a) have also been found associated with learning. Isolating the tutor-student interactivity showed that effectiveness of tutoring highly depends on the interaction effect between collaborators (Chi et al., 2001). For cases when student’s background knowledge does not match the expected target audience of learning materials, tutorial dialogue has been found to be even more effective (VanLehn et al., 2007). Tutorial dialogue captures rich information about student’s learnings and affective states. Today, most intelligent tutoring systems (ITSs) do not support tutor-student dialogue. Only a small portion of ITSs support natural language interaction and today’s systems are not yet as effective as expert one-on-one tutoring (VanLehn, 2008). Motivated by the proven contribution of tutorial dialogue to tutorial effectiveness, there is a growing body of work in improving natural language interaction in tutorial dialogue systems.

In order to provide the effectiveness of one-on-one tutoring within intelligent tutoring systems, tutorial dialogue has been worked on extensively. Studies showed that in tutorial instruction, a common pattern is formed consisting of five levels: tutor asking question, learner answering question, tutor giving short feedback on quality of the answer, collaboratively improving the answer, and tutor assessing learner’s understanding (Graesser et al., 1995). The last two steps involve natural language interaction, which is present in tutorial dialogue systems as opposed to ITSs without dialogue support.

Based on the last two steps of the common pattern (collaboratively improving the answer and tutor assessing learner’s understanding), natural language interaction has been focus of researchers. Improving natural language interaction is a challenging task due to the engineering effort required to build and integrate natural language dialogue components. To date, several tutorial dialogue systems have been built by manual efforts such as AutoTutor (Graesser, Chipman, Haynes, & Olney, 2005), Rimac (Jordan et al., 2013), Why2-Atlas (VanLehn et al., 2002), and ITSPoKE (Litman & Silliman, 2004) for teaching physics, DeepTutor for complex science topics in the classroom (Rus, D’Mello, Hu, & Graesser, 2013), iList for basic data structures and algorithms (Fossati, Di Eugenio, Brown, & Ohlsson, 2008), BeeDiff for symbolic differentiation problems (Callaway et al., 2007), ProPL for introductory programming problems (Lane & VanLehn, 2005), CycleTalk for thermodynamics (Rosé, Alevén, & Torrey, 2004), The Geometry Explanation Tutor for geometry (Alevén, Popescu, & Koedinger, 2001) and Circsim-Tutor for the human circulatory system (Evens et al., 1997). In the following, these tutorial dialogue systems are explained.

***AutoTutor.*** AutoTutor is a tutorial dialogue system that was adapted to several domains including computer literacy and physics (Nye, Graesser, & Hu, 2014). The system interacts with the students in rich natural language through an animated talking head and considers the problem of helping students generate correct explanations to solve a problem more important than remedying students’ misconceptions. The choice of prioritizing students’ explanations higher than misconceptions is based on the goal of making the system less domain-dependent.

The natural language generation module relies on a set of correct answers (expectations) and frequently expressed invalid answers (misconceptions) to evaluate how correct each student explanation is. Semantic pattern matching (Latent Semantic Analysis (Landauer & Dumais, 1997)) is conducted to measure correctness of explanations and pattern completion is used to provide responses that the students need to fill in missing information. The semantic matching algorithm combines Latent Semantic Analysis technique, regular expressions and word overlaps. The addition of regular expressions was intended to overcome the challenges produced by LSA (i.e., word ordering and negations). Two high-level goals are conducted through the tutor discourse: eliciting information from the student or delivering. AutoTutor has several move types (*main question, pump, hint, prompt, short feedback, connection, assertion, answer, summary*) that determine tutor's discourse. Based on the move types, production rules are used to drive dialogue.

**Why2-Atlas.** Why2-Atlas is a tutoring system that teaches qualitative physics by having students write explanations of their ideas in paragraph format (VanLehn et al., 2002). Then the system converts students' essays into proofs. Why2-Atlas and AutoTutor research groups collaborated on trying different natural language processing techniques where the Why2-Atlas version is based on deep syntactic analysis and Why2-AutoTutor is based on Latent Semantic Analysis. Similar to AutoTutor, if Why2-Atlas identifies any misconceptions in the essay, remedial dialogue takes place and the system asks the student to correct the essay.

The natural language module behind the system is composed of four modules. The first module is the sentence-level understanding module which processes student's essay and converts it into first-order logic by using lexical preprocessing to stem words. The module parses input to create syntactic trees and uses a Bayesian bag-of-words classifier to compute most likely classifications. The second module is discourse level understanding module that turns the logical form into a proof. Then the tutorial strategist module is run, which finds flaws in the proof to remedy. The dialogue engine module uses knowledge construction dialogues to remedy misconceptions found by the tutorial strategist module. The knowledge construction dialogue module builds on finite state networks where each node is a question asked to the student and each link are students' responses to questions. If nodes in the finite state machine are exhausted, the system elicits the solution to the student.

**ITSPOKE.** ITSPOKE is a spoken tutorial dialogue system for teaching physics that builds on Why2-Atlas text-based tutoring system (Litman & Silliman, 2004). The system engages in dialogue by assessing student's essays answering physics problems and aims to elicit more correct explanations while asking the student to correct any misconceptions by rewriting essays. Tutoring/essay revision rounds continue until either all misconceptions are resolved or the strategies of the system are exhausted. Student speech is transcribed from a microphone input by speech synthesizer and is sent to the Why2-Atlas back-end for semantic and syntactic

analysis. Then the text response produced by Why2-Atlas is sent to the text-to-speech system. Several findings have been published from ITSPOKE system regarding students' affective states (Forbes-Riley & Litman, 2006b; Forbes-Riley, Litman, & Rotaru, 2008; Forbes-Riley & Litman, 2009a).

**Andes.** The Andes physics tutoring system aims at increasing learning gains by improving problem-solving support for homework problems (VanLehn et al., 2005). Andes asks students to enter whole derivation rather than only the answer. By providing immediate feedback in each step of the problem solution rather than only at the end, the system achieves higher success than systems that take only the answer into account (VanLehn et al., 2005). Immediate feedback is not necessarily a tutor utterance, but includes coloring student's responses into green for correct and red for incorrect. There are three types of help that the system provides: error messages for cases probably due to lack of attention rather than lack of knowledge, explaining the error in a red entry and helping with the next step to do. The feedback moves are designed utilizing templates, leaving dynamic parts such as variable names and values to be filled in real time.

**BEETLE-II.** The BEETLE-II tutorial dialogue system teaches introductory electric and electronics concepts while aiming to make the system generalizable to different domains (Dzikovska, Steinhauer, Farrow, Moore, & Campbell, 2014). The system utilizes both domain-independent and domain-dependent components for interpreting student utterances. The pipeline of interpretation starts with a parser, which produces a domain-independent semantic representation. The output of the parser is combined with a contextual interpreter and an ontology mapping mechanism to turn the reasoning into a domain-specific representation showing objects and relations between them. The domain-specific information is stored in a knowledge base consisting of 14 objects. The student explanations are evaluated in terms of two criteria: factual correctness, measuring if the student's explanation is factually correct, and explanation correctness, evaluating if the explanation helps to solve the problem. The factual correctness category utilizes the knowledge base whereas the explanation correctness category makes use of heuristic matching algorithm.

Once the interpretation phase is completed, one or more tutorial strategies are chosen using an information-state update approach. The tutorial strategies include acknowledging correct part of student's answer, suggesting reading material, prompting for missing parts of the answer, hinting and eliciting the answer (Di Eugenio, Xie, & Serafin, 2010). Utilizing the decision of the tutorial planner, dynamic feedback generation takes place. The natural language generator uses a domain-specific sentence planner.

**CIRCSIM-Tutor.** The CIRCSIM-Tutor engages in dialogue with students to teach cardiovascular physiology (Evens et al., 1997). To interpret user's utterances, a parser and lexical-functional grammar annotations are used. An ontology determines if the student's answer is within the set of possible answers. In the latest versions of the system, information extraction

techniques were used for input understander (Glass & Evens, 2008). The spelling errors are corrected from student's answer and abbreviations are expanded. Similar to several other systems, CIRCISM-Tutor also uses a finite state machine for planning next steps in the dialogue. A planner module chooses an action and calls discourse planner. Following the planner comes the text generation module that takes the logic forms produced by the planner and produces sentences by filling abstract templates. The system assumes that students answer the most recent tutor question and extracts information taking this information into account.

**Rimac.** Rimac Tutor is a new natural language tutoring system that teaches physics concepts to high school students (Jordan et al., 2013). Before students interact with the system, a video is shown about a worked-out solution. Then, the students iteratively engage in 'reflective dialogues' with the system. The dialogues are system-initiative and system asks short answer questions throughout the dialogue. The system utilizes decision rules to simulate human tutoring (Glass & Evens, 2015).

**CycleTalk.** The CycleTalk system provides a simulation-based exploratory learning environment called CyclePad for teaching thermodynamic cycles (Rosé, Kumar, Alevin, Robinson, & Wu, 2006; Rosé et al., 2004). The system aims for students to engage in discovery-based simulation where they can make inferences based on their observations. The desired behavior of the system was determined by analyzing a Wizard-of-Oz study with a human tutor. The tutorial dialogue system builds on knowledge construction dialogues.

**ProPL.** The ProPL dialogue-based tutoring system taught computer programming for novices by eliciting program design ideas/pseudocode from the student prior to letting them attempt the solution (Lane & VanLehn, 2005, 2004). The students were asked to identify their programming goals and explain their reasonings for programming problems. The system utilized the Atlas dialogue management system that handled sentence-level understanding and dialogue planning. The system considered student solutions as goals (objectives to solve the programming problem), schemas (methods followed to achieve goals) and objects (the data components to be utilized in the solution).

For understanding student language, a corpus was analyzed and a 5-step student response was identified that helped interpretation (Lane & VanLehn, 2004). With the help of knowledge construction dialogues, the system requested explanation from students about their answers.

**BeeDiff.** The BeeDiff tutoring system helps students solve symbolic differentiation questions (Callaway et al., 2007). Students can ask questions while solving problems. Its predecessor is BEETLE tutoring system, which employs the same techniques for user understanding and generating responses. The researchers aim to make the system adaptable to different domains by using a domain-independent TRIPS dialogue parser and then mapping the output of the parser to domain-specific reasoning with the help of ontologies. Publications of BeeDiff showed that the adaptive version of the system outperforms the hand-authored non-adaptive system

(Callaway et al., 2007).

***iList***. The iList intelligent tutoring system teaches linked lists to college-level computer science classes (Fossati et al., 2008). The system employs a constraint-based modeling approach where domain knowledge is modeled with a set of constraints and the logic units have relevance conditions and satisfaction conditions. Student solutions are matched to constraints: correct ones to satisfied constraints and others as violated constraints.

***The Geometry Explanation Tutor***. The Geometry Explanation Tutor builds on the idea that students learn better when they provide explanations for their reasoning and applies this idea to the domain of geometry problem solving (Aleven et al., 2001). Student explanations are matched to a predefined set of correct and partially correct explanations. If this knowledge-based method fails, a statistical classifier is used to model student’s response. The system provides feedback on incomplete or incorrect parts of student solutions and engages in dialogue for students to improve their explanations.

All of the tutorial dialogue systems above aim to enrich students’ experience and achieve higher learning gains through rich natural language interaction. However, because these systems are very labor-intensive due to their domain dependent components, it is challenging and time consuming to improve their performance. As such, tutorial dialogue systems have not yet achieved effect sizes of expert human tutors (Bloom, 1984). The work in this dissertation addresses this challenge by proposing to use fully data-driven unsupervised dialogue act classifiers for understanding student language so that researchers would not have to re-engineer user understanding modules again for each and every different tutorial dialogue system.

## 2.2 Dialogue Act Classification

For natural language dialogue systems, interpreting user input is a key challenge. There are several components involved in this process. One component is *dialogue act classification*. The idea that human conversation contains *dialogue acts* originated with sociolinguistic theorists (Austin, 1975; Searle, 1969). Dialogue act theory suggests that humans not only communicate factual information within natural language utterances, they often express underlying intended action (e.g., to ask a question, to give a command). The practical value of dialogue acts for computational linguistics has been well demonstrated, with an extensive literature on automated dialogue act classification approaches (Traum, 1999; Shriberg et al., 1998; Stolcke et al., 2000). This step is crucial in dialogue systems in that, the more accurate user understanding is, the better responses are generated. The following two sections provide a summary with respect to the type of data mining technique used: supervised and unsupervised.

### 2.2.1 Supervised Dialogue Act Classification

Supervised dialogue classification involves creating a dialogue act taxonomy that fits the research goals of the project and manually labeling each utterance in the corpus to build a classifier that predicts dialogue acts of unseen utterances. This process is labor-intensive as it consists of analyzing the corpus, engineering a taxonomy and implementing it for every time a new corpus is to be modeled. Researchers have extensively studied this pipeline, proposing several techniques. These techniques can be grouped in two categories: sequential approaches and vector-based approaches. Sequential approaches rely on the idea of modeling natural language in the form of a Markov chain, assuming that each observation depends on previous observations (Stolcke et al., 2000). The most widely used application of sequential approaches is Hidden Markov Models (Stolcke et al., 2000; Boyer et al., 2010). Extension of latent semantic analysis with features (Serafin & Di Eugenio, 2004), Maximum Entropy models (Rangarajan Sridhar, Bangalore, & Narayanan, 2009), conditional random fields (Quarteroni, Ivanov, & Riccardi, 2011), decision trees (Shriberg et al., 1998) and support vector machines (Sadohara et al., 2013) are some of the methods proposed by researchers for supervised dialogue act classification.

In addition to the techniques utilized, supervised approaches for dialogue act classification aimed at improving performance by using several features as well, including dialogue structure such as position of the turn (Ferschke, Gurevych, & Chebotar, 2012), speaker of an utterance (Tavafi, Mehdad, Joty, Carenini, & Ng, 2013), previous dialogue acts (Kim, Cavedon, & Baldwin, 2010), lexical features such as words (Stolcke et al., 2000), syntactic features including part-of-speech tags (Bangalore, Di Fabbri, & Stent, 2008; Marineau et al., 2000), task-subtask structure (Boyer et al., 2010), acoustic and prosodic cues (Rangarajan Sridhar et al., 2009; Jurafsky, Shriberg, Fox, & Curl, 1998), and body posture (Ha, Grafsgaard, Mitchell, Boyer, & Lester, 2012).

But all of these techniques require engineering a dialogue act taxonomy for each corpus, manually annotating every utterance in the corpus and training a classifier. This pipeline constitutes a bottleneck for development of automated systems. To address this challenge and to reduce the labor-intensive tasks, unsupervised modeling techniques are the focus of the proposed research.

### 2.2.2 Unsupervised Dialogue Act Classification

Following the line of investigation into supervised approaches, recent years have witnessed a growing body of work in unsupervised dialogue act modeling. Ritter et al. (2010) utilized Hidden Markov Models (HMMs) with topic information for modeling Twitter conversations. They separated content words (topic words) with the help of a Latent Dirichlet Allocation framework. The findings of Ritter and colleagues suggest that interpretable set of dialogue

acts can be generated for open-topic conversations. Other research has followed this direction by proposing a variation of HMM for asynchronous conversations such as e-mails and forums, defining dialogue act emission distributions as mixture models and adding dialogue structure features (Joty, Carenini, & Lin, 2011). Joty and colleagues showed that taking the conversational structure into account leads to more accurate learning of sequence dependencies.

Dirichlet Process Mixture Models with a non-parametric Bayesian approach for train fares and timetables have also been explored (Crook, Granell, & Pulman, 2009). Crook and colleagues modeled utterances in three levels, from few utterances to more utterances in each class. A subsequent improvement on that work used a hierarchical Dirichlet Process with Hidden Markov Models for extracting semantics from utterances (D. Lee, Jeong, Kim, Ryu, & Lee, 2013). Lee and colleagues used a three-step approach: dialogue act, intent and slot entity recognition applied on spoken language. Similar to Ritter and colleagues, Lee et al. assume that each word is generated by one of three sources: words in the current dialogue act, general words and domain words.

Another non-parametric Bayesian method, infinite HMM, has been explored within a Japanese discussion domain, and the results were compared with those obtained using the Chinese Restaurant Process showing that the infinite HMM performed better in terms of purity and F-measure (Higashinaka et al., 2011). The authors underline an important distinction in corpora and divide corpora into two areas: human-human corpora and human-machine corpora. They state the difficulty of modeling human-human dialogue due to its unpredictable nature, which is also the case for this dissertation in tutorial dialogue systems. There have also been attempts at clustering dialogue acts on educational corpora using  $k$ -means (Rus, Moldovan, Niraula, & Graesser, 2012) as well as combining query-likelihood with clustering (Ezen-Can & Boyer, 2013b). Our work and the work of Rus et al. showed that mining educational data yields valuable information in terms of student behavior which can be utilized for developing educational systems.

From the point of features used for modeling, a subset of features utilized in supervised models have also been used in unsupervised models, such as words (Crook et al., 2009), state transition probabilities in Markov models (D. Lee et al., 2013), topic words (Ritter, Cherry, & Dolan, 2010), function words (Ezen-Can & Boyer, 2013b), a smaller subset of words containing the beginning portions of utterances (Rus et al., 2012), part-of-speech tags and dependency trees (Joty et al., 2011).

Overall, the prior unsupervised approaches have substantially underperformed current supervised approaches, which have been observed to achieve higher than 75% test set accuracy (Bangalore et al., 2008; Forbes-Riley & Litman, 2005; Serafin & Di Eugenio, 2004). Unsupervised approaches, while evaluated with varying metrics, typically achieve 58% F-measure (D. Lee et al., 2013), 42% training set accuracy (Rus et al., 2012) and 79% accuracy with a 70% baseline (Joty

et al., 2011). This dissertation project represents a confluence of research on tutorial dialogue and unsupervised dialogue act modeling for advancing the performance of unsupervised dialogue act classifiers and rapid development of tutorial dialogue systems.

## Chapter 3

# Corpora

This dissertation focuses on building unsupervised dialogue act classifiers from human-human tutoring corpora in order to use those models in real-time tutorial dialogue systems. In this chapter, two corpora (which we will refer to as JavaTutor 2007 and JavaTutor 2011) will be described. These corpora were collected in previous work (Boyer et al., 2010; Ha et al., 2012) within human-human tutoring studies for teaching the Java programming language. This dissertation uses them to learn the dialogue act classifiers. Both the JavaTutor 2007 and 2011 corpus reflect tutor-student natural language interaction via a computer-mediated online environment, as detailed below.

### 3.1 JavaTutor 2007 Corpus

The JavaTutor 2007 corpus consists of dialogues collected between pairs of tutors and students collaborating on the task of solving a programming problem as part of the JavaTutor project during spring 2007. The tutor and student interacted remotely with textual dialogue through computer interfaces (see screenshot in Figure 3.1). There were forty-three dialogues in total, with 1,525 student utterances (averaging 7.54 words per utterance) and 3,332 tutor utterances (averaging 9.04 words per utterance). This dissertation work focuses on classifying the dialogue acts of student utterances only because within an automated tutoring system, tutor utterances are system-generated and their dialogue acts are therefore known.

The corpus was manually segmented and annotated with dialogue acts, one dialogue act per utterance, during prior research that focused on supervised dialogue act annotation and dialogue structure modeling (Boyer, Phillips, et al., 2011). While the manual dialogue act labels are not used in model training within the current dissertation research, these tags are used to evaluate the models produced by the novel unsupervised approaches. Table 3.1 shows manually labeled tags and their frequencies. The Cohen’s Kappa for agreement on these manual tags was

0.76. An excerpt from the corpus is presented in Table 3.2.

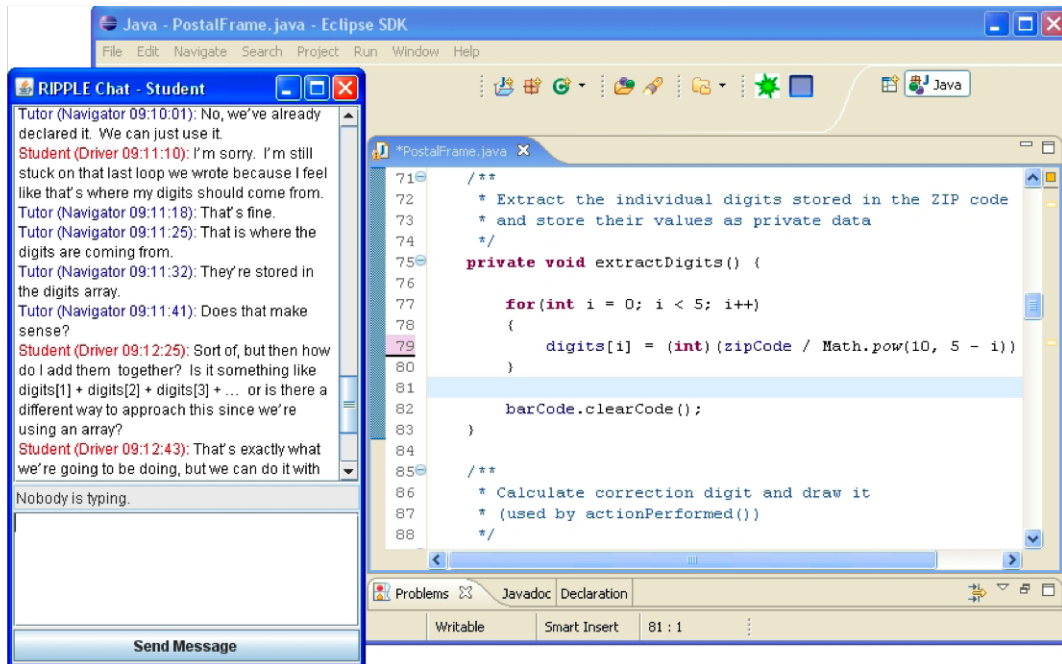


Figure 3.1: The tutorial dialogue interface of JavaTutor 2007 study.

## 3.2 JavaTutor 2011 Corpus

The JavaTutor 2011 corpus was collected in a computer-mediated human tutorial dialogue study. Students ( $n = 42$ ) and tutors interacted through textual dialogue within an online learning environment for introductory Java programming (Ha et al., 2012). The students were novices, never having programmed in Java previously. The tutorial dialogue interface consisted of four windows, one describing the learning task, another where students wrote programming code, beneath that the output of either compiling or executing the program, and finally the textual dialogue window (Figure 3.2).

As students and tutors interacted through this interface, all dialogue messages and keystroke-level task events were logged to a database. Only students could compose, compile, and execute the code, so task actions represent student actions while dialogue messages were composed by both participants. The corpus contains six lessons for each student-tutor pair, of which only the first lesson was annotated with dialogue act tags for a subset of 21 students ( $\kappa=0.80$ ).

Table 3.1: Dialogue act tags from JavaTutor 2007 corpus with examples and student frequencies.

<b>Tag</b>	<b>Act</b>	<b>Description</b>	<b>Freq.</b>
Q	Question	A general question which is not specific to the task	276
EQ	Evaluation	Question A question about the task	416
S	Statement	A statement of fact	211
G	Grounding	Acknowledgement of previous utterance	192
EX	Extra-Domain	Any utterance that is not related to the task	133
PF	Positive Feedback	Positive assessment of knowledge or task	116
NF	Negative Feedback	Negative assessment of knowledge or task	92
LF	Lukewarm Feedback	Assessment having both positive and negative assessments	32
GRE	Greeting	Greeting words	57

This annotated set contains 2,564 utterances (1,777 tutor and 787 student). The average number of utterances (both tutor and student) per tutoring session was 122 (min = 74, max = 201). The average number of tutor utterances per session is 84.6 (min=51, max=137) whereas for students it is 37.4 (min=22, max=64) for the annotated set. The average number of words per utterance for students is 4.4 and for tutors it is 5.4. This annotated set is used in the current analysis for both training and testing where cross-validation is applied. As described later, a separate set containing 462 un-annotated utterances is used as a development set for determining the number of clusters.

The dialogue stream of this corpus was manually annotated as part of previous work on supervised dialogue act modeling (Ha et al., 2012). A brief description of the student dialogue act tags is shown in Table 3.3. In the current work, these manually applied dialogue act labels are not utilized during model training, but are only used for evaluation purposes as our models' accuracies are reported for manual tags on a held-out test set.

An excerpt from the corpus is shown in Table 3.4.

Table 3.2: Excerpt from the JavaTutor 2007 corpus.

<b>Student:</b> so obviously here im going to read into the array list and pull what we have in the list so i can do my calculations [S]
<b>Tutor:</b> something like that, yes [LF]
<b>Tutor:</b> by the way, an array list (or ArrayList) is something different in Java. this is just an array. [S]
<b>Student:</b> ok [G]
<b>Student:</b> im sorry i just refer to it as a list because thats what it reminds me it does [S]
<b>Student:</b> stores values inside a listbox(invisible) [S]
<b>Tutor:</b> that's fine [EX]
<b>Tutor:</b> ok, so what are we doing here? [EQ]
<b>Student:</b> im not sure how to read into the array [NF]

Table 3.3: Student dialogue act tags and their frequencies for JavaTutor 2011 corpus.

<b>Student Dialogue Act</b>	<b>Distribution</b>
Answer (A)	39.85
Acknowledgement (ACK)	21.31
Sstatement (S)	21.20
Question (Q)	15.15
Request for Feedback (RF)	0.98
Clarification (C)	0.79
Other (O)	0.61

The screenshot displays the JavaTutor interface with four distinct windows:

- TASK WINDOW:** Contains an assignment instruction: "Now your game needs to get and store the player's latest choice (3 or 4). But remember, your program must not 'forget' the player's previous choice (1 or 2), because the newest scene you output will depend on both choices." It also includes a code example: `leftVar = rightVar;` and a task description: "Task 4 of 9: Without writing any code, make a plan with your tutor to store the player's latest choice. (Hint: you will need another variable.)"
- JAVA CODE WINDOW:** Shows a Java code snippet for a game:
 

```

3 String namelocation;
4 namelocation = "textastic";
5 System.out.println(namelocation);
6 String playername;
7 Scanner playerInput;
8 playerInput = new Scanner(System.in);
9 System.out.println("Enter your name here:");
10 playername = playerInput.nextLine();
11 System.out.println("Our hero's name is:" + playername);
12 System.out.println("You are standing in a field of corn.");
13 System.out.println("You can: 1. Look North, 2. Sit down.");
14 System.out.println("Please enter 1 or 2:");
15 int choiceone;
16 choiceone = playerInput.nextInt();
17 if(choiceone == 1) { System.out.println("Looking north you see a farmhouse."); System.out.p

```
- CHAT WINDOW:** A vertical chat area with messages from a tutor and a student. The tutor's messages include: "So I'm thinking I should make a choicetwoa and a choicetwob here", "Like those are going to be my new variables or something.", "Okay, so what would you store in those two new variables?", "choicetwoa would have the options if you had entered 1 for choiceone and choicetwob would have the options if you had entered 2 for choiceone", "Hmm, that's not bad, but you could store the player's second choice in just one new variable, regardless of what thre first choice was, right?", "Let's say that teh player chose 1 first", and "They still either choose 3 or 4 in the second choice".
- COMPILER OUTPUT WINDOW:** Shows the result of a compilation: "Compiled Successfully!".

Figure 3.2: The tutorial dialogue interface with four windows.

Table 3.4: Excerpt of dialogue from the JavaTutor 2011 corpus and the task action that follows utterances.

<p><b>Tutor:</b> ready? [<i>Q</i>] <b>Student:</b> yep [<i>A</i>] <i>Tutor moves on to next task</i> <b>Student:</b> cool [<i>S</i>] <i>Student compiles and runs the code.</i> <i>Program output: 'Hello World'</i> <b>Tutor:</b> excellent [<i>PF</i>] <b>Tutor:</b> add a space to make the output look prettier [<i>DIR</i>] <b>Student:</b> why doesnt it stop on the next line in this case? [<i>Q</i>] <i>Program halts</i> <b>Tutor:</b> it did [<i>A</i>] <i>Student runs the program successfully.</i> <b>Tutor:</b> good. [<i>PF</i>]</p>
--

## Chapter 4

# Dialogue Act Classification: Query-Likelihood Clustering

This dissertation project has investigated 5 different approaches to unsupervised dialogue act classification. This chapter and the following four chapters present these classification approaches.

This chapter presents a novel approach toward unsupervised dialogue act classification: *query-likelihood clustering* (Ezen-Can & Boyer, 2013b). This approach was inspired by an information retrieval technique, query likelihood, which is used in search engines for finding relevant documents for a given query (Manning, Raghavan, & Schütze, 2008). Applying this approach to dialogue act classification yields a three step process: first, identify utterances that are similar to a target dialogue utterance; second, use these similarities to cluster dialogue utterances; third, evaluate the model. This novel approach was evaluated on the JavaTutor 2007 corpus in Section 4.2.2.

How best to evaluate unsupervised techniques is an open research question since there is no “perfect” model that the results can be compared to. Therefore, two complementary evaluation criteria were examined, both of which have been used in prior work: quantitative evaluation with respect to manual labels (Crook et al., 2009; Rus et al., 2012), and detailed qualitative inspection of the clustering to determine whether it learned “natural” groupings of utterances (Ritter et al., 2010). The results demonstrate that query-likelihood clustering performs significantly better than majority baseline chance compared to manual labels. In addition, the proposed algorithm outperforms a recently reported unsupervised approach for speech act classification within a learning-centered corpus. Finally, qualitative analysis suggests that the clustering does group together many categories of utterances in an intuitive way, even highlighting in a fully data-driven fashion some ways in which the original hand-authored dialogue act taxonomy could be revised and improved in the future.

## 4.1 Query-Likelihood Clustering Methodology

This section describes the novel approach of adapting information retrieval (IR) techniques combined with clustering to the task of unsupervised dialogue act classification. IR is the process of searching available resources to retrieve results that are similar to the query (Ricardo & Ribeiro-Neto, 1999). In the proposed approach, the target utterance that is to be classified is used as a query and its similar utterances are gathered using query likelihood. Then, the query likelihood similarity scores are provided to the clustering technique.

### 4.1.1 Natural Language Preprocessing

At its core, query-likelihood information retrieval operates at the token, or word, level. In order to prepare the corpus for this application, several preliminary experiments were conducted to determine the appropriate type of preprocessing. It was observed that preprocessing is a crucial step in order to increase the discriminating cues extracted from the corpus. Part-of-speech (POS) tagging is a technique for labeling each word according to its grammatical part of speech such as noun, verb, and adjective. This procedure allows us to generalize words to their functionalities in sentences. For example, the pronouns “you” and “it” are grouped in the same POS tag: PRP standing for personal pronoun. The generalization provided by this part-of-speech tagging can be useful in dialogue act classification (Becker, Basu, & Vanderwende, 2012; Boyer et al., 2010; Di Eugenio et al., 2010). We experimented on querying with both actual words and with the more abstract POS tags. The best results were produced by a combination of words and POS tags. This hybrid approach replaces function words such as determiners (“the”, “a”), conjunctions (“and”, “but”), and prepositions (“in”, “after”) with their POS tags. Content words were retained but stemmed (e.g., “parameter” becomes “paramet”, “completely” becomes “complet”) to reduce the number of distinct words in the vocabulary of the corpus under consideration. This choice was motivated by the observation that in this task-oriented domain, important information about the dialogue act resides in content words. For instance, the word “confused” reveals important information about the state in which the student is and it is likely that the student might be requesting a hint. It was noted that in this domain of computer science tutoring, the natural language contains special characters that indicate a semantically important entity related to the domain, such as short bits of programming code. Although they are important with regard to the tutoring task, they require additional preprocessing in order to be handled appropriately by automated natural language processing techniques. Therefore, code segments in the corpus were replaced with meaningful tags representing them. For instance, segments about array indexing, which may originally have appeared as “ $x[i]$ ” and been mishandled, were replaced with the text “ARRAY INDEXING”. If-statements, loops and arithmetic operations were all replaced in the corpus using similar conventions. All procedures in natural language

processing is automated using parser therefore, the human-intervention was in deciding on the procedure to use (retain content words, replace function words) not in its implementation.

### 4.1.2 Query-Likelihood Representation

The query likelihood model treats each utterance as a document. The target utterance whose dialogue act is to be predicted becomes the query, and we apply information retrieval by querying the target utterance in the corpus. This query produces a ranked list of documents, from most likely to least likely, and this list is used to identify those utterances that are most similar to the target. The query likelihood implementation from the Lemur Project was used in this work (Strohman, Metzler, Turtle, & Croft, 2005). The ordering of words contains important information about the structure of utterances. For example, the word pair “I am” is mostly found in statements whereas if we regard them separately, “I” can belong to a question such as “What should I do next?” or “am” can be part of a evaluating question “Am I doing this correct?” (After preprocessing “I am” becomes “PRP (personal pronoun) VBP (present tense singular verb, non third person)” however, the same issue applies to the POS tags as well.) Because of the importance of word ordering on inferring the structure of utterances, we utilized a modified query approach that considers bigrams (pairs of adjacent words occurring in each utterance) rather than unigrams (individual words). Table 4.1 displays several original utterances, their modified forms after POS backoff and stemming, and the query combinations that were submitted to the algorithm. The POS tag VBZ represents third person present tense singular verbs, DT represents determiners, TO is the same as the word “to”, VBD stands for past tense verbs, WDT and WRB are interrogatives, and MD represents modal verbs. Table 4.2 presents two sample queries and top three most similar utterances retrieved by the query-likelihood model.

### 4.1.3 Clustering

The similarity results from querying were used as the distance metric in a  $k$ -means clustering algorithm. The implementation of this idea relies on creating binary vectors for similar utterances and then grouping those vectors. Each utterance that is present in the similarity list is represented as a 1, while the others are represented with a value of 0. In this way, each target utterance in the corpus is represented by a vector indicating the utterances that are similar to it. The entire unsupervised dialogue act classification algorithm is summarized in Table 4.3 and Figure 4.1 illustrates this process.

Table 4.1: Original utterances, their processed versions and query combinations.

Utterance	POS + stemming	Query combination
I'm reading it right now	VB read PRP right now	(VB read) (read PRP) (PRP right) (right now)
what is the basic structure to begin an array?	WDT VBZ DT basic structur TO begin DT array?	(WDT VBZ) (VBZ DT) (DT basic) (basic structur) (structur TO) (TO begin) (begin DT) (DT array) (array ?)
that was correct	WDT VBD correct	(WDT VBD) (VBD correct)
how do you think you should start it?	WRB do PRP think PRP MD start PRP?	(WRB do) (do PRP) (PRP think) (think PRP) (PRP MD) (PRP start) (start PRP) (PRP ?)

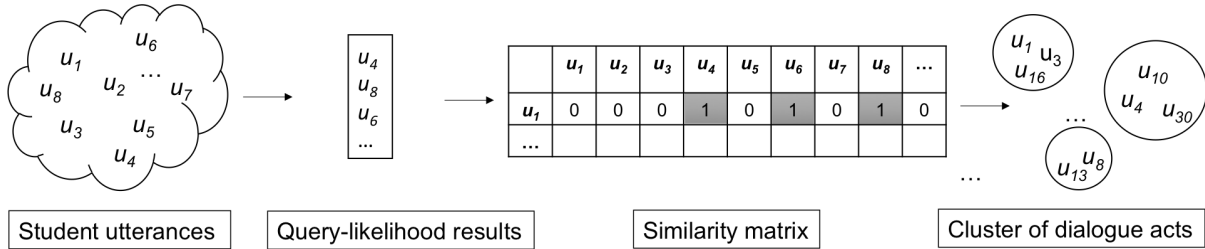


Figure 4.1: The query-likelihood clustering framework.

## 4.2 Experiments with Query-Likelihood Clustering

The goal of the experiments is to apply the novel unsupervised technique of query-likelihood clustering to discover student dialogue act clusters within the corpus of tutorial dialogue. We utilize a two-pronged evaluation consisting of quantitative comparison in terms of accuracy on manual labels, as well as qualitative examination of the resulting clusters. This section first presents the model-learning process, including parameter tuning on a development set, and then presents quantitative and qualitative evaluations on the remainder of the corpus. We also compare performance of the proposed approach to a state-of-the-art unsupervised technique for speech act labeling in a learning-centered corpus.

Table 4.2: Sample queries and their top three query-likelihood results.

Target utterance	Top three most similar student utterances
I am confused	<ul style="list-style-type: none"> <li>- here’s the part I am really confused on is this where I have to call up another class</li> <li>- and if so, then I guess I am sort of confused about how to retrieve the appropriate values from the table array</li> <li>- I’m confused on what I am going to put inside the loop</li> </ul>
how can I solve	<ul style="list-style-type: none"> <li>- how can I pull values out of an array or can I reference them with code like ARRAY_INDEXING</li> <li>- Is it like I have it on my screen And can I also set how long my array will be when I say private int PARAMETER</li> <li>-yea but i just can’t remember to how to use the METHOD_CALL to get each individual number</li> </ul>

### 4.2.1 Parameter Tuning

In order to train the unsupervised model, three parameters had to be set. Two of these parameters are used within the query phase and the last one applies to the clustering phase. The two parameters to be determined in the query phase are  $b$ , the blind relevance feedback threshold, and  $n$ , the number of top query likelihood results to be used while creating vectors for clustering. The parameter related to the clustering phase is the distance metric. In order to tune these parameters, a 25% validation set, constituting 10 randomly selected sessions, was drawn from the corpus. The parameter tuning was conducted in a sequential manner that allows the latter steps to use already fixed parameters.

**Token weighting.** Prior to tuning the parameters, an additional optional set of token weights was tuned for use during querying. This optional parameter was desirable based on observations that some POS tags should be weighted more than others for identifying similar results to a target utterance. For example, interrogatives (question words such as what, where, when, how, and why) and question marks are highly discriminating for question dialogue moves. Weights were learned for this subset of tokens using an incremental approach as shown in Table 4.4. First, the mean average precision values of query likelihood results without any weights were given. Then, weights for WDT (POS tag for what and which) were utilized within the experiments. Having determined the proper weight for WDT, different weights for WRB (POS tag for why, where, how, when) were tried. Finally, the weight for question marks was set.

**Relevance feedback threshold ( $b$ ).** In a typical query likelihood scenario, relevance feedback on the retrieved results is provided by human users and used to improve the model

Table 4.3: Query-likelihood clustering algorithm.

Let  $D$  be a corpus of utterances  $D = u_1, u_2, \dots, u_n$ . Then the goal is:  
 $\forall u_i \in D$ , identify  $l_i$  dialogue act label of  $u_i$   
 Procedure:  
 For each  $u_i$   
 1. Set target utterance  $q_i = u_i$   
 2. Let the query-likelihood result of utterance  $u + i$  be  $R = u_t, u_{t+1}, \dots, u_z$  such that  $R$  is the result of  $queryLikelihood(q_i)$   
 3. Create vector of query results indicator variables  $V_i = (v_1, v_2, \dots, v_j, \dots, v_n)$  such that  $v_j = 1$  if  $u_i \in R$  else  $v_j = 0$   
 Let the total vector be  $V_t = (V_1, V_2, \dots, V_n)$   
 Return clusters  $C = c_1, c_2, \dots, c_k$  such that  $C$  is the result of  $kmeans(V_T)$

Table 4.4: Mean average precision (MAP) results for weighting interrogative parts of speech and punctuation.

Weights	MAP
no weight	0.1239
WDT = 10	<b>0.2359</b>
WDT = 100	0.2326
WRB = 10	<b>0.2358</b>
WRB = 100	0.2339
“?” = 10	0.2457
“?” = 100	<b>0.2567</b>

performance. However, in a fully unsupervised scenario, human relevance feedback is not available. Our unsupervised approach utilizes pseudo-relevance feedback (blind relevance feedback), which assumes that the top  $b$  documents retrieved are relevant to the query (Salton & Buckley, 1990). Taking the top  $b$  documents into account, the algorithm automatically finds words that are important for those documents and therefore may be useful for the query. In order to find the important words, relevance models for each retrieved document in the ranked list are computed, where a relevance model is the probability of features used in the query given the document. In our experiments, the features are composed of bigrams, which are pairs of adjacent words within an utterance. Therefore, the relevance model of a document is the probability of its bigrams given the whole utterance. Then, relevance models of the top  $b$  documents are sorted and the top terms are determined, which are used to expand the original query. Having chosen those

words, the algorithm expands the initial query by appending the newly found terms and running the query again. This procedure of enriching the query with top relevant results is done in order to avoid sparse ranked lists. Like the other parameters described in this section,  $b$  was tuned on a development set. Table 4.5 shows the resulting best  $b$ -value of 30. This means that in the models reported here, the top 30 utterances in the initial ranked list returned by the query are assumed as relevant to the query and they are used to expand the initial query. This is intended to prevent sparse result sets.

Table 4.5:  $b$ -value MAP results.

<b>b</b>	<b>5</b>	<b>10</b>	<b>15</b>	<b>20</b>	<b>25</b>	<b>30</b>	<b>35</b>
<b>MAP</b>	0.343	0.34	0.342	0.346	0.351	<b>0.352</b>	0.351

**Top query threshold ( $n$ ).** Given the  $b$ -value, we moved on to tuning  $n$ -values, which represent the number of top query-likelihood results to be used in forming the similarity vectors for subsequent clustering. A higher  $n$  value will treat a larger number of utterances retrieved during querying as “similar” to the target. A minimum of  $n=5$  was chosen to sufficiently populate the vectors since the non-zero entries determine clusters, and we explored whether larger  $n$  values performed better; however, the optimal value was  $n=5$  as shown in Table 4.6.

Table 4.6:  $n$ -value MAP results with set  $b$ -value.

<b>n-value</b>	<b>5</b>	<b>10</b>	<b>15</b>	<b>20</b>	<b>30</b>	<b>100</b>
<b>MAP</b>	<b>0.517</b>	0.432	0.409	0.398	0.395	0.307

**Distance metric.** We experimented with multiple distance metrics to determine which metric performed best within this novel context. The candidate distance metrics included the widely used Euclidean and Manhattan distances. Also considered was another widely used similarity metric in text mining, cosine similarity, which increases as the two vectors have non-zero entries in the same positions. This approach is particularly effective for sparse vectors. The final candidate was Borda count (Jain, Nandakumar, & Ross, 2005), a metric that weights non-zero entries according to their ranks in query likelihood. In our scenario, the first utterance

retrieved by query likelihood gets the highest rank. This approach reforms vectors in a weighted manner, after which Euclidean distance is applied. Within the development set, cosine distance performed with highest accuracy at 43.43% compared to manual labels. Borda count achieved 42.63%, Euclidean distance 41.64%, and Manhattan distance 41.82%. Since cosine similarity is computed by the dot product of two vectors divided by the product of their magnitudes, it takes the size of vectors into account and provides a ratio of matching 1 values in the same position in both vectors. In this way, intersecting non-zero entries are valued with respect to the size of the vectors.

#### 4.2.2 Accuracy in Identifying Manually Labeled Dialogue Acts

Having tuned the parameter values using the development set, we trained a model on the remainder of the corpus. This unsupervised model did not take manual labels into account during training, but we evaluate its performance with respect to manual labels. Due to the nature of the unsupervised approach, the optimal number of clusters was not known. Therefore, we explored varying numbers of clusters using  $k$ -Means, a standard clustering algorithm that aims to cluster observations into  $k$  clusters so that each element has the highest similarity to every other element in the cluster (Kanungo et al., 2002). In addition to  $k$ -means clustering, we experimented with a non-parametric Bayesian approach used in recent unsupervised dialogue act classification work in a non-tutoring domain (Crook et al., 2009).

We calculate the accuracy of the approach for classification by comparing to manual labels. Figure 4.2 presents the accuracy of the proposed approach using the  $k$ -means clustering algorithm in Weka (Hall et al., 2009). To label a target utterance, the query likelihood results were retrieved for that utterance, and then its vector was provided to the clustering algorithm. The resulting cluster in which the target utterance resides was interpreted as the dialogue act label of the target utterance. The majority label of the cluster was taken as its dialogue act label. For comparison, the accuracy results are compared against the majority class baseline of 25.87%, Evaluation Question (EQ). This is the most commonly occurring student dialogue act in the corpus and therefore represents the expected accuracy of a model that performs equal to chance. Additionally, we compare our implementation against that of the recent approach of Rus et al. (Rus et al., 2012), which clusters utterances via word similarity using a specified number of leading tokens of each utterance.

The highest accuracy achieved by query-likelihood clustering was 41.84% with  $k=8$ . This accuracy therefore constitutes a 61.9% improvement over baseline chance. We experimented with the Rus et al. leading-tokens clustering using two to five leading tokens as they suggest. Five leading tokens performed best, yielding 34.90% accuracy at  $k=7$ . In order to provide a comparison across corpora, we consider the results from Rus et al. on their corpora of educational

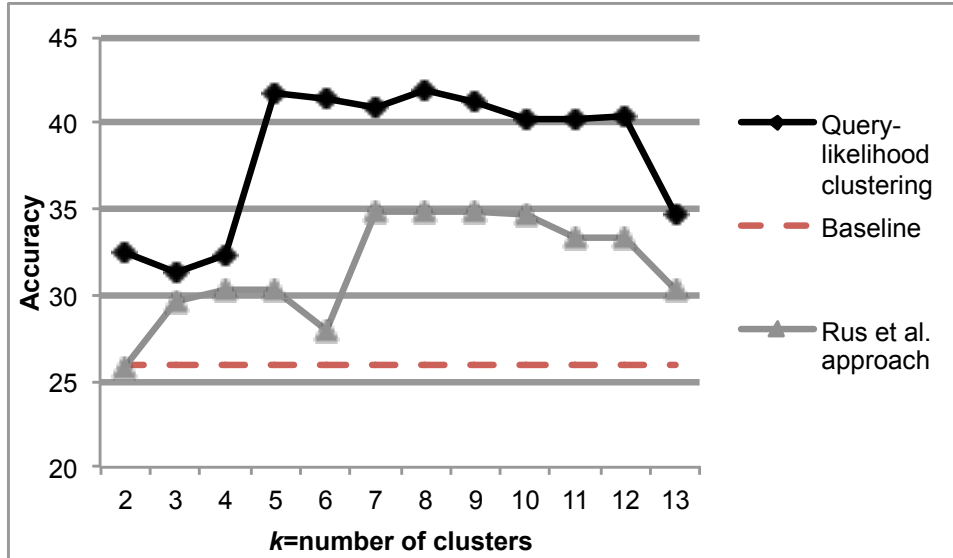


Figure 4.2: Accuracy results (%) for query-likelihood clustering, Rus et al. approach with 5 leading tokens and majority baseline.

games. Their highest accuracy was 37.9%, compared with a majority class baseline of 28.5% (statements). This is a 32.98% improvement over the majority baseline chance. Another algorithm tried by Rus et al. was Expectation Maximization, which achieved 37.9% accuracy on their combined corpus of educational games. This algorithm achieved its highest accuracy of 30.47% with four leading tokens on our corpus. We also experimented with Dirichlet process clustering as used by Crook et al. (Crook et al., 2009). Dirichlet process clustering gave substantially lower results on our corpus after natural language preprocessing, with an accuracy of 24.48% compared to the 41.84% of query-likelihood clustering.<sup>2</sup>

Finally, in order to evaluate the relative contribution of the query-likelihood clustering components of automatic natural language preprocessing (POS tagging, stemming) and vector enhancement with information retrieval, we performed two experiments omitting each of these components from the procedure. Parameters were re-tuned for each experiment utilizing the same development set split used earlier (25% of the corpus). With the best-performing model size of  $k=8$  clusters, omitting the natural language preprocessing step produced accuracy of 37.59% and omitting the information retrieval step produced accuracy of 35.68%, each of which is substantially lower than the query-likelihood clustering approach of 41.84%. However, these simpler approaches resulted in a smaller number of clusters for their best-fit models, achieving their highest accuracies of 41.06% and 40.45%, respectively, when  $k=4$ . This accuracy is only modestly lower than the query-likelihood clustering accuracy of 41.84%; however, a significant

<sup>2</sup>The Dirichlet clustering implementation from mahout.apache.org was used for this analysis.

drawback is that the number of clusters is much smaller than are typically distinguished by dialogue act classification schemes, and would likely not result in sufficiently fine-grained distinctions to support dialogue management.

### 4.2.3 Analysis of Clusters

As shown in Table 4.7, the approach was particularly successful in clustering negative feedback utterances (NF), evaluation questions (EQ) and groundings (G). 64.06% of all NF utterances are grouped in one cluster (Cluster 2), while 64.09% of EQ utterances are in one cluster (Cluster 5) and 94.74% of all G utterances are in another cluster (Cluster 1). The Q and EQ tagged utterances, which are structurally very similar in that they both are questions, were grouped into one cluster (Cluster 5), which constitutes 58.54% of all Q and EQ tagged utterances. In another cluster, 14.26% of all Q and EQ utterances were grouped together (Cluster 6).

Table 4.7: Student utterance distributions over clusters using manual tags.

Clusters/Tags	C1	C2	C3	C4	C5	C6	C7	C8
<i>GRE</i>	31	1	0	4	8	0	0	2
<i>EX</i>	22	21	5	16	8	0	1	21
<i>Q</i>	1	13	11	20	121	33	0	36
<i>PF</i>	57	16	6	7	0	1	0	5
<i>S</i>	11	32	<b>21</b>	<b>29</b>	5	0	0	<b>51</b>
<i>G</i>	<b>144</b>	0	0	1	0	0	6	1
<i>EQ</i>	1	18	17	14	<b>191</b>	<b>43</b>	0	14
<i>NF</i>	3	<b>41</b>	9	4	0	0	0	7
<i>LF</i>	1	15	2	3	0	0	0	1

In order to examine the structure of the clusters in more detail, Table 4.8 provides sample utterances from four of the eight clusters. The manual labels of the utterances are also given in parenthesis. Some clusters perform intuitively; for example, Cluster 1 is dominated by grounding, with many positive feedback utterances as well. These positive feedback and grounding utterances have similar surface forms such as “yeah”, “right”, and “oh”, and distinguishing them further will require modeling the broader dialogue structure (for example, a notion of dialogue history, as we explore in Chapter 6).

Cluster 2 is dominated by negative feedback (NF) with a large number of statements (S). This cluster captures the vast majority of negative feedback moves, indicating that these moves

are highly distinguishable. Most of the statements were in negative tone or expressing confusion. For example, the utterance “Im not sure if it is asking if the PARAMETER is how ever far you are away from NUMBER or the actual number you are away from NUMBER”, which is labeled as statement, shows confusion although it is a statement in terms of its intention.

Cluster 3 and Cluster 4 are highly impure. Closer inspection reveals that Cluster 3 mostly has broken utterances such as “correction digit”, “is in”, “digit” and some statements. Cluster 4 contains statements and extra domain utterances that are primarily statements. Moreover, there are some implicit questions such as “thats another thing I was going to ask am I just storing the values in METHOD\_NAME and sending them to PARAMETER” and “So I have to call upon the PARAMETER class and use a method in there right?”. Cluster 7 is highly sparse, containing only seven utterances. These tend to be highly similar in structure, such as “oh ok”, and “oh dear”. The very low distance between these utterances pulled them tightly into one cluster that was judged as distinct from the others.

Cluster 5 and Cluster 6 are dominated by evaluation questions (EQ), with a substantial number of general questions (Q) as well. The dominance of EQ in both clusters may relate to its high frequency within the corpus. However, these two question acts tend to exhibit close structural similarity although their roles are different (asking for feedback versus asking a general conceptual question). For example, the utterance “do i have to set it to a PARAMETER?” is a question; however, a deeper analysis shows that it is more specifically an evaluation question that requests feedback on the task.

The presence of impure and sparse clusters prompted an experiment to explore whether other models with similar but slightly lower overall accuracy would yield a more desirable clustering. Therefore, we explored using an information criterion, Hartigan’s rule of thumb, that utilizes the number of parameters in the model. This metric identified ten clusters as optimal, with a slightly lower accuracy (40.28%) compared to eight clusters, and the sparse clusters remained. We also experimented with the *X*-Means algorithm that utilizes Bayesian Information Criterion for splitting clusters (Pelleg & Moore, 2000), which resulted in four clusters with 36.72% accuracy.

### 4.3 Query-Likelihood Clustering: Discussion

A strength of unsupervised approaches is that because they do not rely on any manually engineered tagging schemes, they reflect the structure of the corpus in a fully data-driven way. In the case of query-likelihood cluster, the results highlight challenges of utilizing pedagogically driven manual dialogue act classification taxonomies within automated approaches.

While the proposed algorithm is promising in that it outperforms current unsupervised approaches for dialogue act modeling, it has several notable limitations. One limitation is algorithmic complexity, which is quadratic over the size of the corpus. This complexity is

inherent in the binary representation of each utterance as a vector with similarity to other utterances. Another limitation of the proposed approach arises with clustering algorithms in general, which is that a significant amount of human intelligence is often required to decide on the number of suitable clusters for the corpus. Nonparametric approaches to automatically identifying the number of clusters performed worse than parametric approaches in the current analyses; however, nonparametric approaches in general are an important area for future study. Additionally, the query-likelihood clustering approach does not consider higher-level dialogue structure; it clusters one utterance at a time. This limitation leads to trouble disambiguating utterances with similar surface features. A highly promising direction to address this limitation is to enhance the algorithm with structural features such as dialogue history as done in Chapter 6. Finally, the query-likelihood clustering approach treats all learners under the same dialogue act classification model, while it is likely that learner characteristics are highly influential for dialogue act classification. The next chapter explores this possibility.

Table 4.8: Utterances from selected clusters with manual tags.

<i>Cluster 1</i>
- right (G)
- ahh (G)
- ok (G)
- yeah (G)
- yes (PF)
- heheh yeah that would work (PF)
- I see that (PF)
- gotcha (EX)
- Yes Giving me definitions to various commands and such (EX)
- Ohh yes substantially (EX)
<i>Cluster 2</i>
- not really yet (NF)
- im not completely sure about how to do this (NF)
- the parsing im not sure about (NF)
- to be honest im not even sure what an array is (NF)
- im not sure how to read into the array (NF)
- I don't know how to do this (NF)
- and I know there is more to this line but I cannot remember the command (NF)
- Not yet (NF)
- im not so good at ARRAY just yet (NF)
- but I'm not exactly sure how to do that (NF)
- Im not sure if it is asking if the PARAMETER is how ever far you are away from NUMBER or the actual number you are away from NUMBER (S)
- I am asking how to do whatever drawing I need to do in the <i>METHOD_CALL</i> method (S)
<i>Cluster 5</i>
- So what's wrong with this? (Q)
- Can't manually turn an integer into a string? (Q)
- Then how would I incorporate that with the <i>METHOD_CALL</i> ? I think it's asking me to use that in some why but it's not supplying arguments to do so (Q)
- are we done extracting digits? (Q)
- how do i sum the digits? (Q)
- do i have to set it to a PARAMETER? (EQ)
- thats another thing I was going to ask am I just storing the values in <i>METHOD_NAME</i> and sending them to PARAMETER? (EQ)
- why is what I just highlighted underlined in red doesn't that mean its wrong? (EQ)
- does extracting have to do with <i>METHOD_NAME</i> ? or anything (EQ)
<i>Cluster 6</i>
- What is the next step? (Q)
- What do I write in it? (Q)
- what do i do first? (Q)
- so what can i do to fix what i was doing? (Q)
- does that look ok? (EQ)
- is this correct? (EQ)
- is this what i need to do? (EQ)

## Chapter 5

# Dialogue Act Classification: Markov Random Field Based Approach

This chapter presents a novel approach to unsupervised natural language dialogue modeling that builds on Chapter 4 and improves upon query-likelihood clustering by taking word orderings into account rather than assuming they are independent from each other (Ezen-Can & Boyer, 2015c). Leveraging highly effective techniques from the computational linguistics subfield of information retrieval, we propose a clustering approach based on a graph-based Markov Random Field (MRF) framework to group dialogue utterances with the same dialogue act in an unsupervised way, that is, without requiring the dialogue acts to be labeled manually ahead of time. This approach outperforms the previously reported approaches, including our earlier work on query-likelihood clustering for dialogue act modeling explained in Chapter 4 (Ezen-Can & Boyer, 2013b).

### 5.1 Unsupervised Dialogue Act Modeling

The novel contribution of the work in this chapter is the adaptation of information retrieval techniques combined with clustering for modeling student dialogue acts. In this section, the framework utilized for dialogue act classification is described. First we perform natural language preprocessing similar to the techniques described in Section 4.1.1, followed by the information retrieval techniques used to calculate utterance similarity and finally the clustering phase, which utilized calculated similarities to group utterances.

As described in Chapter 4, information retrieval is the process of searching available resources to retrieve results that are similar to the query (Ricardo & Ribeiro-Neto, 1999). Probabilistic models such as language models are commonly used for this purpose (Manning et al., 2008). For example, in document search, documents that are expected to be relevant to a query are sorted according to their relevance to the query based on a similarity score. We adapt this information

retrieval approach to compute similarity between utterances. This similarity information can then be used for clustering relevant utterances together, forming groups of utterances that we hypothesize represent the same dialogue act.

This section presents a novel approach to unsupervised dialogue act modeling: Markov Random Field based clustering. The approach that MRF-based clustering will be compared against is described in Chapter 4, which is also based on information retrieval techniques. Query-likelihood clustering treats each student utterance as a query and retrieves other similar utterances from the corpus. MRF-based clustering builds on this approach with one key difference: MRF takes word ordering into account, unlike prior unsupervised dialogue act modeling approaches (Metzler & Croft, 2005). We hypothesize that MRF will perform better than all prior techniques. The experiments described in the next section demonstrate this with empirical results.

In this chapter, the same natural language processing technique is utilized as described in Chapter 4.1.1. Having preprocessed the corpus, we move on to similarity calculations in Section 5.1.1 where we detail the MRF-based similarity calculations. Then, in Section 5.1.2 we describe the clustering.

### 5.1.1 Utterance Similarity Calculation

Having preprocessed the corpus, we now have a set of utterances that will be used for dialogue act classification. The next step is to calculate how similar these utterances are to each other so that we can cluster them. We report on the novel MRF-based model and compare it against query-likelihood modeling, both of which adapt information retrieval techniques as described in the following subsections.<sup>3</sup>

#### Query-Likelihood Model

We have previously reported on a technique to calculate similarities between utterances using query-likelihood (Ezen-Can & Boyer, 2013b) (Chapter 4), and we briefly describe this technique here for comparison. Given a query, a list of documents that are expected to be relevant to the query are returned. We adapt this information retrieval goal to our purposes where we focus on finding similar utterances instead of documents. The process can be summarized as follows: *given a target utterance, find a list of utterances that are similar to the target*. The similar utterances are obtained from our corpus of student utterances.

---

<sup>3</sup>The Lemur Project’s information retrieval implementation (Indri) was used in this work (Strohman et al., 2005).

## Markov Random Field Model

Chapter 4 described the query-likelihood technique. The new approach presented here explicitly models the token ordering within a graphical representation, MRF graphs. In this way, while finding similarities between two utterances, we aim to consider the sequential order of each token rather than using a bag-of-words approach. For example, the utterances “I am correct” and “am I correct” have higher similarity scores to each other using query-likelihood clustering than MRF because all tokens are the same without considering the token ordering. However, in this case a lower similarity is desirable as one of the utterances is a statement and the other is a question in terms of their communicative intentions. This distinction is the main motivation of MRF-based clustering. We hypothesize that if the similarities of utterances utilized in clustering technique are calculated more accurately by taking the token ordering into consideration, the clustering performance will improve as well. We therefore use a formulation of MRF that takes word ordering into account (Metzler & Croft, 2005).

MRF has been shown to be one of the best performing algorithms in the information retrieval community (Metzler & Croft, 2005). Therefore, proving its theoretical background is beyond the scope of this work. However, we would still like to provide motivation for using MRF for similarity calculation by discussing some differences with widely used metrics in the natural language processing literature. *Skip n-grams* are a generalization of *n-grams* where the words need not be consecutive, but there may be gaps within a window size. For example, if the window size is  $w$ , a bigram ( $n$ -gram with  $n=2$ ) will consider a sequence of four tokens with the window of 2 tokens changing in between. Although this technique is widely used, skip  $n$ -grams require a window size to be set before computation whereas MRF does not, which is a motivating factor for using MRF. In addition, the use of  $n$ -grams gets more computationally expensive as  $n$  increases. While representing  $n$ -grams as feature vectors, the length of the vector is given by the whole set of  $n$ -grams in the corpus, making the vector large and the effect of the non-zero values lower, whereas in MRF likelihood summation, the non-zero values do not affect the similarity calculation. This property of MRF enables the nonzero values to be given more importance as desired. Further, MRF does smoothing for frequently occurring tokens, taking into account more important and representative words.

Recall that the purpose of using MRFs is to identify similar utterances. Another way to do this would be *longest common subsequence* to compute the similarity of not necessarily contiguous sequences of words, as utilized in the analyses presented in Chapters 6 and Chapter 8. This technique is costly because all window sizes are computed, whereas it is possible to limit the window size (e.g.,  $w=2$ ) in MRF. In addition, it is possible to weight some values of  $w$  and  $n$  more than others in MRF by giving different weights to edges. The ordered tokens can be weighted more than single words ( $n \geq 2$  more than  $n = 1$ ) and phrases with one skipped token

weighed more than multiple skipped tokens ( $w = 1$  more than  $w \geq 2$ ), whereas it is very difficult to do weighting with the longest common subsequence metric. It would require weighting some of the columns in the feature vector compared to a better structured way provided by MRF. Motivated by the flexibility of MRF models, we apply this technique to calculate similarities between utterances.

First, MRF models are drawn and the probabilities obtained from the undirected graph serve as inputs to the clustering algorithm (i.e.,  $k$ -medoids clustering) as shown in Figure 5.1. Suppose  $y_i$  is an utterance that we would like to calculate the similarities to every other utterance  $u_j$  in the corpus. We draw an MRF model with  $u_j$  being the root and  $t_1 \dots t_k \dots t_n$  the leaves where  $t_k$  are the tokens of the target utterance  $y_i$ . This graph is drawn for every utterance in the corpus for which we would like to calculate proximity to  $y_i$ . The edges represent how well the token  $t_k$  describes the utterance  $u_j$ . Using the cliques formed via edges between one utterance and each token, we create a similarity matrix. For instance, the first row of the similarity matrix shows the similarities of utterance  $u_1$  to every utterance in the corpus and similarly the second row for utterance  $u_2$ . This produces a symmetric matrix, therefore it is sufficient to compute only the half above the diagonal. Then, these similarity values are input into the clustering technique that outputs groups of utterances that are hypothesized to share the same dialogue act.

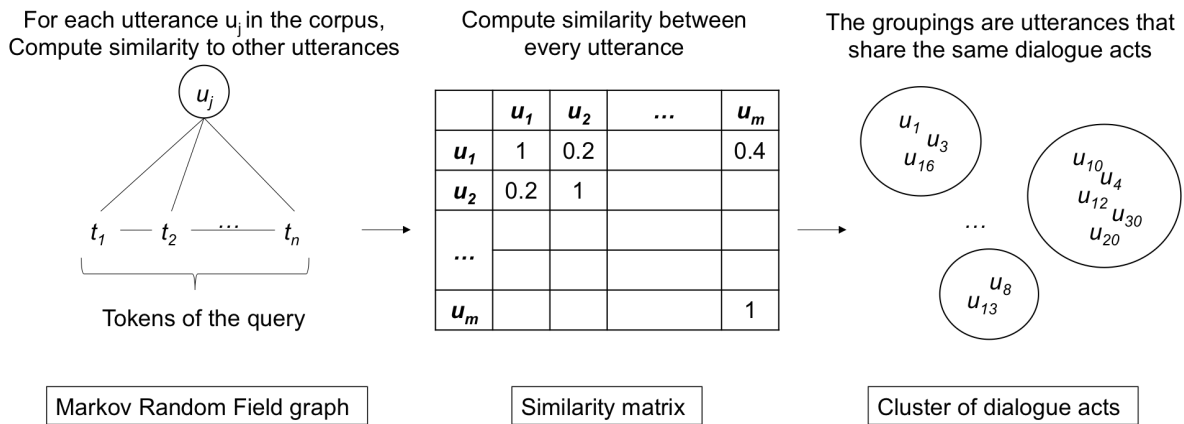


Figure 5.1: The MRF-based clustering framework.

To compute the needed similarities between every pair of utterances in the corpus, we consider 3-node cliques and 4-node cliques which are analogous to bigrams ( $n$ -grams with  $n = 2$ ) and trigrams ( $n = 3$ ) considering the root. Additionally, we consider all values of window size  $w$ , which is analogous to skip bigrams and trigrams. In the MRF, a 2-clique is formed when an

edge exists between two nodes (a token  $t_k$  and utterance  $u_j$ ). This represents a bag-of-words approach where we calculate how well token  $t_k$  describes utterance  $u_j$ . For calculating the edge values of a 3-clique, we search for skip bigrams for all values of  $w$  limited by the length of  $u_j$ . We use the edge values obtained from the graphs as similarity measures between utterances. This approach is a generalization of all three techniques previously mentioned: bag-of-words, skip  $n$ -grams and longest common subsequence.

For the bag-of-words approach ( $n = 1$ , unigrams), we calculate a smoothed language model estimate  $P(t_k|u_j)$  where the probability measures the likelihood of token  $t_k$  describing  $u_j$ . In other words, we wish to estimate the importance of this token in the utterance compared to the importance of this token in the entire corpus. This probability can be estimated as follows:

$$P(t_k|u_j) = [(1 - \alpha_d) \frac{freq(t_k, u_j)}{|u_j|} + \alpha_d \frac{freq(t_k, C)}{|C|}] \quad (5.1)$$

where  $freq(t_k, u_j)$  is the frequency of token  $t_k$  in  $u_j$ ,  $|u_j|$  is the total number of tokens in  $u_j$ ,  $freq(t_k, C)$  is the frequency of token  $t_k$  in the corpus, and  $|C|$  is the total number of tokens in the corpus. The smoothing is included to assign a non-zero probability to unseen words in utterance  $u_j$  that are present in the corpus, a common technique for natural language distributions where many words occur with low frequency (Zhai & Lafferty, 2001).

In addition to unigrams, we also wish to consider the more flexible skip  $n$ -grams with  $w \geq 0$ . To do this, we estimate the probability that tokens  $t_k$  and  $t_m$ , possibly separated by  $w$  other tokens, describe  $u_j$ . This probability  $P(t_k, t_m|u_j)$  for 3-cliques is computed as follows:

$$P(t_k, t_m|u_j) = [(1 - \alpha_d) \frac{freq(t_k * t_m, u_j)}{|u_j|} + \alpha_d \frac{freq(t_k * t_m, C)}{|C|}] \quad (5.2)$$

where  $freq(t_k * t_m, u_j)$  is the frequency of the phrases in utterance  $u_j$  that start with  $t_k$  and end with  $t_m$  with  $w$  tokens between them, and similarly  $freq(t_k * t_m, C)$  is the count of such phrases in the entire corpus. By varying  $w$ , we identify all phrases in which the tokens occur in that order.

The process described above uses cliques to compute similarities among individual constituents  $t_k$  of each target utterance  $y_i = t_1 t_2 \dots t_n$  and all other utterances  $u_j \in C$ . We need an overall similarity between the target utterance  $y_i$  and all other utterances  $u_j$ . To compute this overall similarity, we sum over the 2-clique and 3-clique similarities using the following formula adapted from information retrieval (Metzler & Croft, 2005):

$$M(u_j, y_i) = \sum_{c \in G} \lambda_i P(t_k|u_j) + \sum_{c \in G} \lambda_d P(t_k, t_m|u_j) \quad (5.3)$$

where  $\lambda_i$  is the weight of 2-cliques and  $\lambda_d$  is the weight of 3-cliques. These similarities between utterances are then placed into a matrix  $M$ .

### 5.1.2 Clustering

The similarity results obtained as described above are used as the distance metrics for clustering dialogue acts. For query-likelihood clustering, each utterance that is present in the similarity list (above the defined similarity threshold) is represented as a 1, while the others are represented with a value of 0 (Chapter 4). For MRF-based clustering, each utterance is represented by a vector where similarities are obtained from probabilities in the Markov Random Field model, the matrix  $M$  described above. In this way, each target utterance in the corpus is represented by a vector indicating the utterances that are similar to it. Then the clustering takes the produced matrix as an input to group utterances that are similar to each other. We utilize a widely used clustering algorithm  $k$ -medoids (Ng & Han, 1994) for MRF-based clustering. The entire unsupervised dialogue act classification algorithm for MRF-based clustering is depicted in Table 5.1.

Table 5.1: Markov Random Field-based clustering algorithm.

<p>Let <math>D</math> be a corpus of utterances <math>D = u_1, u_2, \dots, u_n</math>. Then the goal is:  <math>\forall u_j \in D</math>, identify <math>l_j</math> as dialogue act label of <math>u_j</math>          Procedure:            For each utterance <math>u_j</math>              1. Set target utterance to <math>y_i</math>                so that the similarities of <math>y_i</math> to every other utterance will be calculated              2. Build the Markov Random Field graphs <math>G</math> with the tokens of <math>y_i</math>                as the leaf nodes such that every other utterance in the corpus                is represented as roots of <math>G</math>              3. Create vector of similarity results indicator variables from <math>G</math> as                <math>V_j = (v_1, v_2, \dots, v_t, \dots, v_n)</math> such that <math>v_t</math> is obtained from cliques                formed by target utterance <math>y_i</math> and an utterance <math>u_j</math> from the corpus in <math>G</math>            Let the total vector be <math>V_T = (V_1, V_2, \dots, V_j, \dots, V_n)</math>            Return clusters <math>C = c_1, c_2, \dots, c_k</math> such that <math>C</math> is the result of <math>kmedoids(V_T)</math></p>
---

## 5.2 Experiments with Comparisons

The goal of the experiments is to determine whether MRF-based clustering outperforms query-likelihood clustering as hypothesized. Additionally, we compare our implementation against that of the recent approach of Rus et al. (2012), which clusters utterances in an educational corpus via word similarity with Euclidean distance, using a specified number of leading tokens of each utterance. We use accuracy to compare these three models. How to best measure accuracy is a non-trivial question for unsupervised models, but we follow the standard practice of comparing to manual labels, though those labels were not used during model training.

For evaluating the MRF-based dialogue act classification approach, we follow an isomorphic approach to the one used in Chapter 4 (Ezen-Can & Boyer, 2013b): we first retrieve the similarity results for each utterance, and then send the matrix  $M$  of similarities to the clustering algorithm. Then using majority voting, we label each utterance with the most frequent dialogue act tag in its cluster.

*Training set accuracy* evaluates the ability of the models to match the manual labels for the data on which they were trained. Following standard practice for unsupervised model evaluation (Higashinaka et al., 2011; Joty et al., 2011; Rus et al., 2012), we utilize training set accuracy for comparison of the models. The training set accuracy is computed as the number of utterances correctly classified divided by the total number of utterances in the training set.

We explore varying numbers of clusters and provide accuracies for each of them separately. Figure 5.2 depicts the training set accuracy results for MRF-based clustering compared to query-likelihood clustering, the Rus et al. approach with 5 leading tokens, as well as the random chance baseline. This random chance baseline is the most frequently occurring dialogue act, Evaluation Question (EQ), at 27.3%. We use this highest frequency class as the random chance baseline, rather than  $1/9 = 11.1\%$  which would be the accuracy of random guesses among all tags disregarding their frequencies. Choosing the highest frequency tag is a more stringent baseline because a classifier that always guesses EQ will achieve 27.3% accuracy, higher than 11.1%. As shown in Figure 5.2, MRF-based clustering outperforms its counterparts substantially, confirming our hypothesis.

## 5.3 Evaluation of MRF-Based Clustering

In addition to the training set accuracy used for comparison of different models, we are also interested in how well MRF-based clustering performs on unseen test utterances. To investigate this, this section reports on the selection of number of clusters  $k$ , followed by quantitative analyses of *test set accuracy*, *precision* and *recall*. Additionally, we examine the distribution of manual dialogue acts across the unsupervised clusters.

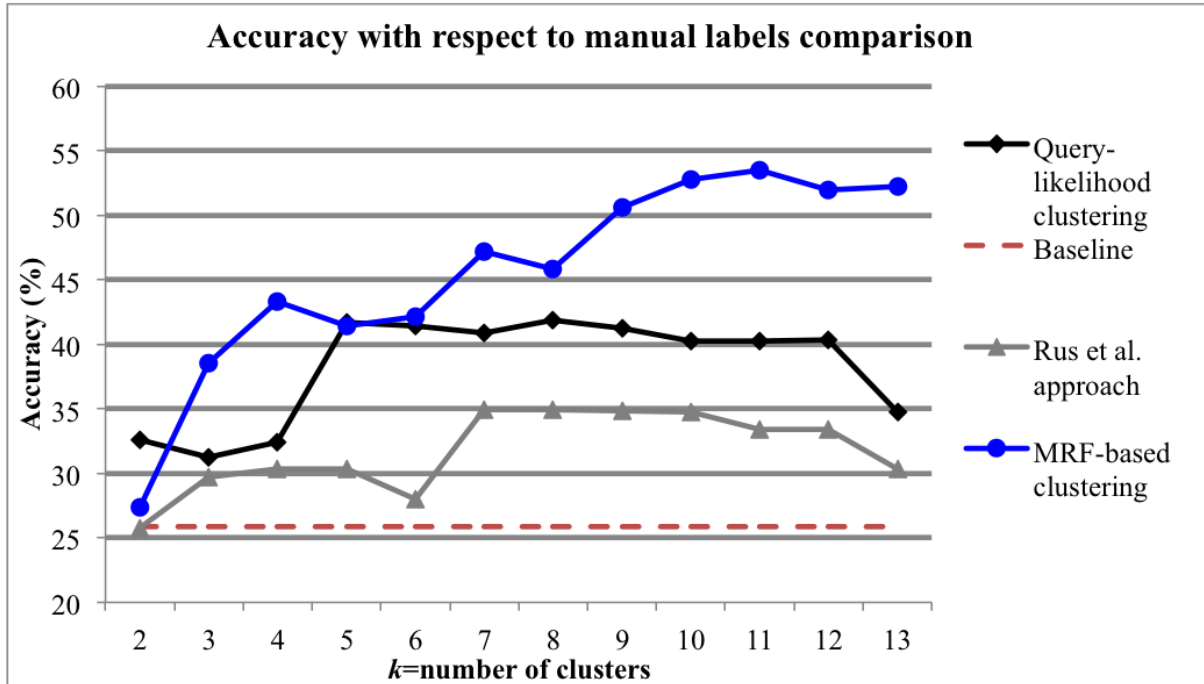


Figure 5.2: Comparison of query-likelihood clustering, MRF-based clustering, Rus et al. approach.

In order to determine the number of clusters for the MRF-based model, the Bayesian Information Criterion (BIC) is computed for each value of  $k$ . BIC penalizes the number of parameters, which is the number of clusters in our case. Lower BIC values represent a better fit to the data (S. S. Chen & Gopalakrishnan, 1998). In our corpus, a model with seven clusters achieved the lowest BIC value (see Figure 5.3). Note that the lowest BIC value does not necessarily correspond to the model with highest accuracy because BIC does not consider manual labels, instead it only measures how coherent the clusters are considering distances between data points. The remainder of this section analyzes the seven-cluster model.

### 5.3.1 Test Set Accuracy, Precision and Recall

For analyzing the model’s performance on unseen test data, we conducted leave-one-student-out cross-validation so that each student’s utterances are included in the held-out test set once. In this way, we avoided providing our model with an unfair advantage because the utterances of the same student are not present both for training and testing. To label each held-out utterance, it was assigned to its closest cluster by taking the minimum average distance to all clusters. The majority label of the closest cluster was assigned as the dialogue act label of the test

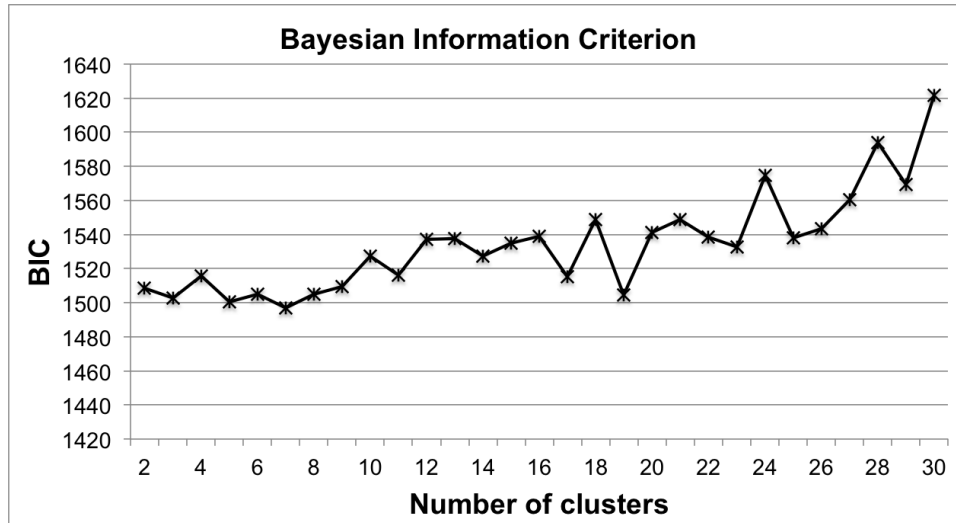


Figure 5.3: BIC values for varying number of clusters.

utterance. Then the test set accuracy was calculated as the number of utterances in the test set that were classified correctly, divided by the total number of utterances in the test set. For the MRF-based clustering, the overall accuracy was 36%. The F-measure for MRF-based clustering was 23.2%, with 24.5% precision and 24.0% recall. Because some dialogue acts have never become majority in any of the clusters, they were never predicted by the model (i.e., NF, LF and GRE). Considering only the dialogue act tags that were predicted by the model, the F-measure was 34.8%, with 36.8% precision and 36.1% recall. This performance is still well above baseline. Note that test set accuracy and F-measure were not reported for query-likelihood clustering nor by Rus et al. The confusion matrix in Figure 5.4 depicts the correct and incorrect predictions for the whole test set.

Similar to the training set evaluations, questions (Q and EQ) were among the most accurately predicted acts in the model. In contrast, there were some dialogue acts (NF, LF, GRE) that were not predicted in the model. The reason for this is that these acts are so infrequent in the corpus that they were not assigned as the majority label of any cluster and therefore were never assigned to any test utterance.

To understand the performance seen above, we examined the distribution of dialogue acts among clusters for one of the folds of the MRF-based clustering (Figure 5.5). The majority dialogue act of each cluster is shown in bold.

The first cluster was mostly composed of extra-domain utterances (utterances that are off-topic) as well as statements and groundings. Because the clustering approaches investigated here did not explicitly use topic information, they grouped on the surface-level features of

		Predicted								
		Q	EQ	S	G	EX	PF	NF	LF	GRE
True	Q	55	195	13	13	0	0	0	0	0
	EQ	107	263	23	23	0	0	0	0	0
	S	66	83	49	11	1	1	0	0	0
	G	6	13	17	130	12	14	0	0	0
	EX	40	44	13	18	7	11	0	0	0
	PF	24	25	7	12	5	43	0	0	0
	NF	53	21	7	7	2	2	0	0	0
	LF	14	9	3	2	2	2	0	0	0
	GRE	18	15	2	2	11	8	0	0	0

Figure 5.4: Confusion matrix for leave-one-student-out cross validation.

utterances, which made it challenging to distinguish utterances that were not related to the task. The second cluster and seventh cluster were mainly questions, both in the form of Q, a general question, and EQ, an “evaluation” question more specific to the task. In examining the second cluster we saw that the model had difficulty differentiating these two question types.

For the difficulty in differentiating questions and statements, we observed an interesting point about the corpus. Because the data collection was with novice students, it is very common for these students to use a declarative with a question mark attached such as “while declaring the loop I can use *i* again since it’s local right?” and “no wait we need *if* for the conditions right”. Given the features used within the models, these questions and statements appeared similar in structure, which may explain the model’s difficulty in differentiating them. The third cluster had mostly statements, and the model was successful in grouping grounding dialogue acts in the fourth cluster. The fifth cluster was highly impure and its interpretation was not straightforward. Finally, the sixth cluster grouped positive feedback utterances together.

### 5.3.2 Qualitative Evaluation

In this section, we evaluate the resulting MRF-based clusters qualitatively by examining the utterances grouped in each cluster. Table 5.2 presents a sampling of utterances from each cluster of the model with seven clusters.

As shown by the examples of cluster 1, many extra-domain utterances and statements were grouped together. From surface-level features, it is difficult to distinguish dialogue acts that may be labeled differently by human coders; for example, “oh” was tagged as extra-domain

<i>Cluster No.</i>	<i>Q</i>	<i>EQ</i>	<i>S</i>	<i>G</i>	<i>EX</i>	<i>PF</i>	<i>NF</i>	<i>LF</i>	<i>GRE</i>
1	16	16	53	52	<b>63</b>	36	18	7	43
2	<b>120</b>	114	77	2	31	25	54	12	5
3	10	15	<b>26</b>	0	6	0	2	1	0
4	7	14	4	<b>126</b>	10	4	4	1	1
5	18	<b>19</b>	11	7	11	11	5	2	0
6	1	0	1	0	5	<b>32</b>	0	2	0
7	98	<b>225</b>	37	1	4	1	2	2	6

Figure 5.5: Distribution of dialogue acts over clusters in one of the folds. Bold face indicates the majority dialogue act in each cluster.

and “oohhhh” as positive feedback. (Dialogue history is highly influential on different labels of these utterances and as discussed in Section 5.4, it is important to leverage within dialogue act models.) Likewise, the utterance “beginning now” which was labeled as off-topic is syntactically a statement.

Cluster 2 was mostly composed of questions and evaluation questions. The same finding we noted when examining the distribution of dialogue act tags within clusters was visible here, as both questions and evaluation questions were similar in structure.

The utterances in cluster 3 were generally in the form of statements regardless of their dialogue act tags. For instance, the two utterances, “in a while loop” (which is tagged as question) and “so it would be like assignment” (which is tagged as evaluation question) were syntactically closer to statements than questions when the utterances were considered alone. Once again incorporating dialogue structure so that the model benefits more from the whole dialogue rather than the surface-level features alone is an important challenge moving forward, to address this issue.

It was clear that some clusters were influenced by words: cluster 4 saw many words like “ok” while cluster 6 saw many occurrences of “yes.” In cluster 5, in addition to the questions, some utterances which seemed like statements but which were manually tagged as questions appeared. For example, “well this array is taking ints and we are putting in characters from a string” had a question tag from manual labeling.

## 5.4 Discussion: MRF-Based Clustering

Automatically understanding natural language that students exchange as they are learning is an increasingly prominent problem for educational data mining research. To achieve truly scalable models, unsupervised approaches hold great promise as they aim to address the labor-intensive nature of engineering taxonomies and manual labeling. We described an unsupervised model aimed at modeling student dialogue acts without requiring labor-intensive efforts for annotations and showed that 36% cross-validated test set accuracy is achievable by MRF-based clustering. As promising as unsupervised models are, they pose important challenges. The dialogue act classifiers presented here have exemplified both the great promise and challenges of unsupervised dialogue modeling.

First, the results illustrate that while dialogue acts are a very useful distinction for educational dialogues, we also need other distinctions such as topic. For example, the dialogue act tag EX indicating off-topic utterances was not distinguished successfully by MRF-based clustering or the prior approaches used for comparison. We have begun to explore combinations of unsupervised dialogue act models with unsupervised topic models, which may address this challenge.

Second, the discrimination of dialogue act tags according to whether they referred specifically to the task (e.g., for questions and evaluation questions) was hard for the dialogue act classifier. For example, the data-driven groupings did not distinguish “exactly how would I make an array an instance parameter,” which is manually annotated as a question, and “do i set it equal to the conditions of parameter,” which is labeled as an evaluation question. In order to address this challenge, the research community has begun to explore both unsupervised and supervised semantic mapping from utterances to the learning task as a second stage to dialogue act classification. By doing this, first the questions can be successfully grouped together in one step, and then it can be determined whether they refer to the task in a second step.

The results presented in this chapter showed the necessity of utilizing a richer set of features for dialogue act classification, including dialogue history. In the next chapter, we present a novel representation of dialogue history for clustering student utterances.

Table 5.2: Example utterances from each cluster.

<i>Cluster 1</i>
-oh [EX] -oohhhh [PF] -let me fix this real fast [S] -beginning now [EX] -fair enough [G]
<i>Cluster 2</i>
-exactly how would i make an array an instance parameter [Q] -do i set it equal to the conditions of parameter [EQ] -could i just do that with a string [EQ] -ok so how would i add up each digit [Q] -all i remember how to do is convert strings into ints [LF]
<i>Cluster 3</i>
-in a while loop [Q] -this should be a string [EQ] -we want it to equal zero so should this be ten [S] -go on [Q] -so it would be like assignment [EQ]
<i>Cluster 4</i>
-ok [G] -oh ok [G] -ok like loop [EQ] -ok however [NF]
<i>Cluster 5</i>
-is that already declared somewhere [Q] -but since parameter is already given as an int [Q] -is this different than my parameter thing [Q] -and also if my parameter is correct [EQ] -it looks like it is going to come to that [S]
<i>Cluster 6</i>
-yes [PF] -yes giving me definitions to various commands and such [EX] -ohh yes [EX]
<i>Cluster 7</i>
-how can you get the sum of parameter [EQ] -ok so it would be like assignment and so on until the last digit [EQ] -what is the name of the postal code in the program [Q] -can the chararray not be used outside because it is in a private method? [EQ] -where are the drawing functions provided [Q]

## Chapter 6

# Enriching Feature Sets for Task-Oriented Dialogue Act Modeling

This chapter extends unsupervised dialogue act classifiers with rich feature sets derived within task-oriented dialogue. In order to close the performance gap between unsupervised and supervised dialogue act classifiers, we suggest that it is crucial to enrich the features available to unsupervised models. In particular, when a dialogue is task-oriented and includes a rich source of information within a parallel task stream, these features may substantially boost the ability of an unsupervised model to distinguish dialogue acts. For example, in situated dialogue, features representing the state of the physical world may be highly influential for dialogue act modeling (Grosz & Sidner, 1986).

Human tutorial dialogue, which is the domain being considered in the current work, often exhibits this structure: the task artifact is external to the dialogue utterances themselves (in the case of our work, this artifact is a computer program that the student is constructing). Task features have already been shown beneficial for supervised dialogue act classification in our domain (Ha et al., 2012). We hypothesize that including these task features within an unsupervised model will significantly improve its performance. In addition, we hypothesize that including dialogue history as a prominent feature within an unsupervised model will provide significant improvement.

This chapter represents the first investigation into combining task and dialogue features within an unsupervised dialogue act classification model (Ezen-Can & Boyer, 2014a). First, we discuss representation of these task features and dialogue structure features, and compare these representations within both flat and hierarchical clustering approaches. Second, we report on experiments that demonstrate that the inclusion of task features significantly improves

dialogue act classification, and that a hierarchical cluster structure which explicitly captures dialogue history performs best. Finally, we break down the model’s performance by dialogue act and investigate which features are most beneficial for distinguishing particular acts. These contributions constitute a step toward building high-performing unsupervised dialogue act models that can be used in the next generation of task-oriented dialogue systems.

## 6.1 Features

A key issue for dialogue act classification in task-oriented dialogue involves how to represent dialogue and task events. This section describes how features were extracted from the corpus of human tutorial dialogue. We use three sets of features: lexical features, dialogue context features, and task features. The lexical and dialogue context features are extracted from the textual dialogue utterances within the corpus. The task features are extracted from the interaction traces within the computer-mediated learning environment and represent a keystroke-level log of events as students worked toward solving the computer programming problems.

### 6.1.1 Lexical Features

Because one of the main goals of our work in the longer term is to perform automatic dialogue act classification in real-time, we took as a primary consideration the ability to quickly extract lexical features. The features utilized in the current investigation consist only of word unigrams. In addition to their ease of extraction, our prior work has shown that addition of part-of-speech tags and syntax features did not significantly improve the accuracy of supervised dialogue act classifiers in this domain (Boyer et al., 2010), and these features can be time-consuming to extract in real time (Ha et al., 2012).

The choice to use word unigrams rather than higher order  $n$ -grams is further facilitated by the fact that our clustering technique leverages the *longest common sub-sequence (LCS)* metric to measure distances between utterances. This metric counts shared sub-sequences of not-necessarily contiguous words. In this way, the LCS metric provides a flexible way for  $n$ -grams and skip- $n$ -grams to be treated as important units within the clustering, while the raw features themselves consist only of word unigrams. To illustrate the LCS distance metric at work, consider two sentences: “I should declare a variable,” and “Should I declare a variable.” When using word unigram features in combination with many widely used distance metrics including Cosine, Euclidean, Manhattan, and Jaccard distance, these two sentences would have perfect similarity, although in the tutorial dialogue domain we consider them to be different dialogue acts. In contrast, the LCS metric computes the longest common subsequence – in this case there are two, of length four – “should declare a variable” or “I declare a variable” and the two utterances are

no longer considered to have perfect similarity because they differ in word ordering. Utilizing LCS, there exists a distance (1-similarity) value from each utterance to every other utterance.

### 6.1.2 Dialogue Context Features

Based on previous work on a similar human tutorial dialogue corpus (Ha et al., 2012), we utilize four features that provide information about the dialogue structure. These features are depicted in Table 6.1. Note that our goal within this work is to classify *student* dialogue moves, not tutor moves, because in a dialogue system the tutor’s moves are system-generated with associated known dialogue acts.

Table 6.1: Dialogue context features and their descriptions.

<b>Feature</b>	<b>Description</b>
<i>Utterance position</i>	The relative position of an utterance from the beginning of the dialogue.
<i>Utterance length</i>	The number of tokens in the utterance, including words and punctuation.
<i>Previous author</i>	Author of the previous dialogue message (tutor or student) at the time message sent.
<i>Previous tutor dialogue act</i>	Dialogue act of the previous tutor utterance.

### 6.1.3 Task Features

As described previously, the corpus contains two channels of information: the dialogue utterances, from which the lexical and dialogue context features were extracted, and in addition, the task stream consisting of student problem-solving activities such as authoring code, compiling, and executing the program. The programming activities of students were logged to a database along with all of the dialogue events during tutoring.

A set of task features was found to be important for dialogue act classification in this domain in prior work, including most recent programming action, status of the most recent task activity and task activity flag representing whether the utterance was preceded by a student’s task activity (Ha et al., 2012). We expand this set of features as shown in Table 6.2.

Table 6.2: Task features extracted from student computer programming activities.

<b>Feature</b>	<b>Description</b>
<i>prev_action</i>	Most recent action of the student (composing a dialogue utterance, constructing code, compiling or executing code).
<i>task_begin</i>	Whether the student utterance is the first utterance since the beginning of the subtask.
<i>task_stu</i>	Whether the student utterance was preceded by a task event.
<i>task_prev-tut</i>	Task activity flag indicating whether the closest tutor utterance in this subtask was preceded by a task activity.
<i>task_status</i>	The status of the most recent coding action ( <i>begin</i> , <i>stop</i> , <i>success</i> , <i>error</i> and <i>input_sent</i> ).
<i>time_elapsed</i>	Time elapsed between the previous tutor message and the current student utterance.
<i>errors</i>	Number of errors in the student’s latest code.
<i>delta_errors</i>	Difference in the number of errors in the task between two utterances in the same dialogue.
<i>stu_#_task</i>	Number of student dialogue messages sent within the current task.
<i>stu_#_dial</i>	Number of student dialogue messages sent within the current dialogue.
<i>tut_#_task</i>	Number of tutor dialogue messages sent within the current subtask.
<i>tut_#_dial</i>	Number of tutor dialogue messages sent within the current dialogue.

## 6.2 Enriched Feature Set: Methodology

The goal of this work is to investigate the impact of including task and dialogue context features on unsupervised dialogue act models. We hypothesize that incorporating task features will significantly improve the performance of an unsupervised model, and we also hypothesize that properly incorporating dialogue context features, which are at a different granularity than the lexical features extracted from utterances, will substantially improve model accuracy.

### 6.2.1 Dialogue Act Modeling With $k$ -medoids Clustering

As in the previous chapter, the unsupervised models investigated here use  $k$ -medoids clustering, which is a well-known clustering technique that takes actual data points as the center of each cluster (Ng & Han, 1994), in contrast to  $k$ -means which may have synthetic points as centroids.

In  $k$ -medoids, the centroids are initially selected and then the algorithm iterates, reassigning data points in each iteration, until the clusters converge. In standard  $k$ -medoids clustering the initial seeds are selected randomly and then a correct distribution of data points is identified through the iteration and convergence process. For dialogue act classification, the influence of the initial seeds is substantial because the frequencies across dialogue tags are typically unbalanced. To overcome this challenge, the models reported in this chapter use a greedy seed selection approach similar to the one used in  $k$ -means++ (Arthur & Vassilvitskii, 2007) which selects the first seed randomly and then greedily chooses seeds that are farthest from the chosen seeds. The goal of using this approach in our application is to choose seeds from different dialogue acts so that the final model achieves good coverage. Our preliminary experiments demonstrated that this greedy seed selection combined with  $k$ -medoids outperforms other clustering approaches including those utilized in our prior work (Ezen-Can & Boyer, 2013a).

In order to select the number of clusters  $k$ , a subset of the corpus, constituting 25% of the full corpus (that were not tagged) composed of 462 utterances, was separated as a development set. First, we examined the coherence of clusters at different values of  $k$  using intra-cluster distances. This technique involves identifying an ‘elbow’ where the decrease in intra-cluster distance becomes less rapid (since adding more clusters can continue to decrease intra-cluster distance to the point of overfitting) (Figure 6.1). The graph suggests an elbow at  $k=5$ . Because there may be multiple elbows in the intra-cluster distance, a second method utilizing Bayesian Information Criterion (BIC) was used which penalizes models as the number of parameters increases. The lower the BIC value, the better the model is, achieved at  $k=5$  as well.

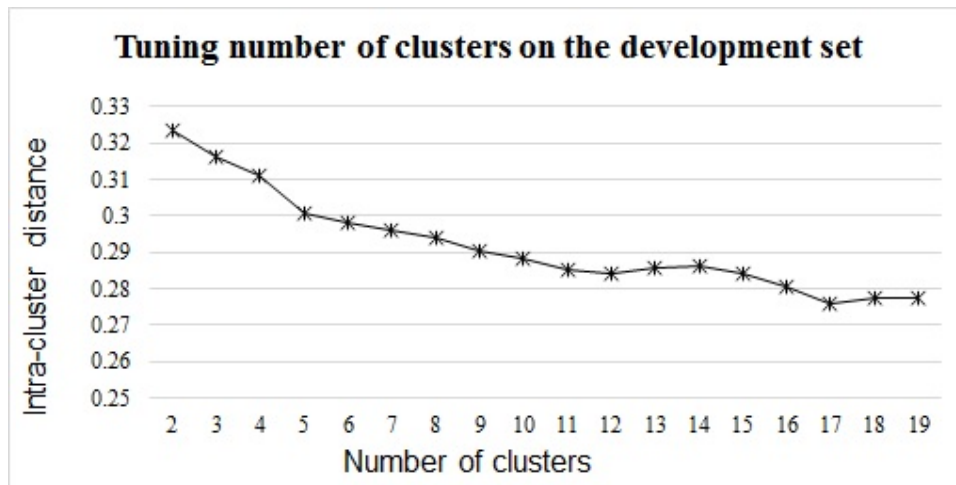


Figure 6.1: Intra-cluster distances with varying number of clusters.

Unlike many other investigations into unsupervised dialogue act classification, the current approach reports accuracy on held-out test data, not on the data on which the model was trained. Even though the model training process does not utilize available manual tags, requiring the learned unsupervised model to perform well on held-out test data more closely mimics the broader goal of our work which is to utilize these unsupervised models within deployed dialogue systems, where most utterances to be classified have never been encountered by the model before.

The procedure for model training and testing uses leave-one-student-out cross-validation. Rather than other forms of leave-one-out or stratified cross-validation, leave-one-student-out ensures that each student’s set of dialogue utterances are treated as the testing set while the model is trained on all other students’ utterances. This process is repeated until each student’s utterances have served as a held-out test set (in our case, this results in  $n=42$  folds). Within each fold, the clusters are learned during training and then for each utterance in the test set, its closest cluster is computed by taking the average distance of the test utterance to the elements in the cluster. The majority label of the closest cluster is assigned as the dialogue act tag for the test utterance. If the assigned dialogue act tag matches the manual label of the test utterance, the utterance is counted as correct classification. The average accuracy is computed as the number of correct classifications divided by the total number of classifications.

### 6.2.2 Experimental Results

We conducted experiments with seven different feature combinations:  $L$ , lexical features only,  $T$ , task features only,  $D$ , dialogue context features only, and then the combinations of these features,  $T+D$ ,  $T+L$ ,  $D+L$ , and  $T+D+L$ . We hypothesized that the addition of task features would significantly improve the models’ accuracy. As shown in Table 6.3, adding task features to dialogue context features significantly outperforms dialogue context features alone ( $T+D > D$ ). Similarly, adding task features to lexical features provides significant improvement ( $T+L > L$ ). However, adding task features to the dialogue context plus lexical features model does not provide benefit, and in fact slightly (not significantly) degrades performance ( $T+D+L \not> D+L$ ). As reflected by the Kappa scores, the test set performance attained by these models is hardly better than would be expected by chance. In Table 6.3, “L” represents lexical features, “D” represents dialogue context features and “T” represents task features, \*indicates statistically significant compared to the similar model without task features ( $p < 0.05$ ).

### 6.2.3 Utilizing Dialogue History

The importance of dialogue history, particularly the influence of the most recent turn on an upcoming turn, is widely recognized within dialogue research, notably by work on adjacency pairs

Table 6.3: Test set accuracies and Kappa for the flat clustering model.

	<b>Features</b>	<b>Accuracy (%)</b>	<b>Kappa</b>
Flat Clustering	L	33	0.02
	T	37.7	0.07
	D	37.6	0.07
	T+D	39.1*	0.07
	T+L	38*	0.06
	D+L	38.3	0.07
	T+D+L	37.3	0.05

(Schegloff & Sacks, 1973; Forbes-Riley, Rotaru, Litman, & Tetreault, 2007; Midgley, Harrison, & MacNish, 2009). Based on these findings, we hypothesized that dialogue history would be substantially beneficial for unsupervised dialogue act models as it has been observed to be in numerous studies on supervised classification. However, as seen in the previous section, the addition of these dialogue context features to the model only improved its performance slightly though statistically significantly (for example,  $T + D > T$ ), but the overall performance is still barely above random chance.

In an attempt to substantially boost the performance of the unsupervised dialogue act classifier, we experimented with a hierarchical clustering structure in which the model first branches on the previous tutor move, and then the clustering models are learned as described previously at the leaves of the tree (Figure 6.2).

This branching approach results in some branches with too few utterances to train a multi-cluster model. To deal with this situation we set a threshold of  $n=10$  utterances. For those subgroups with fewer than 10 utterances, we take a simple majority vote to classify test cases, and for those subgroups with 10 or larger utterances we train a cluster model and use it to classify test cases. For the entire corpus, the number of utterances in each branch is presented in Table 6.4.

As the results in Table 6.5 show, the performance of the model with hierarchical structure is significantly better than the flat clustering model. Note that each feature in this table leverages previous tutor dialogue act while branching. Branching on previous tutor move boosted the model’s accuracy for student move dialogue act classification by approximately 30% accuracy across all feature sets, a difference that is statistically significant in every case. With the hierarchical model structure, the best performance is achieved by including all three types of features: lexical, dialogue context and task. However, our hypothesis that task features

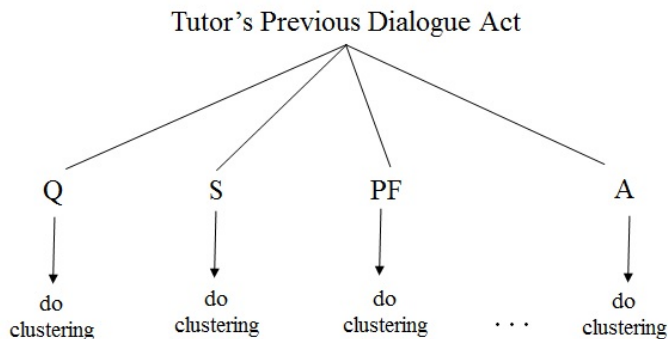


Figure 6.2: Branching student utterances according to previous tutor dialogue act.

Table 6.4: The number of student utterances after branching on the previous tutor dialogue act.

Tutor Dialogue Act	# of student utterances
Q	818
S	464
H	125
PF	91
A	61
ACK	11
C	8
O	8
RACK	6

would significantly improve the accuracy does not hold within the hierarchical clustering model ( $T + D \not\asymp D$  and  $T + L \not\asymp L$ ). In Table 6.5, “L” represents lexical features, “D” represents dialogue context features and “T” represents task features, \*indicates statistically significant compared to the similar model without task features and † indicates hierarchical clustering performing significantly better than flat with same features. ( $p < 0.05$ ).

### 6.3 Results

The experimental results provide compelling evidence that an inclusive approach to features for unsupervised dialogue act modeling holds great promise. However, we observed a stark difference

Table 6.5: Test set accuracies and Kappa for branching on previous tutor dialogue act.

	Features	Accuracy (%)	Kappa
Hierarchical	T	64.2 <sup>†</sup>	0.45
	D	63.2 <sup>†</sup>	0.46
	L	60.7 <sup>†</sup>	0.41
	T+D	62.1 <sup>†</sup>	0.44
	T+L	63.3 <sup>*†</sup>	0.45
	D+L	63.6 <sup>†</sup>	0.46
	T+D+L	<b>65<sup>*†</sup></b>	0.48

in model performance when the tutor’s previous move was simply included as one of many features within a flat clustering model compared to when the previous tutor move was treated as a branching feature. In this section we take a closer look and discuss the features that help distinguish particular dialogue acts from each other.

Using the hierarchical  $T + D + L$  model which performed best within the experiments, we examine the confusion matrix (Figure 6.3). Statements and acknowledgments prove challenging for the model, 51.3% and 61.5% accuracy overall. Moreover, these two tags are easily confused with each other: 29.7% of statements were misclassified as acknowledgments, while 21.2% of acknowledgments were misclassified as statements. The worst overall classification accuracy was for questions (6%) and the best was achieved for answers (95.3%).

When we analyze the performance of different sets of features with respect to individual dialogue acts, some interesting results emerge. The analysis shows that task features are especially good for classifying statements. Using only task features, the model correctly classified 61.8% statements, compared to the lower 51.3% accuracy that the overall best model ( $T + D + L$ ) achieved on statements. When we consider the nature of the statement dialogue act within this corpus, we note that it is a large category that encompasses a variety of utterances, some of which have lexical features in common with acknowledgments. In this case, task features are particularly helpful.

For acknowledgments, a combination of task and lexical features performed best (63.6% accuracy) compared to the overall best performing model which achieved a slightly lower 61.5% accuracy on acknowledgments. Acknowledgments are another example of an act that may take ambiguous surface form; for example, in our corpus an utterance ‘yes’ appears as both an answer and an acknowledgment depending on its context. Therefore, higher level features such as the ones provided by task may be more helpful.

		Predicted									
		S	A	Q	ACK	C	RACK	RF	O	H	PF
True	S	181	48	17	105	2	0	0	0	0	0
	A	15	645	6	11	0	0	0	0	0	0
	Q	83	78	14	58	0	0	0	0	0	0
	ACK	71	44	14	206	0	0	0	0	0	0
	C	8	4	1	1	0	0	0	0	0	0
	RACK	4	1	1	1	0	0	0	0	0	0
	RF	1	4	1	1	0	0	0	0	0	0
	O	5	5	0	0	0	0	0	0	0	0
	H	1	0	0	0	0	0	0	0	0	0
	PF	3	0	0	0	0	0	0	0	0	0

Figure 6.3: Confusion matrix for hierarchical model utilizing all features: T+D+L.

For questions, the highest performing feature set is  $L$ . However, as shown in Table 6.6, the model performed poorly on questions. Inspection of the models reveals that questions are varied in terms of structure throughout the corpus and it is hard to distinguish them from other dialogue acts. For instance there are two consequent utterances “i need a write statement” and “don’t i”, both of which are manually labeled as questions. However, in terms of the structure, the first utterance looks very similar to a statement and therefore the model has difficulty grouping it with questions. Due to the large variety of question forms in the corpus, it is possible that the clustering performed poorly on this dialogue act. In future work it will be promising to investigate the dialogue structures which produce questions and to weight them more in the feature set in order to increase performance of clustering for questions.

This chapter analyzed the addition of dialogue history and task-context features to improve dialogue act classification performance. Next chapter considers a different perspective: learner characteristics. We investigate whether addition of learner characteristics obtained from pre-surveys help improve understanding students.

Table 6.6: Accuracies for individual dialogue acts. Acts with fewer than 10 utterances after branching are omitted from the table.

<b>Features</b>	<b>S</b>	<b>A</b>	<b>Q</b>	<b>ACK</b>
L	21.5	41.3	<b>14.2</b>	20.4
T	<b>61.76</b>	95.27	7.30	40.90
D	48.16	95.27	3.00	60.30
T+D	52.69	94.68	3.43	51.64
T+L	42.78	95.13	6.01	<b>63.58</b>
D+L	43.63	94.98	8.58	62.09
T+D+L	51.27	95.27	6.01	61.49

## Chapter 7

# Incorporating Learner Characteristics into Dialogue Act Models

Despite the growing focus on developing unsupervised dialogue act classifiers such as the approaches described in the previous chapters, these models still underperform compared to supervised approaches in their accuracy for classifying according to manual tags. However, while unsupervised models to date have considered such things as lexical features (the words found in the utterance) and syntactic features (the structure of the sentence), they have not considered learner characteristics, such as gender and self-efficacy, which are believed to influence the structure of tutorial dialogue (Boyer, Vouk, & Lester, 2007). Learner characteristics play an influential role in learning, not only in tutoring but in classroom settings (Ames & Archer, 1988), and in web-based courses (HersHKovitz & Nachmias, 2011). Prior work on learner characteristics has focused on building adaptive systems based on different user groups (Forbes-Riley & Litman, 2009b), tutorial feedback selection (Boyer, Phillips, Wallis, Vouk, & Lester, 2008) and identifying students that need remedial support (Merceron & Yacef, 2005). Identifying clusters of student characteristics is also an active area of research (Azarnoush, Bekki, & Bernstein, 2013; Meece & Holt, 1993; Merceron & Yacef, 2003, 2005).

This chapter investigates whether the performance of an unsupervised dialogue act classifier can be improved by taking learner characteristics into account (Ezen-Can & Boyer, 2014b, 2014c). In order to successfully divide by learner characteristics, we depart from the previously reported query-likelihood clustering and utilize a clustering approach that performs better with smaller data sets. We utilize three learner characteristics: gender, as self-reported by students on a survey; domain-specific self-efficacy, as measured by a validated instrument for determining a student's confidence in her own abilities, and skill perception, a student's ranking of her own

skill as she perceives it compared to others (see Table 7.1 and 7.2 for excerpts). Specifically, we train unsupervised dialogue act models that are tailored to students of specific gender, self-efficacy level, and skill perception, and we compare those models to corresponding ones trained without restricting by that learner characteristic. This unsupervised training is conducted entirely without the use of manual tags. We then test all of the models on held-out test sets within leave-one-student-out cross validation, and compare the resulting classification accuracy according to their previously applied manual tags. The results show that for female students, utilizing learner characteristics statistically significantly improves dialogue act classification models. For the other groups, improvement is observed but not at a statistically reliable level.

Table 7.1: Excerpt of dialogue with a female student in the low self-efficacy and low skill perception groups.

Role	Utterance	Dialogue Act
Tutor	Alright are you ready to start writing Java code yourself?	Q
Student	yes!	A
Tutor	Great!	S
Student	i finished reading it	S
Tutor	Perfect	PF
Student	so whenever i start to write something there should be in front of it?	Q
Tutor	No this is a special type of Java line	A
Tutor	The // in the line tells Java to ignore whatever you write after it	S
Student	oh i see!	ACK

## 7.1 Dialogue Act Modeling with Learner Characteristics: Methodology

Students were given a pre-survey that included survey items on computer science self-efficacy, such as I am sure I can learn programming. This self-efficacy scale was adapted directly from the Domain-specific Self-Efficacy Scale (Bandura, 2006), with five items measured on a Likert scale from 1-5 (1 being lowest self-efficacy, 5 being highest). Students also completed a demographic questionnaire from which gender was obtained. Finally, the pre-survey included an item that

Table 7.2: Excerpt of dialogue with a male student in the low self-efficacy and low skill perception groups.

Role	Utterance	Dialogue Act
Tutor	You'll need to end every Java statement with a semi colon	S
Student	Got it!	ACK
Tutor	This is to let Java know where each statement ends	S
Tutor	Ah no prompt!	S
Tutor	Why do you think that is?	Q
Student	I wish I knew...	A
Student	I don't think I spelled anything wrong	S
Tutor	Ah it's actually pretty easy	S
Tutor	The order of the lines matters	S
Tutor	When the Java program is executed it runs line by line	S
Tutor	So when you read your program you can imagine each line happening in sequence	S
Student	Oh so I need to put that above the last step?	Q
Tutor	Yes!	A

asked students to rate how skilled they are in the domain compared to others, and we call this response skill perception. This too was a 1 to 5 Likert scale with 5 being the most skilled. For self-efficacy and skill perception, students were divided into classes based on the median score across all students on that scale. Along with gender, this produces three partitions of the 42 students: females and males, low and high self-efficacy students, and low and high skill perception level (Figure 7.1). Note that although the partitions were determined based on median, for our ordinal scales the repetition of the median value means that the high and low groups for these characteristics are of unequal size.

Analysis of the corpus indicates patterns of differences in language expression between

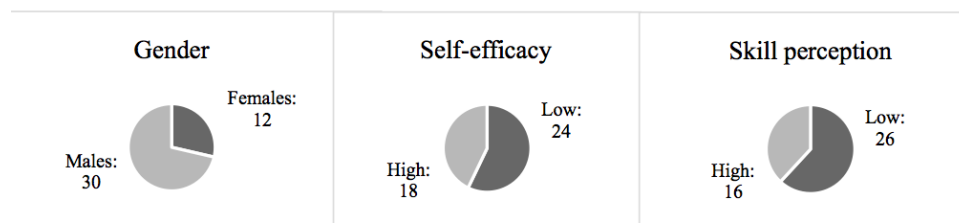


Figure 7.1: Distribution of students in the corpus.

learner characteristics are from two-tailed  $t$ -tests with equal variance that have not undergone a Bonferroni correction because of the exploratory nature of the investigation.<sup>4</sup> First, although the percentage of dialogue act tags does not vary in a statistically significant way between any of the learner characteristic groups, usage of some punctuation differs significantly. Women are significantly more likely to use question marks (4.4% of their tokens) than male students (2.9% of their tokens;  $p < 0.05$ ), though women and men have statistically similar percent of question dialogue acts (18% for women, 13% for men,  $p = 0.07$ ). Similarly, women tend to use significantly more exclamation marks (average 2% of tokens) than males (average 0.9%) ( $p < 0.05$ ). Additionally, although the relative frequency of exclamation marks is very low, we also observe that students with low skill perception use 1.7% of tokens as exclamation marks on average, while students with high skill perception use only 0.5% of tokens as exclamation marks ( $p < 0.05$ ). In addition to these differences in punctuation, differences were observed in the length of utterances. The utterances were segmented into tokens, where a token is either a word or a punctuation mark. As depicted in Table 7.3, students of different characteristics tend to use different lengths of utterances. To illustrate this, we computed the percent of utterances with 1, 2, and 10 tokens respectively for each learner group. For very short utterances of 1 or 2 tokens, there was no difference by gender or self-efficacy but for skill perception, students with high skill perception made a significantly larger proportion of 1-token utterances (52%) than students with low skill perception (35%;  $p < 0.005$ ). Conversely, students with low skill perception made significantly more 2-token utterances, with 16% compared to 7.9% being 2-token utterances for students with high skill ( $p < 0.005$ ). When comparing longer utterances, again arises as shown in Table 7.3 for 10-tokens. The students with high skill perception made significantly fewer 10-token utterances (1.69%) than students with low skill perception (2.95%). In contrast, students with high self-efficacy made more 10-token utterances than students with low self-efficacy. In Table 7.3, \* indicates statistically significant with  $p < 0.05$ ; \*\* indicates  $p < 0.005$ .

We hypothesize that dialogue act models built using unsupervised machine learning will perform substantially better when customized to specific learner groups. Specifically, we investigate whether by training a model only on students of a particular learner characteristic, that model would perform significantly better at predicting the dialogue acts of unseen students with the same learner characteristic compared to a model that was trained on students of all learner characteristics.

For experimental purposes, the corpus is partitioned into three different splits, each containing utterances from different learner groups. For the first split, the corpus is partitioned by gender and we examine whether an unsupervised dialogue act classifier trained on females only performs better on a test set of dialogue acts of females, compared to a classifier trained on a mixture of

---

<sup>4</sup>The  $p$ -values reported in Section 7.1 However, Bonferroni correction is applied to the experimental comparison  $p$ -values in Section 7.2.

Table 7.3: Average percentage of utterance length.

	<b>1 token</b>	<b>2 tokens</b>	<b>10 tokens</b>
Females	37.75	13.91	2.24
Males	43.53	12.86	2.56
Low-efficacy	41.24	12.88	1.58*
High-efficacy	42.73	13.54	3.65*
Low skill perception	35.57**	16.38**	2.95*
High skill perception	52.14**	7.93**	1.69*

females and males. We perform similar comparisons for the partitions by self-efficacy and by skill perception.

First, we note that because the same corpus is being partitioned in three different ways, the same student will occur in one of the gender groups, one of the self-efficacy groups, and one of the skill perception groups (Figure 7.2). This choice to partition in 2-way splits rather than  $2n$ -way splits where  $n$  is the number of learner characteristics is because of issues that arise with sparsity. For example, the number of female students with high self-efficacy and low skill perception is three, while the number of male students with these levels of self-efficacy and skill is five. This interdependence between partitions is a limitation to note; however, as discussed in Section 7.3, this interdependence can be taken into account for making decisions within a tutorial dialogue system by employing a suite of classifiers within a voting scheme.

### 7.1.1 Experimental Design

For each learner characteristic partition, we will test whether an unsupervised dialogue act classifier trained only on students with that characteristic outperforms a classifier that is not specialized by this characteristic. In order to gather accuracy data across these characteristics, we conduct leave-one-student-out training and testing folds. The testing set for each of the  $n$  folds (where  $n$  varies depending on which learner group is being considered) consists of all of a single student’s dialogue utterances and the model is trained on the remaining  $n - 1$  students. The average number of utterances per student in the corpus is 36.8 ( $\sigma=12.07$ ;  $\text{min}=16$ ;  $\text{max}=64$ ). These are therefore the average, minimum, and maximum number of utterances across the leave-one-student-out test sets. We compute the average test set performance of the model across all folds for each learner characteristic partition. The performance metric utilized in this study is accuracy compared to the manually labeled dialogue acts described in the previous section, where accuracy is computed as the number of utterances in the test set that were classified according to their manual label, divided by the number of utterances total in the test set. The

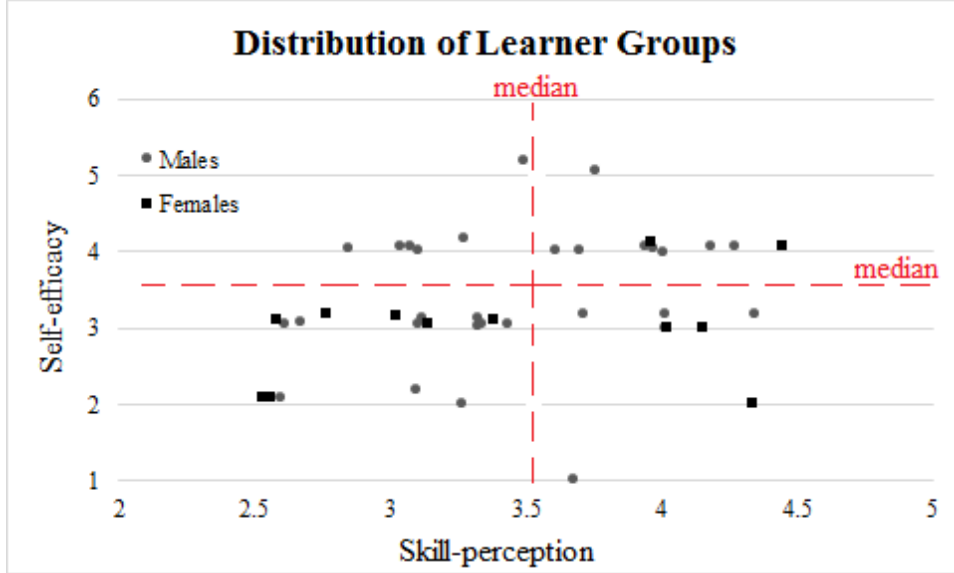


Figure 7.2: Distributions of students with respect to learner characteristics (data points jittered to reveal overlap).

process of labeling via unsupervised classification involves taking the majority vote within each cluster.

For constructing the folds, we take an approach to balance the sample size available to model training. This balancing approach is needed to ensure that each model is trained on a similar size of data. Consider, for example, the partition of gender. Without a balanced sampling approach the leave-one-student-out testing folds for the un-specialized classifier for female students would include  $n_{\text{female}}=12$  test folds but the available data for each training fold would be  $n_{\text{total}-1}=41$ . In contrast, the specialized classifier trained only on female students would still include  $n_{\text{female}}=12$  test points but the available data for each training fold would be  $n_{\text{female}-1} = 11$ . Therefore, each un-specialized classifier was trained on a randomly selected subset of the corpus. In the case of females, each of the 12 testing folds will utilize a model trained on 11 data points. The specialized classifier will use 11 female data points, and the un-specialized classifier will use 11 randomly selected data points. In this way, we investigate how well a model predicts dialogue acts of a student with and without utilizing learner characteristic information.

### 7.1.2 Unsupervised Dialogue Act Modeling with Learner Characteristics

Our unsupervised dialogue act classification approach leverages the  $k$ -medoids clustering technique (Ng & Han, 1994). This approach groups similar utterances together, and is similar to the more familiar  $k$ -means algorithm except that in  $k$ -medoids, the centroid of each cluster must be

an actual data point within the corpus rather than a potentially artificial data point computed as the mean of distances. This technique is more suitable for modeling with learner characteristics because once the corpus was divided by these characteristics the data sets became too small for query-likelihood to perform well. Our experiments with  $k$ -medoids have demonstrated that it outperforms a variety of other unsupervised machine learning approaches for the task of dialogue act classification in tutorial dialogue, although the results of such experiments are beyond the scope of this chapter since our goal is to investigate the differential benefit of adding learner characteristic features to the model, not to compare different unsupervised approaches.

The  $k$ -medoids algorithm requires seeding clusters at the beginning of each training fold and then proceeds by distributing data points to clusters according to their closest centroids until convergence upon the model. In the standard  $k$ -medoids algorithm, the seeds are randomly selected. However, we employ a greedy seed selection approach intended to mitigate the effects of the unbalanced distribution of dialogue acts in the corpus (Arthur & Vassilvitskii, 2007). Within this greedy seed selection, an initial seed is randomly selected and then each of the subsequent seeds are selected by choosing the point that maximizes its distance from the already-selected seeds. The goal in using this approach is to select the seeds from diverse utterances so the algorithm produces better clusters, and our initial experiments indicated that it substantially improves the model.

In addition to its seeding approach, the  $k$ -medoids approach requires the number of clusters  $k$  to be set prior to model training. To discover the number of clusters, we experimented with  $X$ -Means and Expectation Maximization clustering, both of which attempt to identify the optimal number of clusters. Both of these algorithms converged at four clusters as the optimal choice, so we proceed with  $k=4$ . However, perhaps in part due to the benefit of the greedy seed selection made possible by  $k$ -medoids, these models performed with substantially worse overall accuracy than  $k$ -medoids.

The utterances were represented as vectors with each column matching a token (punctuation and words) in the corpus and each row matching an utterance. There were a total of 877 distinct tokens.

With these parameters in place, first the clusters were formed using each training set, and then for each utterance of the student held out within the leave-one-student-out fold, we computed the closest cluster to that utterance as indicated by average cosine distance to each point in the cluster. The closest cluster was selected as the cluster to which the test utterance belongs, and the majority vote of the cluster was assigned to the test utterance as its dialogue act label. For each leave-one-student-out testing fold, the accuracy was computed by comparing these cluster-assigned labels to the manual dialogue act tags.

## 7.2 Results of Dialogue Act Classification by Learner Characteristics

In this section, we present experimental results for unsupervised dialogue act classification based on learner characteristics. We compare each model built on specific learner characteristics (females and males, low and high self-efficacy students, and students with low and high skill perception) to the models that are built using utterances from randomly selected students, i.e. not utilizing learner characteristic information. Each comparison in this section is conducted with a one-tailed  $t$ -test with a post-hoc Bonferroni correction. The threshold for statistical reliability after the correction has been taken as  $\alpha=0.05$ .

**Gender.** As shown in Figure 7.3, the average leave-one-student-out cross-validation accuracy for the model built using female students’ utterances ( $n_{\text{female}}=12$ ) is higher than the model built on randomly selected students. In each test run, all of one females utterances were left out and the dialogue act model was built on the remaining eleven female students’ utterances. This process was repeated for each female student. Note that for each of the eleven students, all utterances from that student were considered. Average test set accuracy for the model with randomly selected students was 0.41 ( $\sigma=0.2$ ), whereas the average test set accuracy for the dialogue act classification model that was built utilizing female students utterances only was 0.56 ( $\sigma=0.19$ ). After a Bonferroni correction this difference was statistically significant ( $p_{\text{Bonf}} < 0.05$ ).

For male students ( $n_{\text{male}}=30$ ), the average accuracy is only slightly higher with the models tailored to males 0.43 ( $\sigma=0.13$ ) than the models learned for randomly selected students 0.40 ( $\sigma=0.12$ ), and this difference is not statistically significant (Figure 7.3). Looking more closely at the results, we find that for eight of the thirty males within the corpus, a tailored model outperformed the random model (with five of these seeing more than 10% increase in accuracy), while twenty-two of the cases saw no difference in accuracy between the random and tailored conditions. Two of the males saw a decrease in accuracy for the tailored condition.

**Self-efficacy.** Models built using the self-efficacy learner characteristic predict the unseen utterances dialogue acts marginally more successfully than models that do not use this information, as shown in Figure 7.4, though these differences are not statistically reliable. For students with low self-efficacy ( $n_{\text{lowEff}}=24$ ) the average test set accuracy for dialogue act models that selected students randomly is 0.38 ( $\sigma=0.16$ ) and it increases to 0.43 ( $\sigma=0.17$ ) with dialogue act models that learn only from low-self-efficacy students’ utterances (Figure 7.4). In 15 out of 24 cases the dialogue act models tailored to low self-efficacy groups outperform models that are trained on randomly selected students (eight of the cases with more than a 10% increase), while in seven of the cases the performance is decreased by utilizing the learner characteristic (five of them by more than a 5%) and in two of the cases the accuracy remains the same.

The improvement obtained by utilizing learner characteristics in dialogue act classification

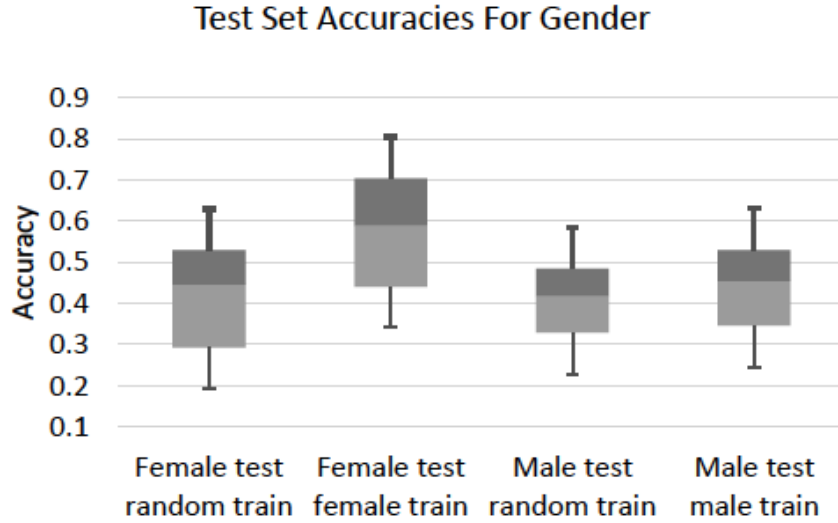


Figure 7.3: Leave-one-student-out test set accuracies for models by gender.

task is also marginal for high-self-efficacy students, where  $n_{nHighEff}=18$ . The average performance for the random model is 0.41 ( $\sigma=0.14$ ) whereas the model achieves 0.47 ( $\sigma=0.11$ ) accuracy when trained only on utterances of high-self-efficacy students. This improvement was statistically significant before Bonferroni correction but not afterward. In seven out of 18 cases, models trained on utterances of high self-efficacy students improved test set accuracy (five of them above 15% improvement) and in two of the cases the learner characteristic decreases the performance (both of them below 5% decrease). Nine of the cases remained unaffected in their dialogue act classification accuracy.

**Skill perception.** For students with low skill perception ( $n_{nLowSkill}=26$ ) the average performance of the dialogue act classification model trained on utterances of randomly selected students is 0.39 ( $\sigma=0.17$ ) whereas the accuracy rises to 0.43 ( $\sigma=0.17$ ) for a tailored model trained only on students with low skill perception (Figure 7.5). This is not a statistically significant difference. For these students, 11 out of 26 cases improved their performance by utilizing the learner characteristic (five of them above 5% and five of them above 15%), six of them were affected negatively (four of them below 5% decrease) and nine of them achieved the same performance.

The same pattern is visible for students with high skill perception ( $n_{nHighSkill}=16$ ). For these students, the average test set accuracy increases from 0.38 ( $\sigma=0.14$ ) to 0.42 ( $\sigma=0.13$ ) gained by using utterances of students with high skill perception rather than learning from utterances



Figure 7.4: Leave-one-student-out test set accuracies for models by self-efficacy.

selected randomly, again not statistically significant after Bonferroni correction. Six of the cases out of sixteen improve test set accuracy (four of them above 15%), three of them degrades (two of them below 5%) and seven cases perform equally.

The average accuracies over the leave-one-student-out cross-validation folds can be found in Table 7.4 (\*\* $p < 0.05$  after Bonferroni correction). Models tailored to learner groups uniformly outperform their counterpart, and the improvement is statistically significant for females.

Table 7.4: Average test set accuracies for each learner characteristic.

Learner characteristic group	Model restricted by learner characteristic	Model built on randomly selected students
Females	<b>0.56**</b>	0.41
Males	<b>0.43</b>	0.40
Low self-efficacy	<b>0.43</b>	0.38
High self-efficacy	<b>0.47</b>	0.41
Low skill perception	<b>0.43</b>	0.39
High skill perception	<b>0.42</b>	0.38

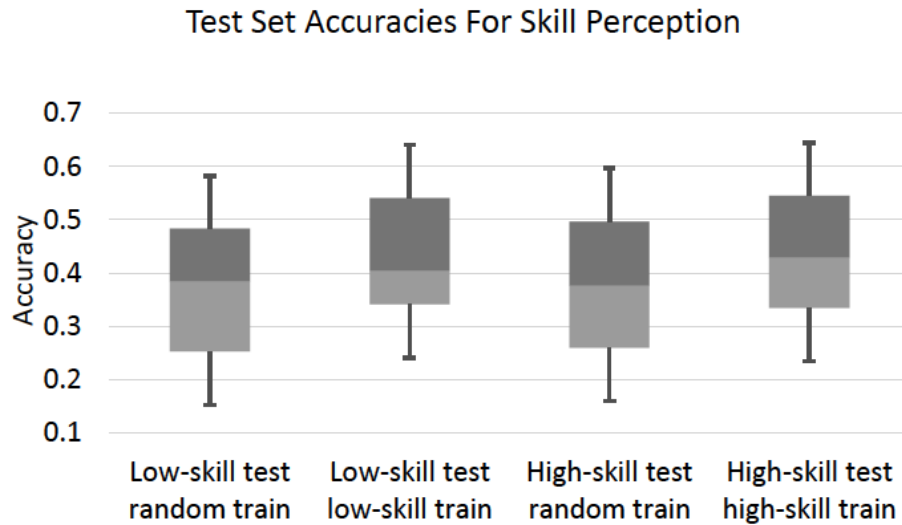


Figure 7.5: Leave-one-student-out test set accuracies for models by skill perception.

### 7.3 Discussion: Learner Characteristics and Dialogue Act Classification

This section first examines clusters from the gender-tailored unsupervised dialogue act classifier. Table 7.5 displays a selection of utterances that were clustered together during the unsupervised training of the model, and afterward the clusters were labeled for testing purposes using the manual tags that comprise the majority of each cluster. For those in Table 7.5 the clusters were labeled as Acknowledgments and Questions. By examining the structure of these clusters we gain some intuition as to the types of regularities that help the tailored models to perform significantly better. We see females in this study tended to use acknowledgment phrases such as, “oh I see” and “makes sense,” while males tended to use the phrasing, “got it” more frequently. Within the cluster labeled as questions, we observe that females tended to request more feedback, an observation that also emerged in prior work within a different corpus in the same domain collected approximately six years earlier (Boyer et al., 2007). On the other hand, male students tended to ask more general questions.

Although the differences in model performance were not statistically reliable for students in different skill perception groups, we observed some interesting patterns within these groups (Table 7.6). (Recall that because of the way the corpus was partitioned, the low and high skill perception groups contain students of both genders.) Students with low skill perception level tended to use more utterances such as, “ok I am getting it,” which may be a type of affective or

Table 7.5: Selected utterances from clusters tailored to gender.

	<b>Females</b>	<b>Males</b>
Acknow.	<ul style="list-style-type: none"> <li>- oh I see</li> <li>- make sense</li> <li>- yup</li> <li>- aha!</li> <li>-hahaha its ok</li> </ul>	<ul style="list-style-type: none"> <li>- got it</li> <li>- ok i got it</li> <li>- alright i got it</li> <li>- gotcha alrighth</li> <li>- cool</li> </ul>
Questions	<ul style="list-style-type: none"> <li>-is this right?</li> <li>-does that work?</li> <li>-should I run it?</li> <li>-so for line number could i have typed system out println monopoly instead of println x if i wanted to?</li> </ul>	<ul style="list-style-type: none"> <li>-so will testing always be related to running the program</li> <li>-so it is kinda like saying x number or something in algebra?</li> <li>-why does not it stop on the next line in this case</li> <li>-oh alright so text or number would be variable type?</li> </ul>

face-saving dialogue move. Students in the high skill perception group seem to exhibit more social, relaxed utterances, reflected by examples such as, “cool cool” and “yeah haha.” There is also an interesting distinction with the topic and structure of questions. Students in the low skill perception group may ask questions that are more related to the task, whereas high skill perception students ask questions that may not be essential for completion of the task, such as about the purpose of code comments. This phenomenon may also be present in statements, in which we notice students with high skill perception more often mention higher order concepts, for example relating the current task to another task, while students with low skill perception tend to remain topically focused on the task at hand.

Finally, we observe some example clusters from the models based on self-efficacy in Table 7.7. Students with high self-efficacy tend to use more confident utterances such as “absolutely” compared to “ok” used by low-self efficacy students. We note that questions in the low self-efficacy group often make an implicit request for reassurance within their task-based questions, such as, “and that is it?” In contrast, students in the high self-efficacy group more often ask contentful questions.

**Limitations.** As the first attempt to leverage learner characteristics within an unsupervised dialogue act classifier for tutoring, the present work has several notable limitations. First, as mentioned previously, the partitions of the corpus are not independent; that is, the same student, and associated utterances, are present within one gender group, one self-efficacy group, and one

Table 7.6: Selected utterances from clusters tailored to skill perception.

	<b>Low Skill</b>	<b>High Skill</b>
Acknow.	<ul style="list-style-type: none"> <li>- oh</li> <li>- ok I am getting it</li> <li>- ok I get it!</li> <li>- interesting</li> <li>- oh ok</li> </ul>	<ul style="list-style-type: none"> <li>-cool cool</li> <li>-yeah haha</li> <li>- yep lol</li> <li>-yep! exciting stuff!</li> <li>- sure</li> </ul>
Questions	<ul style="list-style-type: none"> <li>- what do i do now</li> <li>- can you explain more about the scanner line</li> <li>- so what is this doing exactly?</li> <li>-why is not it prompting me to enter my name?</li> </ul>	<ul style="list-style-type: none"> <li>-comments are just a way to write notes to others to help them understand right?</li> <li>- out of curiosity would not it make sense to switch those last two lines of code?</li> </ul>
Answers	<ul style="list-style-type: none"> <li>-no i believe i am understanding the concept i just have to see it completed a couple of times before i will finally memorize it</li> </ul>	<ul style="list-style-type: none"> <li>-pretty simple it is a lot of code for so little though</li> <li>-i am taking basic fortran right now never seen literal before</li> </ul>

skill perception group. Because these partitions are not independent care must be taken when interpreting the findings. However, we believe that the current approach holds great promise for real-time tutorial dialogue classification. By building separate classifiers by learner characteristic, a suite of classifiers (each smaller and faster than one built on the entire corpus) can be run in parallel and can vote for the classification of a given students utterance. However, as is the case with the work presented here, splitting the corpus results in a substantially reduced sample size on which to train, which partially explains the lack of statistically reliable results observed here. The evaluation study presented in this dissertation work focuses (Chapter 10) on exploring the use of intrinsic metrics for accuracy (rather than relying on manual tags), which has the potential to dramatically increase the available data to any dialogue act classifier and mitigate issues of sparsity that arise when splitting by learner characteristics. Before moving on to evaluation in real time, in the next chapter, we incorporate multimodal features that are extracted from students' posture and gesture during the tutoring session and investigate whether these non-verbal cues improve the dialogue act classification performance.

Table 7.7: Selected utterances from clusters tailored to self-efficacy.

	<b>Low Self-Efficacy</b>	<b>High Self-Efficacy</b>
Acknow.	<ul style="list-style-type: none"> <li>- ok</li> <li>- yes there were a lot of things i felt like i had to switch around</li> <li>- that makes sense now!</li> </ul>	<ul style="list-style-type: none"> <li>-cool</li> <li>-oh ok that works</li> <li>- absolutely</li> </ul>
Questions	<ul style="list-style-type: none"> <li>-so what exactly am i supposed to be doing?</li> <li>- is there something specific i need to call my game</li> <li>- i finished reading should i click compile again?</li> </ul>	<ul style="list-style-type: none"> <li>-what is the best way to do that?</li> <li>- ok so tell me if this makes sense string declares the variable and then line number tells me what that variable is value is?</li> </ul>

## Chapter 8

# Incorporating Multimodal Features into Dialogue Act Models

This chapter extends unsupervised dialogue act classification models presented in Chapter 6 with multimodal features extracted from visual cues obtained during tutoring session (Ezen-Can, Grafsgaard, et al., 2015)<sup>5</sup>. We hypothesize that these features will improve the performance of unsupervised dialogue act models as they were shown to be promising for the supervised dialogue act classification task (Ha et al., 2012). Automatically identifying dialogue acts has long been a goal for dialogue researchers (Stolcke et al., 2000), yet only very recently have multimodal features been considered for this task.

Multimodal learning analytics incorporate features of different categories into learning analytics tasks (Blikstein, 2013). Some categories of multimodal features, including posture (Ha et al., 2012) and facial expressions indicating confusion (Boyer, Grafsgaard, Ha, Phillips, & Lester, 2011), have been found useful for dialogue act classification. However, these prior approaches have relied upon supervised machine learning techniques, which suffer from a manual annotation bottleneck that is problematic for scaling these models across domains or even across corpora. Therefore, how to utilize multimodal features within unsupervised dialogue act models is an important open research question.

This chapter presents the first model to incorporate multimodal features into an *unsupervised* dialogue act classifier for the learning analytics domain. Motivated by the importance of analyzing the *process* of learning rather than the end *product* only (Blikstein, 2011), we analyze posture and gesture features of students in the course of tutoring and utilize this information to enhance our understanding of student dialogue. The results demonstrate that information about students' posture and gesture significantly improves dialogue act classification performance when judged

---

<sup>5</sup>The analysis reported in this chapter was conducted in collaboration with Joseph Grafsgaard, who generated the multimodal features as part of his dissertation research.

against gold standard dialogue act labels. Furthermore, analyses show that utilizing solely posture and gesture lead to better performance than majority baseline chance, even in the absence of any linguistic information about the utterances being classified. This finding highlights the importance of multimodal features for building rich understanding models of student utterances.

## 8.1 Features

The goal of this chapter is to investigate the extent to which posture and gesture features improve unsupervised dialogue act models. Table 8.1 presents an excerpt from the JavaTutor 2011 corpus showing the multimodal features as well as the task actions. By utilizing multimodal features, we aim to improve the performance of dialogue act classifiers for predicting student dialogue acts.

Table 8.1: Excerpt of dialogue from the corpus and the corresponding dialogue act tags.

<p><i>Student modifies code.</i>  <i>Student receives a compile error.</i>  <i>One-hand-to-face gesture recognized.</i>  <b>Student:</b> which do i put first? [<i>Question</i>]  <b>Tutor:</b> try it. [<i>Statement</i>]  <i>Change in head depth detected.</i>  <i>Student receives a compile error.</i>  <b>Tutor:</b> what you had was close. [<i>Statement</i>]  <b>Tutor:</b> go back to that [<i>Statement</i>]  <i>Student modifies code.</i>  <i>Student compiles code successfully.</i>  <b>Student:</b> is the order wrong? [<i>Question</i>]  <b>Tutor:</b> no, the literal is just [<i>Statement</i>]  <b>Tutor:</b> Player's name is [<i>Statement</i>]  <i>Student modifies code.</i>  <b>Tutor:</b> dont put your name [<i>Hint</i>]  <i>Student runs the code successfully.</i>  <b>Tutor:</b> that is excellent. [<i>Positive Feedback</i>]  <b>Tutor:</b> i could tell a lot of learning was going on [<i>Statement</i>]  <i>Change in mid-depth detected.</i>  <b>Student:</b> it's very interesting to me [<i>Statement</i>]  <b>Tutor:</b> good. you are good at it. [<i>Statement</i>]  <b>Tutor:</b> try things. make mistakes. learn. [<i>Statement</i>]  <b>Tutor:</b> one more screen. [<i>Statement</i>]</p>
---

Because the parallel streams (student coding activities, multimodal features, and dialogue) offer rich sources of information, we hypothesize that models of student utterances highly benefit from utilizing these features. Four sets of features were considered within the experiments of this chapter. Three of these sets—lexical, dialogue-context and task features—have been shown to improve unsupervised dialogue act classification in prior work as explained in Chapter 6 (Ezen-Can & Boyer, 2014a). The fourth set consists of multimodal features of posture and gesture.

**Lexical Features.** Words and punctuation of each student utterance are provided to the model. Because the overarching goal of dialogue act classification is to understand learners effectively in real-time systems, features such as part-of-speech tags which are time-consuming to extract and have been observed not to improve the accuracy of some dialogue act models (Boyer et al., 2010) are omitted, leaving only unigrams and word-orderings for consideration.

**Dialogue-Context Features.** Four dialogue-context features shown useful in prior work (Ha et al., 2012) are included in the model: utterance position in relation to the beginning of the dialogue, utterance length, author of the previous dialogue message (tutor or student), and previous tutor dialogue act. Because in a tutorial dialogue system the tutor moves are system-generated, their dialogue acts are known. We use the previous tutor dialogue act as feature in our models. This type of dialogue history has been shown effective for dialogue act classification (Ezen-Can & Boyer, 2014a; Litman & Pan, 2002; L. Chen & Eugenio, 2013; Samei, Li, Keshtkar, Rus, & Graesser, 2014).

**Task Features.** The parallel task stream present in tutorial dialogue is a rich information source that may not be directly represented in the dialogue. This stream consists of task actions, in our case compiling, running of code, changing code and sending messages. Utilizing these features can help capture the whole dialogue in a more comprehensive manner. To do this, we use interaction traces between tutors and students to obtain task features that can help the dialogue act classification task (Ezen-Can & Boyer, 2014a). The programming activities logged throughout the course of tutoring include the previous task action preceding each student utterance (composing an utterance, writing/compiling/running code), the status of the most recent coding action (begin, success, error, stop, input\_sent), number of messages sent since the beginning of the task, and number of errors present in the student code.

**Posture Features.** Four posture features are utilized: head distance (distance between camera and head), mid torso, lower torso, and the average of these three features (Grafsgaard, Fulton, Boyer, Wiebe, & Lester, 2012) as shown in Figure 8.1. Approximately eight frames per second were recorded from a Kinect depth camera. However, utterances occur less frequently. Because the granularity of posture features are different from granularity of utterances, representation constitutes a challenge. We take the average of the feature values ten seconds before an utterance

and ten seconds after the previous utterance, which is the minimum granularity that allows us to observe change in the features.

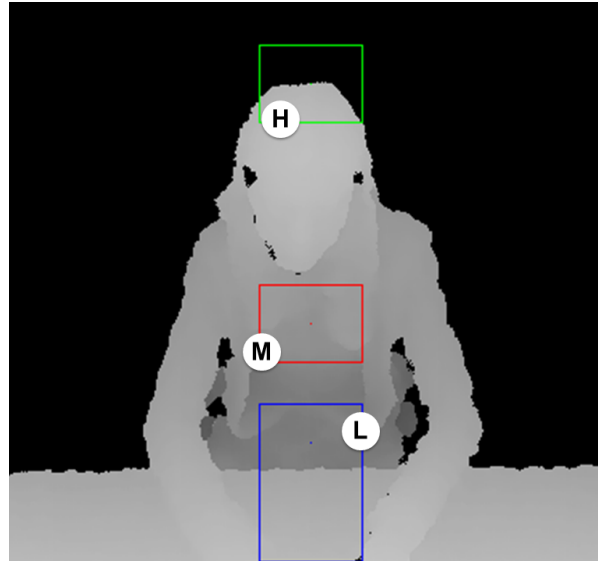


Figure 8.1: Output of the posture algorithm.

***Gesture Features.*** The gesture features include two different hand-to-face features: one-hand-to-face (see Figure 8.2) and two-hands-to-face (see Figure 8.3) indicating the hand positions of students (Grafsgaard, Wiggins, Boyer, Wiebe, & Lester, 2013). For matching the gesture features to utterances, we count the number of values detected between two utterances within a ten-second frame. For instance, for a particular utterance, the number of times the one-hand-to-face feature gets detected after the previous utterance of that particular utterance is counted which allows us to match gesture features to each utterance.

## 8.2 Methodology: Clustering with Multimodal Features

For unsupervised classification of dialogue acts, we used a framework that calculates similarities between utterances using their longest-common-subsequences and then utilized those similarities within  $k$ -medoids clustering as explained in Chapter 6 (Ezen-Can & Boyer, 2014a). For features other than the lexical features (task, dialogue-context and multimodal features) we used Cosine similarity, which captures similarity independent of the length of utterances. To use the similarities within clustering, the number of clusters needed to be selected.  $k=5$  was found to be the optimal

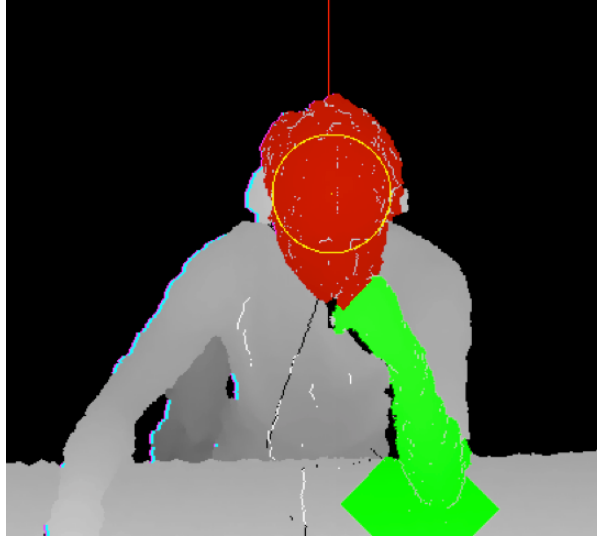


Figure 8.2: Output of the gesture algorithm showing the one-hand-to-face feature.

number of clusters in prior work by using the Bayesian Information Criterion, which penalizes the number of parameters the model uses (Ezen-Can & Boyer, 2014a).

In addition, our approach for representing dialogue history (see Chapter 6), which was shown to significantly improve upon the prior performance of unsupervised dialogue act models, was adopted in this chapter (Ezen-Can & Boyer, 2014a). Specifically, we branched the clustering model by student utterances according to the previous tutor dialogue acts. Nine branches of student utterances were formed, one for each tutor dialogue act. In this way, the student utterances in the training set were clustered while taking the previous tutor move into account. Each branch had student utterances that shared the previous tutor dialogue act and therefore were more granular for clustering. Then, clustering was performed within each branch. Note that each student utterance was clustered only with utterances that had the same previous tutor dialogue act.

***Classifying test utterances.*** Once we had the clusters that were produced using the branching and clustering technique in the training set, each unseen utterance from the test set was classified using the model created in training. For each utterance in the test set, we chose the branching that it should follow in the existing model according to its previous tutor dialogue act. Having chosen the branching, the average distance between the target utterance and each cluster in the clustering group was calculated, where the clustering group represented all clusters in that particular branch. The distance from the target utterance to utterances in each cluster was calculated and divided by the number of utterances in each cluster, producing one average distance to each cluster. The closest cluster which had the smallest average distance

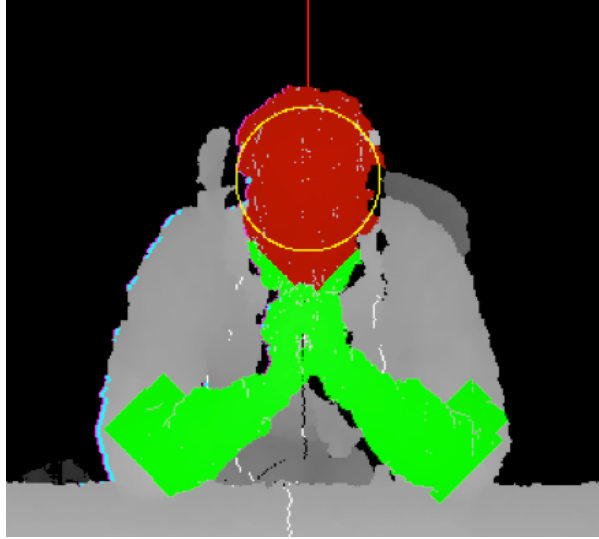


Figure 8.3: Output of the gesture algorithm showing the two-hands-to-face feature.

determined the target utterance’s dialogue act. For instance, if the previous tutor dialogue act of the test utterance was a statement, then the utterance was modeled within clusters that shared the same previous tutor dialogue act in the training set. The process is depicted in Figure 8.4 where the student utterance to be classified is  $u_i$  with its posture and gesture features  $p_i$  and  $g_i$  respectively. The branching was done based on the previous tutor utterance of  $u_i$  ( $u_{t-1}$ ) and the chosen branch was used for clustering  $u_i$ . Using this framework which has been shown to outperform previous state-of-the-art unsupervised dialogue act classifiers (Ezen-Can & Boyer, 2014a), the experimental results (Section 8.3) will demonstrate the additional benefit of using multimodal features for dialogue act classification.

**Distance metric.** For calculating similarities between utterances, we took word ordering into account to better capture the underlying intentions of each utterance. As an example, consider two utterances ‘I should declare a variable’ and ‘should I declare a variable’. These two utterances have the same set of words when compared with a bag-of-words approach that does not take the order of words into account. However, the first utterance is a statement whereas the latter is a question. To distinguish them, it was necessary to take the word ordering into account. We utilized *longest common subsequence* (Hirschberg, 1975), shared subsequences of not-necessarily contiguous words between utterances, to calculate the similarity between two utterances considering word ordering, as described in Chapter 6 (Ezen-Can & Boyer, 2014a). Unlike any distance metric that does not exploit utterance ordering information (Cosine, Euclidean, Manhattan, Jaccard), these two sentences are considered different by longest common subsequence. This discriminative power is desirable.

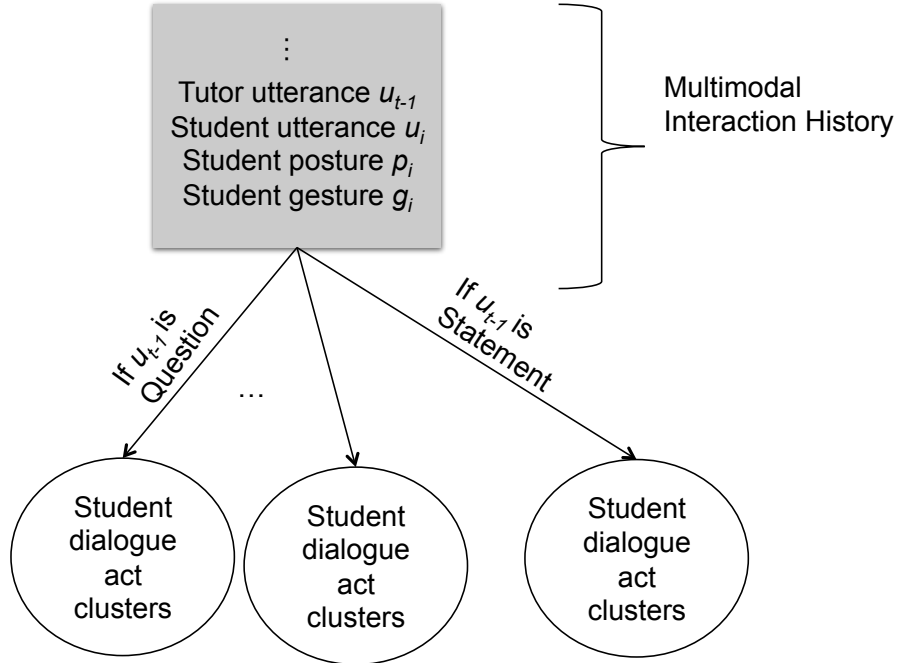


Figure 8.4: Branching student utterances according to previous tutor dialogue act and choosing which clustering group to use for unseen utterances.

### 8.3 Experiments: Clustering with Multimodal Features

The goal of unsupervised dialogue act classification is to group together utterances with the same dialogue act. There are different techniques to accomplish this in an unsupervised way including  $k$ -means clustering (Rus et al., 2012), Dirichlet process mixture model (Crook et al., 2009) and query-likelihood clustering (Ezen-Can & Boyer, 2013b). For this chapter, we utilized  $k$ -medoids clustering because our prior work has established that this technique outperforms its counterparts for our corpus (see Chapter 6) (Ezen-Can & Boyer, 2014a).

We hypothesized that including posture and gesture information would improve dialogue act classification performance significantly. Therefore, we conducted experiments with and without these features. We created unsupervised dialogue act classifiers that utilized posture and gesture features as well as models that did not use these features. To investigate how these models compared to each other, we compared the performance of models with the same test sets. For instance, we compared how well the utterances of a student in the test set were classified using the model having access to multimodal features and using the model that did not take this information into account. In this way, we aim to draw conclusions on the importance of multimodal features for dialogue act classification.

For testing, leave-one-student-out cross-validation was performed: for each fold, each student’s utterances were either all in the test set or all in the training set, but not in both. To evaluate how well the model performed for each unseen utterance, we computed test set accuracy. Test set accuracy calculates how well the clustering model classifies the label of unseen utterances. Accepting the closest cluster as the cluster of the test utterance (as described in Section 8.2), the majority vote of the cluster was given as the label to the test instance. The average accuracy for the test set was computed as the number of correct classifications divided by the number of utterances in the test set. The formula for test set accuracy is as follows where  $n$  is the number of utterances in each fold of the test set and  $c_i$  is the cluster of utterance  $i$  ( $u_i$ ):

$$\frac{\sum_{i=1}^n \text{majority label of } c_i = \text{label of } u_i}{n}$$

Because we applied leave-one-student-out cross-validation, we took an average of all students (folds) to report average test set accuracy. The  $t$ -tests were conducted comparing each classifiers’ performance (with and without multimodal features) for each student.

## 8.4 Results: Clustering with Multimodal Features

This section presents experimental results for unsupervised dialogue act classification based on multimodal features. We compared models built separately using posture and gesture features to models that did not have access to this information. Each comparison in this section was conducted with a one-tailed  $t$ -test for  $n = 37$  students. The threshold for statistical reliability was taken as  $p = 0.05$ .

The leave-one-student-out cross-validation accuracies with respect to manual dialogue act labels were statistically significantly better with the addition of posture and gesture features ( $p < 0.05$ ). The average accuracy for the model without using multimodal features was 61.8% ( $\sigma = 2$ ) and this number increased to 67% ( $\sigma = 1.9$ ) with the inclusion of multimodal features, 8% improvement. The confusion matrices for both cases are shown in Figures 8.5 and 8.6. Less frequent dialogue acts were eliminated from the confusion matrix because the model never predicted those acts (“Request for Feedback” and “Other”).

The experimental results show that including posture and gesture features improved unsupervised dialogue act classification performance significantly. For the most frequent dialogue acts (statements, answers, questions and acknowledgments), only statements’ classification accuracy decreased with multimodal features. In order to gain better insights and understand which dialogue acts benefit more from multimodal features, we compared the two models qualitatively.

We observed that for distinguishing questions that are very similar to statements in structure,

		Predicted				
		S	A	Q	ACK	C
Manual Label	Dialogue acts					
	S	161	38	24	63	2
	A	22	594	2	4	0
	Q	59	70	35	43	0
	ACK	114	43	28	103	0
	C	8	3	1	1	0

Figure 8.5: Confusion matrix for the model *without* posture and gesture (61.8% accuracy).

multimodal features are highly beneficial. In contrast, when multimodal sensors detect features that might be indicative of confusion, although students may utter statements, the models decided that they asked questions requesting help. The nature of the corpus is highly influential here: because the students are engaging in dialogue while completing a learning task, nonverbally expressed confusion may relate to the learning task and not necessarily be indicative that the student is expressing a question dialogue act. Table 8.2 (shown on the next page) depicts sample utterances which were incorrectly classified with the model that did not utilize posture and gesture features and were corrected with the help of multimodal features. We provide five types of corrections in the table, two of which were more frequently seen: questions misclassified as statements and questions misclassified as acknowledgements. These results show that utilizing posture and gesture features, the dialogue act classifier became more successful in distinguishing questions. Especially for utterances that were not syntactically questions such as “so the computer reads it from right to left?”, multimodal features helped enrich the information present in the utterance by incorporating information about students’ posture and gesture. For acknowledgements that were corrected from statements with the help of multimodal features, the students were closer to the workstation (according to lower torso distance) and both one-hand-to-face and two-hand-to-face gestures were present.

Table 8.3 shows sample utterances that caused the dialogue act classification model to be confused with the addition of multimodal features, i.e. utterances that were classified correctly with the model that did not have access to multimodal features but were incorrectly classified

		Predicted				
		S	A	Q	ACK	C
Manual Label	Dialogue acts					
	S	139	38	38	71	2
	A	15	594	6	7	0
	Q	38	69	68	32	0
	ACK	58	43	34	153	0
	C	7	3	0	3	0

Figure 8.6: Confusion matrix for the model *with* posture and gesture (67.05% accuracy).

with the addition of these features. Most of the misclassifications caused by multimodal features were on questions. Comparison between two models showed that increase in student’s mid or lower torso depth i.e., students moving farther from the camera, or one-hand-to-face gesture detection increases the chances of the model classifying the utterance as a question because this pattern is seen in other questions as well. Therefore, even though an utterance may have a statement label, observing students moving farther from the computer triggered a question classification. That may be one of the reasons why a decrease in the accuracy of statements was observed when the multimodal features were incorporated.

Another important finding of the experiments is that when posture and gesture were used with no other features, the average cross-validated accuracy was 53.2%, whereas the majority chance baseline was 43%. This finding suggests that, even before knowing the content of an utterance, it is possible to predict the dialogue acts by analyzing multimodal features of students. This information can be especially helpful for systems that aim to provide remedial support without an explicit request from students. Being able to predict what the next dialogue act would be even before the student utters words can be a significant advantage for understanding students.

A notable limitation of the current approach is that collection of posture and gesture data is not yet a fully scalable approach. However, given the continued decrease in the cost of high-resolution video equipment, these approaches are expected to become more scalable. Additionally, work in building affect detectors suggest that it might be possible to infer these multimodal

events based on streams that do not require expensive sensors (Paquette et al., 2014).

In this chapter and previous chapters on dialogue act classification (Chapter 4, 5, 6, 7) the performance of the models were compared against manual labels. This approach has the limitation in that it restricts the models to only imitating gold standard and not leaving room for flexibility. Therefore, this dissertation project evaluated an unsupervised dialogue act classification model in real time and compared its performance with a version that relies on a supervised classifier. In the next chapter, we first present preliminary in-context evaluation of an unsupervised classifier that motivates the implementation of the tutorial dialogue system described in Chapter 10.

Table 8.2: Sample utterances that were correctly classified with the help of multimodal features and their incorrect classifications by the model that did not utilize posture and gesture.

Sample Student Utterances From Clusters
<b>ACK utterances misclassified as S without multimodal features</b>
<i>ok i am getting it</i>
<i>that makes sense</i>
<i>awesome thank you</i>
<i>ok got it</i>
<i>got it! I just thought it is an example</i>
<b>Q utterances misclassified as S without multimodal features</b>
<i>why is not it prompting me to enter my name?</i>
<i>are we going to learn how the player enters their name next? there is no box to type it in</i>
<i>do I have to?</i>
<i>just means to assign the name write?</i>
<i>so what happens if I do not put what the java is expecting</i>
<i>just comments are just a way to write notes to others to help them understand right?</i>
<i>how do you run it? the run button is not available to press</i>
<i>so the computer reads it from right to left?</i>
<i>name a variable first and then store the value for what I want to call it?</i>
<i>so the contents that come after string are what exactly? declaring a variable?</i>
<i>so anything that someone types in the comments box that will be used with the scanner?</i>
<i>would you still put the prompt after those lines of codes or should you move it up to prompt the user right after you print the game name</i>
<b>S utterances misclassified as ACK without multimodal features</b>
<i>not exactly sure how to go about this one</i>
<i>beautiful!</i>
<b>S utterances misclassified as Q without multimodal features</b>
<i>I guess I am not sure what the codes are currently displayed under the java code section</i>
<i>does not have a problem quitting</i>
<b>Q utterances misclassified as ACK without multimodal features</b>
<i>so you want me to redo number but change the adreamgame name to something else?</i>
<i>could I type in string the adventure quest? or would I need to put in quotes or something?</i>
<i>so I could have put the system.out.println command on line number and the input statement on line number and it would still work?</i>
<i>should I do another player input code?</i>
<i>spaces or no spaces?</i>
<i>oh so I need to insert it before the scanner player input line</i>

Table 8.3: Sample utterances that were incorrectly classified when multimodal features were used but were correctly classified by the model that did not use posture and gesture features.

<b>Student Utterances Correctly Classified with the Help of Multimodal Features</b>
<b>ACK utterances misclassified as Q with multimodal features</b>
<i>ok so I just ask please give me your name</i>
<i>ok I get it now</i>
<i>I understand now</i>
<i>think I got it</i>
<i>I know I was trying to figure out what line it comes from</i>
<i>oh i see</i>
<b>S utterances misclassified as Q with multimodal features</b>
<i>just pops up in blue</i>
<i>closest experience I have to java is playing runescape</i>
<i>sorry to take too long time, I am usually not creative at all so it usually takes long time to think about something</i>
<i>but I think I got it</i>
<i>totally guessed what I needed to do</i>
<i>I am not understanding</i>
<i>well that didn't turn out right</i>
<i>and the name of the variable is the only other thing I could think of</i>

## Chapter 9

# Preliminary In-Context Evaluation of Unsupervised Dialogue Act Classifiers

The previous chapters have described novel unsupervised dialogue act classification approaches, with evaluation that relied upon manual tags and qualitative inspection of the clusters. Such evaluation approaches are used in all prior evaluations of unsupervised dialogue act classifiers (e.g., (Crook et al., 2009; Rus et al., 2012; Ezen-Can & Boyer, 2013b)). Relying on manually tagged dialogue act labels to evaluate an unsupervised model has two major drawbacks: it does not fully avoid the manual annotation bottleneck, and it imposes a hand-authored criterion onto a fully data-driven model, which may be unnecessarily limiting. Distinctions made by an unsupervised model may be useful within a dialogue system, even if these categories are different from the distinctions made within a hand-authored dialogue act tagset.

This chapter presents the next step forward, consisting of completed work to evaluate the usefulness of the unsupervised dialogue act classifier within a simulated tutorial dialogue system context (Ezen-Can & Boyer, 2013a). The goal of this investigation is to evaluate how well an unsupervised dialogue act classification technique can be used to select tutor moves in a tutorial dialogue system, prior to the implementation of the system itself. To this end, a study was conducted in which human evaluators compared the tutor moves selected using unsupervised dialogue act model and using manual dialogue act labels.

### 9.1 Preliminary Evaluation: Methodology

In this section, the unsupervised dialogue act clustering technique and the evaluation framework are explained. Instead of attempting to evaluate the model intrinsically, we evaluate its perfor-

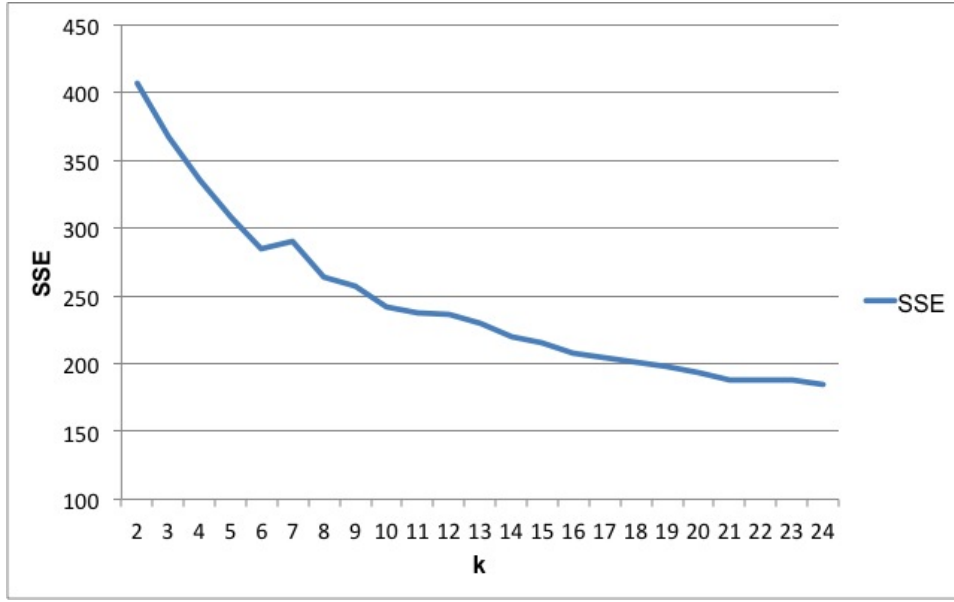


Figure 9.1: Sum of squared errors graph.

mance on an external task: triggering an appropriate utterance via a simple dialogue policy. This evaluation, which does not require an end-to-end dialogue system, judges the model in the simulated context of the target task. The results demonstrate that this in-context evaluation may be equally useful as comparing against gold standard dialogue act labels, while substantially reducing the time required for human annotation.

### 9.1.1 Unsupervised Dialogue Act Classifier Being Evaluated

The unsupervised dialogue act model evaluated here is based on the query-likelihood clustering (Ezen-Can & Boyer, 2013b) described in Chapter 4. Each utterance within the training set is queried against all other utterances within the training set using bigram features. In the work presented in this chapter we add to the feature vectors the first level of the parse tree as provided by the Stanford parser (Klein & Manning, 2003).

The number of clusters was selected based on sum of squared errors (SSE). As with many parameterized models, model fit tends to increase with more parameters, but there are important tradeoffs in computation time and risk of overfitting. In experiments,  $k$ =number of clusters ranged from 2 to 24. 21 clusters were chosen, corresponding to the rightmost “knee” within the SSE graph (see Figure 9.1).<sup>6</sup>

<sup>6</sup>Selecting the number of clusters is a subjective decision. Nonparametric techniques, such as variations on Dirichlet process clustering, hold promise for addressing this limitation in the future.

### 9.1.2 Evaluation Framework

Evaluating unsupervised dialogue act clusters presents numerous challenges. In prior evaluations of query-likelihood clustering, we computed accuracy with respect to the manually applied dialogue act tags described earlier, demonstrating 41.64% accuracy for a model with 8 clusters, compared to 34.90% accuracy for the Rus et al. (2012)  $k$ -means approach and 24.48% accuracy for Dirichlet process clustering (Crook et al., 2009) on JavaTutor 2007 corpus as described in Chapter 4. However, the goal of the work presented within this chapter is to substantially reduce the human tagging required to evaluate the model. We also aim to test the hypothesis that comparing against manual labels under-represents the utility of the unsupervised model. That is, a dialogue policy built on the unsupervised model could perform better than the relatively low classification accuracy for manual tags would suggest. Our evaluation will explore this hypothesis.

In order to achieve these goals, we first trained an unsupervised dialogue act model on 75% of the corpus using the query-likelihood approach described in Chapter 4. The resulting model has 21 clusters. Then, we handcrafted a dialogue policy for tutor responses by qualitatively examining each cluster of training data and creating one tutor response for each cluster. Some clusters and their corresponding tutor utterances are depicted in Figure 9.2. This policy was applied by classifying unseen utterances from a held-out test set (25% of the corpus) using the learned model (Figure 9.3). The result of this process is that for each student utterance from the test set, a tutor response is generated based on the policy. This process resulted in 373 student utterances, one for each utterance in the 25% testing set, each paired with a corresponding tutor response generated by the hand-authored policy.

The evaluation goal is to determine whether the responses made by this policy are reasonable, which will represent the utility of the unsupervised dialogue act model for its intended use within a dialogue manager. We used human judges to rate the output of the policy. Thirty student utterances and tutor responses were randomly selected from the available utterances generated by the test set. An example set of utterances and policies can be seen in the Table 9.1. These items were placed in a survey that asked the reader to rate the extent to which each tutor response makes sense given the student utterance. (One item was inadvertently omitted from the survey, resulting in 29 items that were evaluated by the judges and that will be analyzed here.) To avoid bias introduced by the ordering of items, they were presented in a different randomized order for each of the seven judges who completed the survey. (29 items from a comparison condition using manual tags were also randomly interleaved into the survey, as described later in this section.) Judges used a rating scale from 1 to 4 (*1=makes no sense, 2=makes a little sense, 3=makes a lot of sense, and 4=makes perfect sense*). Since the models only used the current student utterance, the dialogue history was also not shown to the human raters.

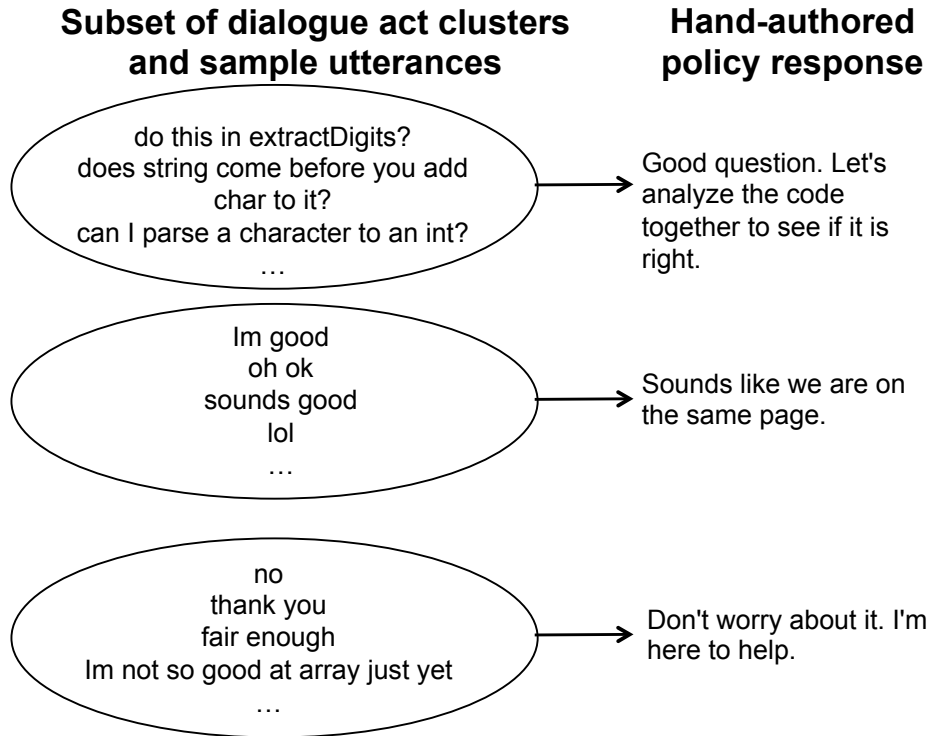


Figure 9.2: Clusters from unsupervised dialogue act modeling and corresponding dialogue policy.

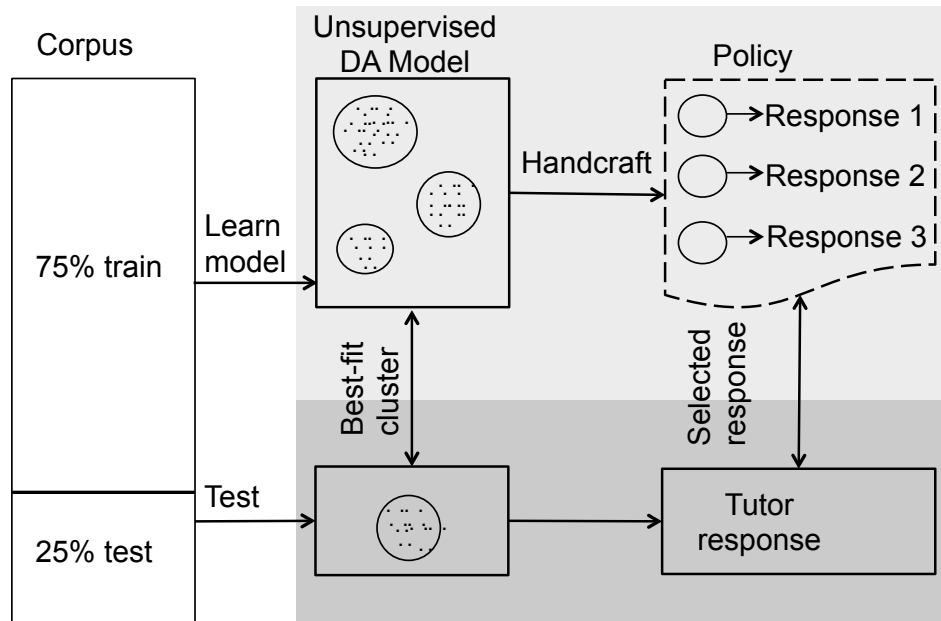


Figure 9.3: Evaluation framework structure.

Across the seven judges, the average rating of the tutor responses selected by the unsupervised policy was 2.35. We also collapsed the ratings into positive ( $\geq 2.5$  average across seven judges) and negative ( $< 2.5$  average). With this binary categorization, 44.8% of the time tutor responses generated by the unsupervised policy were rated positively. It is important to note that no information other than dialogue act was considered for generating the tutor responses; the tutor utterances were relatively content-free and based only on the dialogue act categorization given by the unsupervised model.

For comparison, we also constructed a hand-crafted dialogue policy using the manual dialogue act labels and applied this policy to the same utterances as were used to evaluate the unsupervised model. These pairs of student utterances and tutor responses were interleaved randomly on the same survey provided to seven human judges. The same tutor responses as in the unsupervised policy were used whenever possible for this manual-tag policy. The tutor responses generated from the manual-tag policy received an average score of 2.22, slightly lower than the average of 2.35 for tutor responses generated by the unsupervised policy. The binary positive-negative split for these ratings reveals that 31% were rated positively ( $\geq 2.5$  average), compared to 44.8% for the unsupervised policy.

Table 9.1: Example student utterances and tutor responses.

<b>Student Utterance</b>	<b>Tutor Response (Unsupervised Policy)</b>	<b>Tutor Response (Manual-Tag Policy)</b>
I'm trying to think, heh	Don't worry about it. I'm here to help.	That part is tricky.
how can I pull values out of an array or can I reference them with code like zipDigits[1]?	Great question. Let's look at the task decription together to see if it can help.	Good question. Let's analyze the code together to see if it is right.
thanks for the reminder	I'm here to help!	Great, seems like we agreed.
does that mean I should declare it at the top of the code?	Maybe you should try it out.	Good question. Let's analyze the code together to see if it is right.

## 9.2 Preliminary In-Context Evaluation: Results

Direct comparisons between the unsupervised policy and the manual-tag policy must be interpreted with caution, in part because the unsupervised policy was more granular (based on 21 clusters) than the manual-tag policy (based on 9 tags) and also because it can be difficult to ensure that the two policies were of equal quality. On the other hand, the unsupervised policy utilized no manual labels and was applied to an unseen test set, while the manual-tag policy was based on reliable tags applied to the actual utterances from the testing set.

Finally, we evaluated the extent to which the 4-category rating scheme was reliable across judges. The weighted Kappa (J. Cohen, 1968), used for ordinal scales because it penalizes disagreements less if they are closer together, was 0.30 averaged across all pairs of judges, indicating fair agreement (Landis & Koch, 1994). For the collapsed binary ratings, average pairwise ordinary Kappa was 0.36.

## 9.3 Preliminary In-Context Evaluation: Discussion

It was hypothesized that evaluating an unsupervised dialogue act model against manual labels may be an inappropriately strict metric, requiring the model to conform to the criteria used by humans to handcraft the manual tagset. Indeed, the accuracy of the unsupervised dialogue act model presented here with 21 clusters was 30.4% for identifying manual labels (arrived at by assigning the majority class tag to each unsupervised cluster after clustering was complete). The majority class baseline (most frequent student dialogue act tag) was Evaluation Question with a relative frequency of 25.87%, so on accuracy for identifying manual labels, the unsupervised model improved modestly over baseline. In contrast, when this unsupervised model was used to select a tutor response within a dialogue policy, the response was judged positively 44% of the time by human judges. Moreover, recall that the tutor responses were content free and took only the dialogue act label into account (no information state or topic). Therefore, it is meaningful to consider what percent of the time the responses were rated as making some sense (receiving a 2, 3, or 4 rating average across the human judges). By this criterion, 65.5% of tutor responses selected by the unsupervised policy were rated as sensible.

Finally, this evaluation approach demonstrates promise for alleviating the bottleneck of manual annotation for dialogue act models. Each item within the current evaluation survey required approximately 15 seconds to judge, using untrained human judges, for a total of approximately 1 hour of effort across all seven judges. The time required for handcrafting policies was relatively small, approximately 1 hour. In contrast, the dialogue act annotation scheme required approximately 35 seconds per utterance (amortizing substantial up-front training time for each annotator) when applied as part of previous work, for a total of approximately 50 hours

per annotator.

The results demonstrate that in-context evaluation of an unsupervised dialogue act model, rather than accuracy against manual labels, may better reflect the usefulness of the model for dialogue management which motivates the tutorial dialogue system of this dissertation work. Furthermore, this evaluation technique may greatly reduce the time required by human judges to evaluate the model. However, the best evaluation is a full end-to-end system as described in the following chapter.

## Chapter 10

# A Tutorial Dialogue System for Real-Time Evaluation of Unsupervised Dialogue Act Classifiers

As promising as unsupervised dialogue act models are for improving natural language interactions in tutorial dialogue systems, to date, no deployed dialogue system has utilized an unsupervised dialogue act classifier; rather, researchers have evaluated the performance of unsupervised models as standalone components by comparing to manual labels (Rus et al., 2012; Ezen-Can & Boyer, 2014a) as we have also done in previous chapters. The downside of this approach is that expecting a fully data-driven model to replicate a human annotation scheme may not capture how well that data-driven model will perform in an end-to-end deployment. Therefore, evaluating unsupervised dialogue act models without comparing to manual annotations and within their usage environment is of the utmost importance.

As part of this dissertation project, we have implemented a tutorial dialogue system that can be utilized to compare the performance of two different dialogue act classifiers within a real-time system (Ezen-Can & Boyer, 2015b). We trained an unsupervised dialogue act model on a corpus of human tutorial dialogue in the domain of introductory computer science, and for comparison we trained a supervised model. We hypothesized that the unsupervised dialogue act model, which relies on hierarchical clustering and uses no manual labels during model training, would support better student learning than the supervised dialogue act model, which relies on a decision tree classifier and represents a state-of-the-art, highly accurate dialogue act classifier that agrees with human annotations 89.6% of the time (Vail & Boyer, 2014a). We have conducted

two studies ( $n_1=51$  and  $n_2=47$ ) to evaluate the system. The experimental results show that unsupervised and supervised dialogue act models achieved similar performance for supporting learning gains and user satisfaction. This chapter describes the system and in later chapters, we will present the evaluation (Chapters 11 and 12).

## 10.1 System Design

The primary goal of the study is to evaluate an unsupervised dialogue act classifier in its intended usage environment and to compare it to a supervised classifier within a tutorial dialogue system. This chapter describes two versions of a tutorial dialogue system: one that implements an *unsupervised* dialogue act classifier (JavaTutor-U) and one that uses a *supervised* dialogue act classifier (JavaTutor-S). A screenshot is shown in Fig. 10.1.

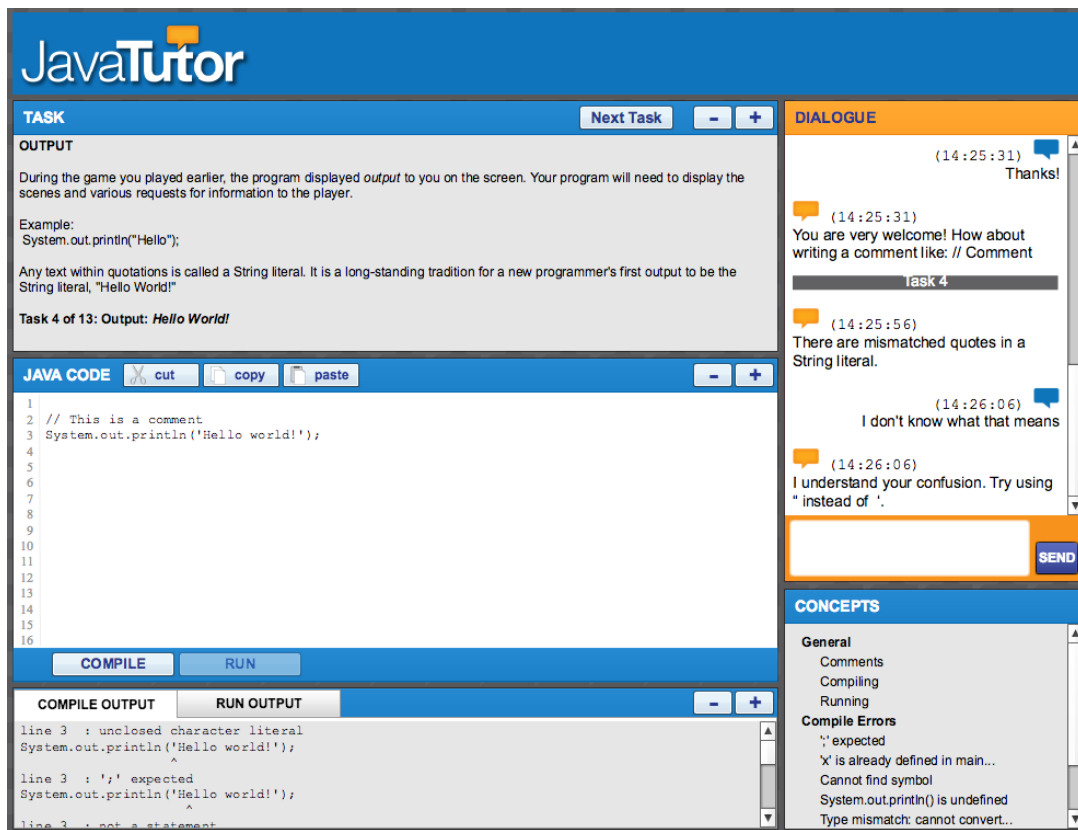


Figure 10.1: Screenshot from the tutorial dialogue system.

Both versions of the tutorial dialogue system depend on the same pipeline. First, dialogue

act classification takes place to interpret student’s words. Then, a code analysis module utilizes regular expressions to identify errors in student’s code. The output of these modules is used in the dialogue manager, where the tutor move is determined. The tutorial dialogue system teaches Java programming language to novice students. Fig. 10.2 presents the architecture diagram of the dialogue system. The following subsections provide details on this system architecture.

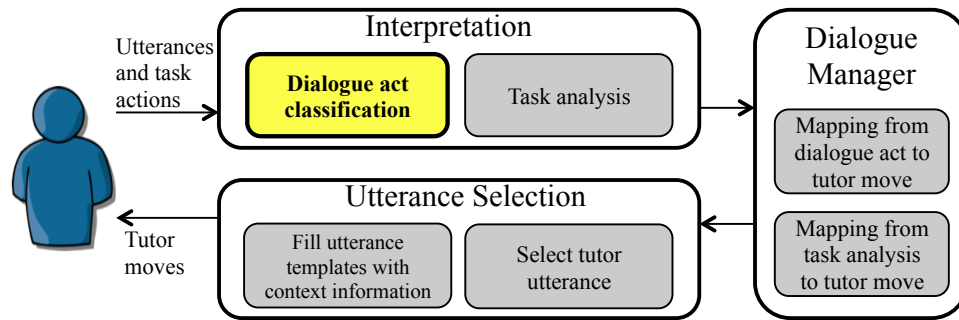


Figure 10.2: System architecture diagram.

In this chapter, we explain how the dialogue act classification task is handled within the system. Both dialogue act classifiers were trained on a corpus of 2,417 student utterances collected in a computer-mediated environment for teaching Java programming language within a study which has been detailed in previous publications (Ha et al., 2012). First, the features were extracted, which were then used as inputs to the dialogue act classifiers (supervised or unsupervised).

**Features.** For classification of dialogue acts, we extract two sets of features: textual and task-related. These are the same set of features extracted both for training (building the dialogue act classifiers) and testing (real-time classification of dialogue acts). The textual features are solely extracted from the student utterances. These include unigrams and bigrams of both tokens (words and punctuation) and part-of-speech tags.

While experimenting with the dialogue interactions, students were asked to complete the tasks provided. To satisfy the requirements, the students wrote and tested their programming code. The task-related features were extracted from the task events that occurred in real-time throughout the tutoring. There were three task-related features used within the system. Two of them were utilized within dialogue act classification and one of them was used to provide remedial help. We used the latest task action (compile, run, writing a message to the tutor) and its result (success, error, begin, stop) to improve the dialogue act classification task. In addition, we used regular expressions to compare the student’s code to the solutions of previous students

to understand whether the student’s code had an error.

**Supervised Dialogue Act Classification.** For supervised dialogue act classification, we used an off-the-shelf decision tree classifier from Weka (Hall et al., 2009) and trained it on dialogue act tags (Vail & Boyer, 2014c). This tagging scheme consists of 18 student dialogue act labels for the portion of the task that the system implements (*Greeting, Extra-Domain Question, Ready Question, Confirmation Question, Direction Question, Information Question, Observation, Correction, Understanding Feedback, Not Understanding Feedback, Explanation, Other, Yes-No Answer, Extra-Domain Answer, Answer, Positive Feedback, Ready Answer, Acknowledgement*), with Cohen’s Kappa of  $\kappa = 0.87$  (89.6 % agreement) showing high reliability (Vail & Boyer, 2014c).

**Unsupervised Dialogue Act Classification.** As for the unsupervised dialogue act classification, we utilized a hierarchical clustering approach that assigns utterances to individual clusters initially and merges the most similar two clusters in each iteration until the hierarchy is completed by having one large cluster. By examining the whole hierarchy, we qualitatively chose the stopping point where the groupings of utterances make sense (31 clusters). The number of clusters determined at this stopping point would have been the number of clusters to be used if no comparison with a supervised classifier were to take place. However, our goal is to provide similar conditions to both of the supervised and unsupervised classifiers. In order to make sure that the number of clusters are not different from the number of manual labels, we merged the clusters that were sparse, i.e., had less than 10 utterances, according to their similarity and used the same number of clusters (18) as the number of manual tags.

### 10.1.1 Tutorial Policies and Utterance Generation

Having obtained the machine-learned models, we authored policies which govern system moves given the output of the classifier. For the supervised version, we authored moves for each manual label (e.g., Question; Negative Feedback) and for the unsupervised version, we crafted tutor moves to each cluster (with clusters interpreted qualitatively).

When students interact with the system, it calls upon the trained dialogue act classifier to classify each new student utterance. Then based upon the dialogue policy, the system chooses tutor moves on the fly. Additional features provided for generating tutorial moves include the output of automated code analysis using regular expressions, which compares student’s code to previous correct student solutions to understand if there are any errors. The output of this code analysis is used to fill slots within the tutorial move templates. Example tutor moves are depicted in Fig. 10.3. In addition to the dialogue act classifiers’ support, the system takes initiative to provide feedback when needed; that is, it is not constrained to respond to student requests (Ha et al., 2012). This task-based feedback is the same across both the supervised and

unsupervised versions of the system.

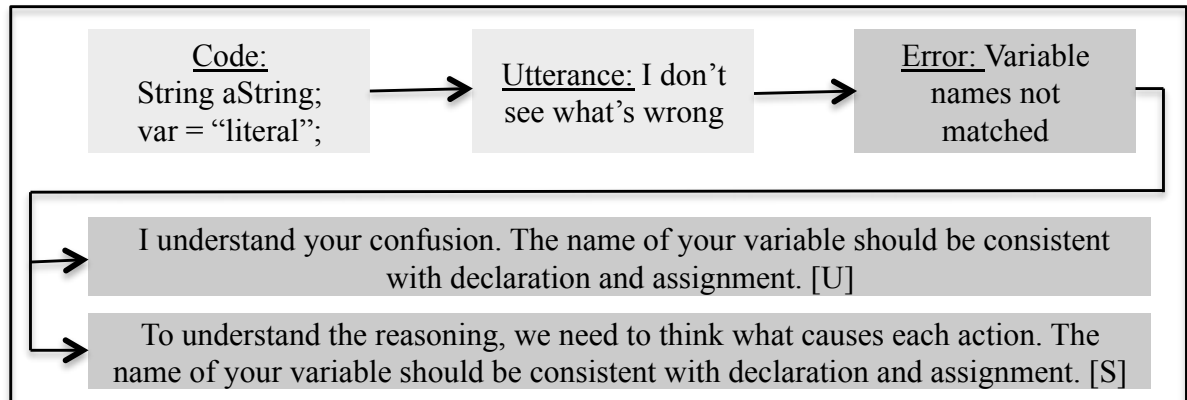


Figure 10.3: Sample tutor moves with sample code and utterance. [U] indicates the unsupervised dialogue act classifier move and [S] indicates the supervised classifier move.

In cases where there are multiple code errors returned by the regular expression module, we apply prioritization of errors. We provide the tutor move that corresponds to the latest error, which is determined by the line number of the error. In addition, less priority is given to the task errors, such as *variable declaration not found* although the task description asks for a variable description, than syntax errors such as trying to declare a variable but could not successfully do so syntactically.

In addition to the classifiers, we incorporate a hybrid approach which makes use of simple rules. The motivation for utilizing rules is that for some phenomena such as greetings and thanking, the responses are clear; therefore the classifiers do not need to be run. If a student utterance falls into one of these categories, the approach returns the corresponding moves from the rules. For more complex utterances, we run the dialogue act classifiers.

Table 10.1: Sample tutor moves from the dialogue policy in the JavaTutor-U condition.

Student Utterance from Cluster	Tutor Response
yep, ohhhh, cool	I'm glad we are on the same page! Ok.
thank you	Good, let's move on. Nice observation.
yay!, yup!	Good. Yup!
ok thanks	How exciting! Nice.
tough decision	Learning takes time. Practice makes perfect
there seems to be an error	I understand your confusion... We can overcome this together.
I'm not understanding	Let's go step by step. I can try to help.
compile	I like the way you analyze each phase. Good observation.
I believe it is just the way the computer reads human input	That part is not trivial.  The best way to reply that question is just to try it!
not really	Learning takes time, keep trying. Focusing on the line of the error might help.
ok that makes sense	Nice.
I'm not sure where to start	Let's work together on the parts that are not clear to you. We need to focus on the parts that is not very clear.
which do I put first?	Trying it out might be a good idea. You can start with trying it out.
not sure if this is right	I see where you might be confused.
it says line 3 : ' ; ' expected	You might want to track line by line. We cannot know without trying it out.
I'm good	Sounds good.
but I think I got it	Good, let's work as a team on this. Great.
ok cool	You are learning fast.

Table 10.2: Sample tutor moves for each dialogue act in the JavaTutor-S condition.

Dialogue Act Tag	Student Utterance	Tutor Response
Acknowledgement (ACK)	ok	Alright.
Answer-Extra Domain (AEX)	same again	Good to hear that.
Answer Ready (AR)	oh yes	Great, let's move on.
Answer WH (AWH)	seems ok	Great question.
Answer YN (AYN)	No I think I got it once I realized you had to switch lines	Good.
Correction (CO)	string*	Thanks for the correction.
Explanation (E)	Well obviously it has to be either 1 or 2	Nice explanation.
Feedback-Not Understanding (FNU)	interesting, but not sure i know what i did	I like it that we are moving further together.
Feedback Positive (FP)	yes	Good.
Feedback Understanding (FU)	Okay I get it!	Good. I like it so much when learning happens.
Greeting (GRE)	hello	Greetings!
Observation (O)	I forgot the exclamation point	Nice observation.
Other Extra Domain (OEX)	oh no problem i am actually enjoying this	That's true.
Question Confirmation (QC)	because the computer doesn't know to look for playerInput?	To be able to confirm, let's analyze the code together.
Question Direction (QD)	Should i just remove whatever is already there?	Let's think about how to proceed.
Question Extra Domain (QEX)	wait do you mean when it asks for 1 or 2 type in a 9 or change the if statement to 9?	What an exciting question.
Question Information (QI)	what was the scanner thing for? i don't really understand why it took two lines to ask for the name.	To understand the reasoning, we need to think what causes each action.
QuestionReady (QR)	yes!	Sure, I am ready to move on.

Table 10.3: Sample tutor moves for each hand-written rule. The rules are the same for both JavaTutor-U and JavaTutor-S.

<b>Rule Name</b>	<b>Utterance Samples</b>	<b>Tutor Move</b>
Bye	bye	Bye!
Default		Hmm...
Greeting	hello, hi, howdy	Hello!
Help	help	Let me see how I can help you.
How	How do I write a comment?	Maybe the concepts pane can help you decide how to <i>&lt; TASK &gt;</i> .
Howareyou	How are you	Fabulous! I hope you are doing good too.
next	what's next?	The goal of this subtask is <i>&lt; TASK &gt;</i> . If you feel that you accomplished it, you can move on. If not, give it a try.
Thank	thank you, thanks	You are very welcome!
Why	Why doesn't it stop on the next line?	Thinking about the causes of actions is a good learning experience.
Question	Was playing the game included in running the game as well?	Good question...

Table 10.4: Regular expressions for capturing errors in student’s code and their corresponding tutor moves.

<b>Regular Expression Name</b>	<b>Tutor Move</b>
Argument in function call	It is sufficient for us to call <code>nextLine()</code> method, without writing anything between parentheses.
Comment slashes wrong	You can try placing the comment slashes at the beginning of the line.
Dot missing in function call	Why don’t we make sure that we have the expected dots in line <code>&lt; LINE_NUM &gt;?</code>
Double equals sign	While assigning a value, try using <code>=</code> instead of <code>==</code> .
Double quote missing	Make sure you are not using <code>'</code> instead of <code>”</code> .
Missing semicolon	Let’s make sure we have the semicolons in place, for example in line <code>&lt; LINE_NUM &gt;</code> .
No comment found	How about writing a comment like: <code>// Comment</code>
No scanner declaration found	As the next step you might want to declare your scanner.
No scanner input statement found	By using scanner, how about getting input from the user?
No variable assignment found	How about assigning a value to your variable?
No variable declaration found	Maybe you can declare a variable next.
Object lower case	For keywords <code>String</code> and <code>Scanner</code> , we start with capital letters.
Parenthesis or brackets mismatch	You may want to check that you have a closing bracket every time you open one.
Print statement not found	Why don’t you try writing a print statement next?
Print string not found	Make sure you print the string in the task description.
Variable name duplicate	We can not have multiple variables with the same name.
Variable name not match specification	Interesting variable name.
Variable name not matched	The name of your variable should be consistent with declaration and assignment.

## Chapter 11

# Evaluation Study I: In-Class Participants

As part of this dissertation project, we conducted two evaluation studies. The first using participants who interacted with JavaTutor in a mandatory class activity. As this chapter will discuss, the results highlighted important differences that may exist between volunteers versus in-class participants. Chapter 12 will describe the volunteer study.

We have hypothesized that our unsupervised dialogue act classifier will support better student learning and satisfaction compared to a state-of-the-art supervised dialogue act classifier. To test this hypothesis we conducted a study with two conditions: 24 participants were in the supervised condition and 27 participants were in the unsupervised condition. The students (12 female, 39 male) were drawn from a university-level first-year-engineering class and participated as part of an in-class activity lasting 2 hours.

The students were randomly assigned to the different versions of the system. Their interactions with the system were logged. They took a pre-test and pre-survey, and after tutoring completed a post-survey and post-test identical to the pre-test. The pre-survey consisted of several widely used measures including goal orientation (VandeWalle, Cron, & Slocum Jr, 2001), general self-efficacy (G. Chen, Gully, & Eden, 2001), and confidence in learning computer science and programming (C. Lee & Bobko, 1994). We investigate the relationships of these measures to system outcomes in Section 11.1.

Students in both conditions received a statistically equivalent number of tutor messages: 35.2 in the supervised condition and 38.7 in the unsupervised condition. To compare the effectiveness of the two systems, we use multiple metrics from the surveys and tests. First, we consider usability as indicated by a set of ten items on the post-survey (e.g., ‘The tutoring system was knowledgeable about programming.’, ‘The tutoring system was supportive.’) (please see

Appendix D for the post-survey). There was no significant difference between two versions of the system with respect to usability, with both sets of users rating the system 2.99 out of 5 ( $p=0.5$ ). The second metric we compare is students' perception of how effective the tutor feedback was. This measure is taken from fourteen post-survey questions (e.g., 'It was easy to learn from the tutor's feedback.', 'I paid attention to the tutor's feedback.'). Similar to the usability questions, the tutor feedback ratings in supervised and unsupervised versions were not significantly different ( $p=0.4$ ).

Finally, we compare the systems in terms of learning gain. We use normalized learning gain to compute how much students learned taking their prior knowledge into account. The normalized learning gain formula is as follows:

$$\text{normalized learning gain} = (\text{posttest score} - \text{pretest score}) / (1 - \text{pretest score}) \quad (11.1)$$

The learning gains were significantly positive in both conditions ( $mean_{sup}=0.12$ ,  $p=0.05$ ;  $mean_{unsup}=0.14$ ,  $p=0.0009$ ) and these means were not significantly different ( $p=0.3$ ). While the mean in the unsupervised condition was slightly higher, there is not sufficient evidence to support Hypothesis 1.2 that the unsupervised dialogue act classifier supported higher learning gain than the supervised one. However, the unsupervised model required only a small fraction of the manual labor (for interpreting clusters) compared to the supervised model (for which extensive labeling of the corpus was required). Next we build descriptive regression models to examine the relationships between pre-measures and post-measures.

## 11.1 Evaluating System Outcomes

Since the unsupervised and supervised dialogue act classifiers produced comparable tutorial dialogue interactions, we aggregated the data from the two conditions in order to explore the factors affecting the outcome of the system. We leveraged the dialogue system evaluation framework PARADISE (Walker, Litman, Kamm, & Abella, 1997) and built multiple regression to reveal relationships between student characteristics and fine-grained logs from the tutorial dialogue interactions (the predictors), and students' perceptions of the tutorial dialogue (the response variables).

We built one multiple regression model for predicting each post-measure of interest from the surveys or tests (endurability, curiosity, felt involvement, focused attention, task difficulty and post-test score), with the same set of independent variables each time that include pre-survey (see Appendix B), pre-test (see Appendix C), number of utterances written by the student, number of total logged activities, number of compile/run events, number of program content

changes logged, number of compile errors and number of tutor messages received. The goal was not to obtain accurate predictive models necessarily, but to investigate *descriptive* models that indicate the aspects of the learners or of the interactions that are significantly associated with outcomes.

As shown in Fig. 11.1, the outcomes of the system were related not only to the effectiveness of the tutoring, but also heavily to incoming perceptions of the students. In the figure, CS stands for computer science, † represents  $p < 0.005$ , all others are  $p < 0.05$  and task difficulty has a range from 0-100, all others 1-5. We present features with significance  $p < 0.05$  and the post-measures they predict. The endurance category had four post-survey items which measure the extent to which the students considered the tutoring session worthwhile and rewarding. The felt involvement category consisted of three survey items measuring how much the students were involved in the task, and the focused attention category involved seven questions about how much the students were focused on the task. Finally, the curiosity category measured how interested the students were with the system using three questions.

The results show that some predictors were correlated with multiple system outcomes. For example, the confidence that students have in computer science was significantly predictive of endurance, curiosity, felt involvement and task difficulty. Similarly, the extent to which students find computer science useful was significantly associated with all presented post-measures except task difficulty, highlighting the fact that the perceptions of students are related to how they feel about the tutoring system. Learning goal orientation (mastery vs. performance) was measured with twelve pre-survey questions, and the models showed that students that were willing to face challenging tasks also felt more involved with the learning task. Finally, students who aimed to score higher than other students felt that the tasks were more difficult.

In addition to these student characteristics or attitudes, several aspects of the tutorial interaction were correlated with outcomes. The number of logged activities, code changes and compile/run events were all positively correlated with the number of utterances written by students throughout the session. One might argue that these measures are correlated with the outcomes because they are also correlated with pre-test scores, which affects the system outcome. However, pre-test score was not significantly correlated with any of the predictors in the regression analyses.

The results suggest that the perceptions of students even before starting the tutoring session are indicative of the outcomes of the system. Such observations are harmonious with those from prior studies (Jackson, Graesser, & McNamara, 2009; S. D’Mello, Williams, Hays, & Olney, 2009). The findings highlight the importance not only of honing the tutorial dialogue within interaction to be as effective as possible, but to adapting to the characteristics and attitudes that students bring in to the tutoring session. In fact, an important limitation of this work arises from students’ attitudes and preferences: namely, as students participated as part of an in-class

### Descriptive Linear Regression Models

<b>Endurability</b>	= 0.4028 * CS confidence + 0.4279 * CS usefulness - 0.7784 * self-efficacy
<i>My learning experience was rewarding.</i>	
<b>Curiosity</b>	= 0.5480 * CS confidence + 0.3681 * CS usefulness
<i>The content of the tutoring system incited my curiosity.</i>	
<b>Felt involvement</b>	= 0.4724 * CS confidence + 0.3171 * CS usefulness - 1.5409 * self-efficacy <sup>†</sup> + 0.7083 * learning goal orientation
<i>I was really drawn into my learning task.</i>	
<b>Focused attention</b>	= 0.4744 * CS usefulness <sup>†</sup>
<i>I blocked out things around me when I was working.</i>	
<b>Task difficulty</b>	= -9.6884 * CS confidence + 9.0752 * achievement goal orientation <sup>†</sup> - 0.1358 * number of compile/run events + 0.4956 * number of tutor moves <sup>†</sup>
<i>How mentally demanding was the task?</i>	

Figure 11.1: Multivariate linear regression analyses for describing the outcomes of the system using measures from pre-survey and from tutorial interaction.

activity and knew that their final product would not be graded for quality, they typically tried to finish as quickly as possible and made fewer than desired utterances to the system. Accounting for and mitigating these types of issues with tutorial dialogue systems is a crucial area for the field as we aim to provide one-on-one adaptive tutoring to very broad populations of students with varying levels of intrinsic motivation toward the task.

## 11.2 Exploring the Relationship Between Learner Personality, Attitudes, and Tutorial Dialogue Participation

As we have just seen, numerous factors appear to influence the outcomes of tutoring. Study 1 has revealed an unexpected phenomenon: many students chose not to interact with JavaTutor's dialogue at all. Students were instructed that they could make comments, pose questions, and request feedback at any time through textual dialogue. Overall, students interacting with JavaTutor achieved significant learning gains from pre-test to posttest (average= 12%, median= 13.4%, stdev = 32%,  $p = 0.001$ ). However, we observed that 58.8% of students never made an utterance. For students who did engage in dialogue with the tutor, the average number of

utterances was 5.1 (stdev=7.36, median= 2). Regardless of the extent to which they chose to engage in natural language dialogue, all students received some tutorial dialogue utterances based upon the system’s model of feedback for task events. Our goal is to identify the factors that may be influential in students’ levels of interaction with the system. To this end, we built multiple regression models. The remainder of this section describes the analysis.

This section presents an investigation into the relationship between student characteristics and interactions with a tutorial dialogue system (Ezen-Can & Boyer, 2015a). We hypothesized that students’ personality profile, for example their tendencies toward extraversion or openness, would be significantly associated with the level of natural language interaction observed within a tutorial dialogue system. We also hypothesized that students’ attitudes toward the learning task would be a significant factor in their interactions with the system. These hypotheses were based on literature. Tutorial dialogue systems effectively support learning through rich natural language dialogue (S. K. D’Mello, Lehman, & Graesser, 2011; VanLehn et al., 2002; Litman & Silliman, 2004; Dzikovska et al., 2014). However, the effectiveness of tutorial dialogue systems, like other adaptive learning environments, depends in large part on students’ willingness to interact with them (VanLehn et al., 2007). Interaction varies tremendously across individual students and student populations. We observe various types of *disengagement* including lack of motivation or interest for the learning task (Forbes-Riley & Litman, 2013), as well as *gaming* an intelligent tutor by exploiting properties of the learning environment (Baker et al., 2009; Beck, 2005).

In addition to these factors, individual differences such as self-reported interest in the task and confidence in learning have been found to be strong predictors of engagement (S. D’Mello et al., 2009). Similarly, students’ hidden attitudes toward learning (Arroyo & Woolf, 2005) and motivation for the task (Beal, Qu, & Lee, 2008) may be highly influential. Boredom, which is associated with reduced motivation to perform the activity (Pekrun, Goetz, Daniels, Stupnisky, & Perry, 2010), has been positively correlated with attention problems and negatively correlated with performance. Students’ participation in tutorial dialogue has also been found to be associated with the students’ expectations (Jackson et al., 2009), and in human-human tutorial dialogue, student personality traits have recently been found to be significant factors (Vail & Boyer, 2014b). However, the field is far from a full understanding of the factors that influence students’ choices to engage or interact with tutorial dialogue systems. These prior work in the literature motivated our analysis explained in the next section.

### 11.2.1 Analysis

Regression models were built that predict both dialogue and task participation by the students, who have the choice to interact with the dialogue system as little or as much as desired over the

course of the learning tasks. The models demonstrate that students' attitudes and personalities are significantly predictive of their willingness to interact with the tutorial dialogue system. The findings suggest that some learner characteristics may put students at risk of low participation with a tutorial dialogue system, and constitute a first step toward proactively adapting the systems to benefit these learners.

### **Response Variables**

Based upon the logged interaction traces, we extracted dialogue and task events and used them to compute a numeric representation of the student's level of interaction with the system. For dialogue interaction we utilized the *number of utterances* written by each student. The range of number of student utterances was between 0 and 33.

We extracted four features that represent interaction of students with the system throughout tutoring. The first of these four features is *number of content changes* which refers to the changes in the student's programming code, as the code they write is referred to as content pane. We also computed *the number of compile events* and *number of run activities*. The number of compile/run events ranges from 4 to 224, whereas the number of content changes ranges from 88 to 1099 to complete the series of learning tasks.

Finally, we computed the *number of tutor messages* each student received. The tutoring systems provided students with feedback. The number of messages received is closely related to the number of actions that triggered tutor feedback, which is also a measure of participation. The minimum number of tutor messages provided to any student was 8, whereas the largest number of tutor messages to a student during a tutoring session was 121. We built separate multiple regression models to predict level of dialogue interaction and level of task interaction.

### **Predictor Variables**

We hypothesized that several learner characteristics were significantly associated with level of interaction in the system. We provided these variables for selection within the models (see Table 11.1). All of the predictors were standardized to a common scale before model building.

#### **11.2.2 Modeling Level of Participation**

We built separate models for each of the response variables (number of utterances, compile/run events, content changes, received tutor messages). For each response variable we used the whole dataset and selected features via stepwise linear regression. Because the goal was to investigate relationships between pre-measures (student characteristics, attitudes) and level of participation, we conducted descriptive analyses using the entire data set for model building. Matlab was used for experiments and the interaction terms were included to the regression model automatically.

The model for number of dialogue utterances (Table 11.2) revealed that students' failure avoidance characteristic is a significant predictor of tutorial dialogue interactivity. Students who indicated that they tend to avoid tasks in which they may have higher chance of failure wrote fewer utterances to the system.

The model for number of compile/run events during tutoring session showed that students' personality scores, particularly the binary agreeableness score, was a significant predictor of participation from a task-related perspective. The students who were more agreeable (indicated as a 1 for the model, rather than a 0) made more task interactions considering compile/run events as shown in Table 11.3. The other regression model having the number of content changes as a response variable did not produce significant results.

Another regression model that showed significant results was the regression model that predicted the number of tutor messages students received. Interestingly, both student perceptions (computer science confidence and motivation) and personality (openness score from Big Five Inventory) were selected by the model as shown in Table 11.4. There was a negative correlation between computer science confidence and tutor messages, however it was the opposite for computer science motivation. The students who were more motivated to study computer science interacted more with the system, triggering more tutor messages. Also, the students who had low confidence towards programming received less tutor feedback. Figure 11.2 shows the scatter plots for both computer science motivation and confidence measures.

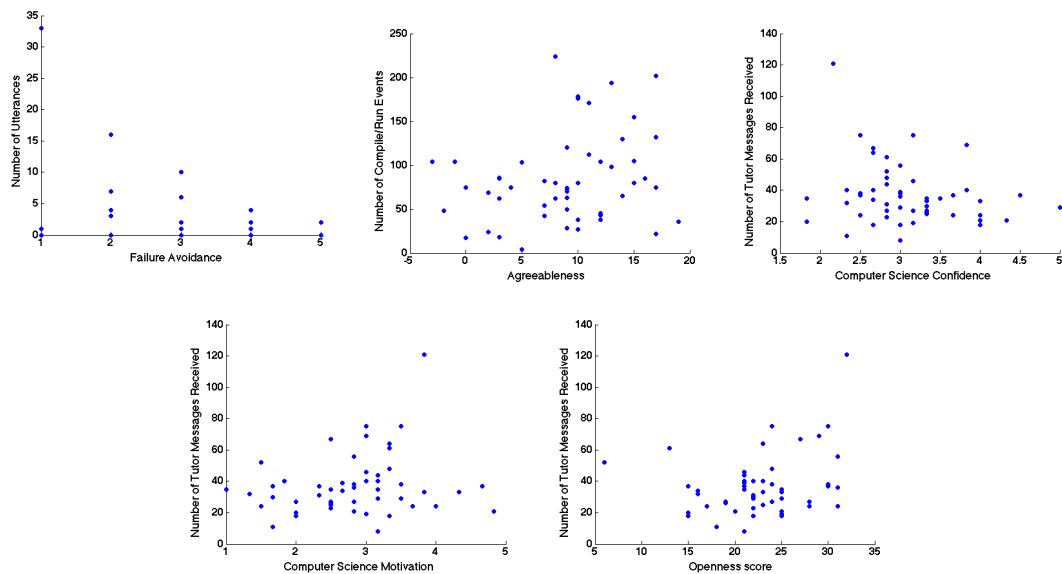


Figure 11.2: Scatter plots of various predictors and response variables.

**Discussion.** Understanding how student characteristics are associated with tutorial dialogue interaction holds great promise for identifying possible disengagement types and taking adaptive action during tutoring sessions to further improve learning effectiveness. The results of the models indicate that as hypothesized, student characteristics such as personality profile were significantly predictive of the student’s level of interactivity with the tutorial dialogue system. We found that students’ attitudes and personalities have significant relationships with their level of participation in terms of both dialogue and task.

Another important finding was that although pre-test was present in all regression models as an independent variable, it was not significantly predictive of either the number of utterances or the task activities. In other words, the level of participation was more correlated to student characteristics than to their incoming knowledge. These results are important for understanding how to better foster interaction with intelligent tutoring systems. If we can identify students who tend to participate less or become disengaged, the system can automatically adapt to these students with scaffolding. For instance, when a student with low motivation toward the task is identified, the tutorial dialogue system might put particular emphasis on moves that are part of “adjacency pairs,” such as asking a question and awaiting a student response. Adapting the task may also be appropriate in these cases. By utilizing information that we can glean from quick pre-measures, we may be able to significantly improve the effectiveness of the system.

While this study gave insight into factors which may influence students’ not to engage with a dialogue system, it was important for addressing this dissertation’s hypothesis to collect data where students interacted with the dialogue system more. Therefore we conducted study 2 described in the next chapter.

Table 11.1: Predictor variables from pre-survey and pre-test.

Predictor variable	Example survey item/ Description
Computer science confidence	<i>I am sure that I can learn programming.</i>
Perceived computer science usefulness	<i>I'll need programming for my future work.</i>
Motivation toward computer science	<i>Programming is enjoyable and stimulating to me.</i>
General self-efficacy	<i>I will be able to achieve most of the goals that I have set for myself.</i>
Learning goal orientation	<i>I often look for opportunities to develop new skills and knowledge.</i>
Performance demonstration	<i>I like to show that I can perform better than my coworkers.</i>
Failure avoidance	<i>Avoiding a show of low ability is more important to me than learning a new skill.</i>
Achievement goals	<i>It is important for me to do better than other students.</i>
Gender	Male/female
Age	Age of the student
University class standing	The year that the student is in the university
Perception of student's own computer skill	<i>How skilled are you with computers, compared to the average person?</i>
Extraversion	<i>I see myself as someone who is talkative.</i>
Agreeableness	<i>I see myself as someone who is helpful and unselfish with others.</i>
Conscientiousness	<i>I see myself as someone who does a thorough job.</i>
Neuroticism	<i>I see myself as someone who is depressed, blue.</i>
Openness	<i>I see myself as someone who is original, comes up with new ideas.</i>
Pre-test score	Score showing the performance of the student before tutoring session

Table 11.2: Stepwise linear regression model for the number of utterances.

Number of utterances =	Coefficient	<i>p</i>
Failure Avoidance	-0.3089	0.0274
~1 (intercept)		1
<b>RMSE = 0.961</b>		
$R^2 = 0.0954$		

Table 11.3: Stepwise linear regression model for number of compile/run events.

<b>Number of compile/run =</b>	<b>Coefficient</b>	<b><i>p</i></b>
Agreeableness (binarized)	0.2897	0.0392
~1 (intercept)		1
<b>RMSE = 0.967</b>		
$R^2 = 0.0839$		

Table 11.4: Stepwise linear regression model for number of tutor messages received.

<b>Number of tutor messages =</b>	<b>Coefficient</b>	<b><i>p</i></b>
Age	0.3802	0.0033
Computer science confidence * Openness	-0.5244	0.0008
Computer science motivation * Openness	0.5317	0.00006
~1 (intercept)		1
<b>RMSE = 0.739</b>		
$R^2 = 0.52$		

## Chapter 12

# Evaluation Study II: Volunteer Participants

The first study, reported in the previous chapter, was conducted as an exercise substituted for an in-class lab activity in the CSC 200 course at North Carolina State University. As previously discussed, this setup had important ramifications for the data collected. First, while students were able to opt out of the research study, they were required to attend the lab for their class. Second, the students knew that their programming solution would not be graded for quality, nor would the extent of their interactions with the tutor count toward their class grade. As a result of this, many students tried to complete the programming exercise as quickly as possible, and many chose not to interact with the tutorial dialogue system at all, as detailed in Section 11.2.

To address these issues, we conducted a second study the following semester in which students in a related course, Engineering 115, were given extra credit if they participated in a study outside of class. A total of 47 students (9 females, 38 males) completed all parts of the study (from pre-survey to post-survey). We expected that these students would be substantially more motivated to interact with JavaTutor. While research questions surrounding how best to support motivation for students are very important, in the present work we focus on the effectiveness of dialogue act understanding assuming a basic willingness on the part of the student to engage with the system.

To recruit volunteers, an email was sent to students asking for volunteers without any programming experience to participate in a study that teaches Java programming language. Students who participated in the study were offered extra credit in their course. Students attended in person and the study sessions were scheduled according to students' availability. Students were required to come only once, for a total of approximately 2 hours. The students were given a pre-survey and a pre-test prior to starting the tutoring study. Each student was

assigned a unique participant id and a password. The tutoring session was limited to 50 minutes and was followed by a post-test identical to the pre-test as well as a post-survey. The students were randomly assigned to either of the two conditions: the version of the system that relied on an unsupervised dialogue act classifier (JavaTutor-U) and the version of the system that used a supervised dialogue act classifier (JavaTutor-S). We calculated normalized learning gains, user satisfaction and tutor feedback evaluation separately for each condition to compare versions.

As was the case for the first study, we hypothesized that the version of the system that relied on an unsupervised dialogue act classifier would perform better than the supervised version in terms of multiple criteria. The criteria we have analyzed include learning gain, user satisfaction and students' evaluation of tutor feedback. In this chapter, we will first provide results for these three criteria and then analyze the participation aspect to understand the underlying reasons that could have produced these results.

## 12.1 Learning Gains in JavaTutor-U and JavaTutor-S

We used the score difference between pre-test and post-test as an indicator of how much each student learned from the tutoring session. The normalized learning gain represents how much each student learned given their prior knowledge and we used the same formula in Chapter 11.

For the learning gain criteria, we saw that JavaTutor-U performed slightly better than JavaTutor-S, but not with statistical significance. The students who interacted with the unsupervised version of the system had higher average learning gains ( $\text{avg}=0.37$ ) than the supervised version ( $\text{avg}=0.29$ ) and the pre-test scores between conditions were not significantly different. In the unsupervised version (JavaTutor-U), 80.77% of students achieved positive learning gain with an average of 119.46% improvement from pre-test score to post-test score whereas in the supervised version of the system (JavaTutor-S), 80.95% of students obtained a positive learning gain with an average of 107.94% improvement. Considering all students, JavaTutor-U helped students to achieve an overall 95.63% improvement from students' pre-test scores to post-test scores and JavaTutor-S was able to obtain 82.68% improvement. The percent test score improvements and learning gains are higher for JavaTutor-U than JavaTutor-S, but the evidence is not enough to support a statistically significant result.

The bar charts (Figures 12.1 and 12.2) show the changes in students' pre-test scores and post-test scores. In the supervised version, 19.04% of students had a less than or equal to 0 learning gain and similarly in the unsupervised version it was 19.23%. We observed that some students were already knowledgeable in Java programming although our participation email requested for students without any prior knowledge in Java. Three students in the unsupervised condition and 4 students in the supervised condition performed higher than 75% correct in the pre-test. Because taking this as an evidence of a possible ceiling effect, we removed the students

from the dataset and compared the learning gains in each condition afterwards. Figure 12.3 shows box plots comparing two conditions according to normalized learning gains and Figure 12.4 depicts box plots for students when we removed the participants that performed high in the pre-test, since this performance indicates a possible ceiling effect for the pre-test and the learning gains may not be as meaningful for these students.

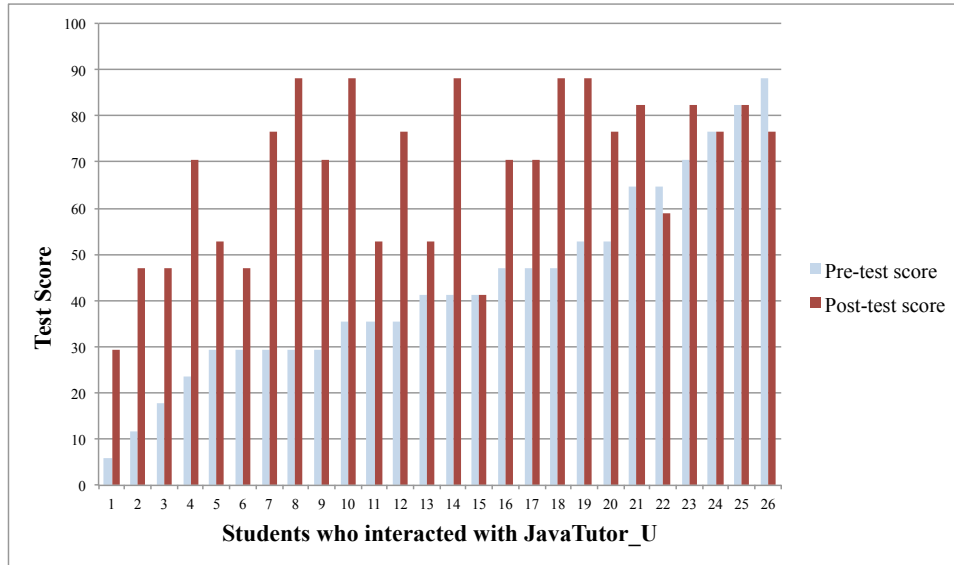


Figure 12.1: Bar charts visualizing students’ pre and post-test scores in the unsupervised condition.

## 12.2 User Satisfaction in JavaTutor-U and JavaTutor-S

For calculating user satisfaction and students’ evaluation of the system, we made use of post-survey questions (please see Appendix D). Both the system usability metric and tutor feedback evaluation metric resulted in similar outcomes for JavaTutor-U and JavaTutor-S ( $p=0.50$  and  $p=0.30$  respectively). The boxplots in Figure 12.5 and Figure 12.6 provide detailed scores of both conditions. Both versions of the system received higher scores for system usability and tutor feedback when we compare this final study and the in-class lab exercise (pilot study). While the system usability was 2.99 in the first study, the voluntary second study achieved 3.59 out of 5 in both conditions. Similarly for evaluation of tutor’s feedback, the mean score increases from 3.00 to 3.47 for the unsupervised condition and from 2.94 to 3.57 for the supervised condition.

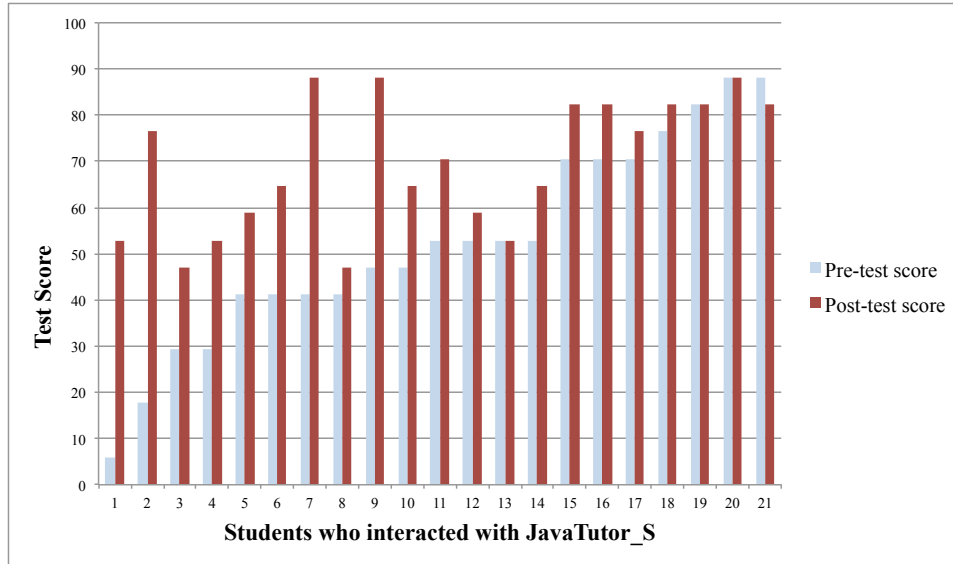


Figure 12.2: Bar charts visualizing students’ pre and post-test scores in the supervised condition.

### 12.3 Student Participation in JavaTutor-U and JavaTutor-S

Considering all three criteria of normalized learning gain, user satisfaction, and tutor feedback evaluation, we have observed that there were no significant differences between the two conditions. This section investigates the ways in which students in each condition participated throughout the tutoring session, in order to determine if the interaction patterns were different between them. To this end, we extracted features from system logs. The extracted features examined here include number of student utterances, changes in the students’ program code, student compile and run events, and tutor utterances received.

Students in the unsupervised condition made more dialogue utterances on average (JavaTutor-U mean=10.08; JavaTutor-S mean=8.24). 8.5% of all students (2 students in each condition) did not make any utterance and the average number of utterances for students who made an utterance was: JavaTutor-U=10.92 and JavaTutor-S=9.11. Students in JavaTutor-U also made a larger number of compile/run attempts (JavaTutor-U mean=95.50; JavaTutor-S mean=89.90) and more content changes in their program code (JavaTutor-U mean=639.19; JavaTutor-S mean=609.76). These higher levels of interaction from students in JavaTutor-U triggered a slightly higher average number of tutor moves (JavaTutor-U mean=43.77; JavaTutor-S mean=39.57). The only metric that the supervised condition (JavaTutor-S) had a higher average is on the number of compile errors students received throughout tutoring (JavaTutor-U mean=10.38; JavaTutor-S mean=11.52). None of these differences was statistically significant at the  $p < 0.05$  level.

To analyze the number of utterances students wrote in each condition in more detail, we

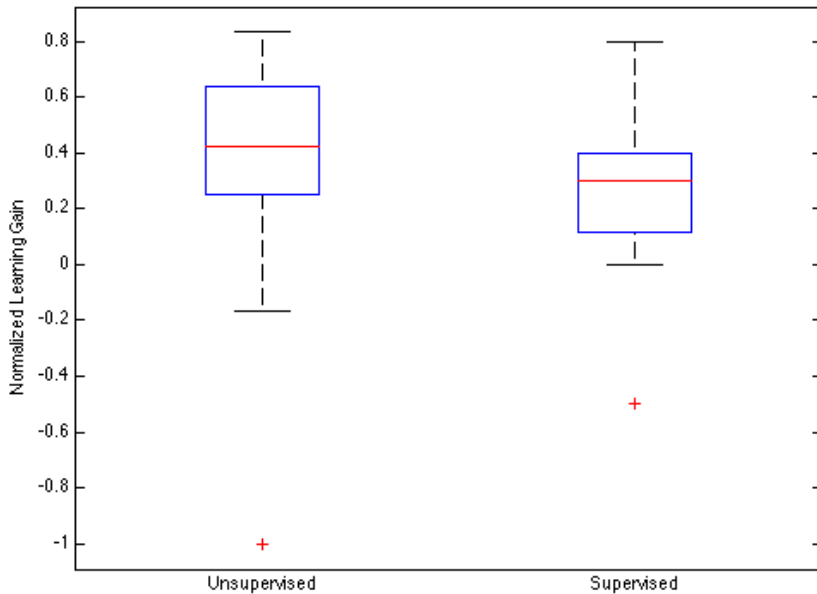


Figure 12.3: Box plots showing learning gains for both conditions

binned the students with five-point increments according to the number of utterances written. The students who wrote fewer than five utterances were grouped in the first bin, the students who wrote more than or equal to five utterances and less than ten utterances throughout the tutoring session were grouped in the second bin, and so on. There were six bins in total, representing up to 30 utterances (the number of student utterances ranged from 0 to 30). As shown in Figure 12.7, the students in JavaTutor-U wrote more utterances than JavaTutor-S when we consider  $\geq 10$  utterances (bins 3, 4, 5 and 6) (JavaTutor-U: 42.30%, JavaTutor-S: 28.57%). Also, while there was no student who wrote more than 25 utterances in the supervised condition, 7.7% of the students in the unsupervised condition chose to interact frequently with the system by writing more than 25 utterances during the tutoring session (bin 6).

Similar to the dialogue utterance analysis, we also analyzed the task progresses of students. Because the task was Java programming, compile and run events of the programming code constitute an important aspect of task progressions. Therefore, we binned the students by 50-event increments according to the total number of compile/run activities that they made throughout their interactions with the system. There were a total of 6 bins, representing up to 300 events. Similar to the dialogue aspect, the results show that the students in the JavaTutor-U condition compiled and ran their programming code more than the students in the JavaTutor-

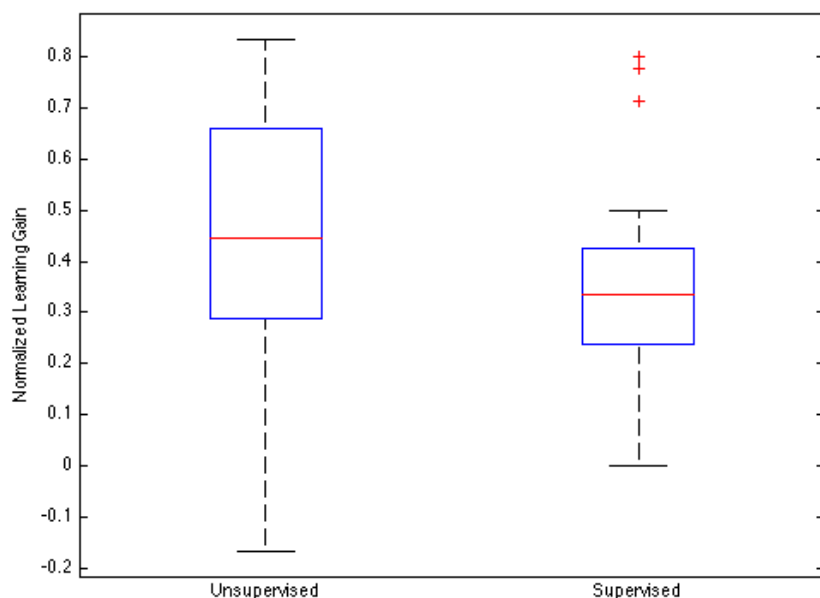


Figure 12.4: Box plots showing learning gains for both conditions considering the students who scored less than 75% of the questions correctly in the pre-test.

S condition. As shown in Figure 12.8, there were no students in JavaTutor-S condition who were in the 4<sup>th</sup> bin ( $150 \leq event < 200$ ) or 6<sup>th</sup> bin ( $250 \leq event < 300$ ) whereas students in JavaTutor-U condition participated that much in terms of their compile and run events.

## 12.4 Incoming Student Characteristics

Although the differences in learning and participation levels were not statistically significant between the two conditions, slight differences were present in numerous dimensions. Therefore, we investigated whether these differences are due to students' incoming characteristics, specifically their perceptions, attitudes, and their personality profile. If, for example, the students in JavaTutor-U were significantly more interested in computer science, or if they reported significantly higher extraversion scores on a personality profile, those characteristics could explain their increased participation.

When we compared pre-survey measures, the results showed that there was no significant difference between students in the conditions. Considering several metrics as was done in Chapter 11 (computer science confidence, computer science usefulness, computer science motivation,

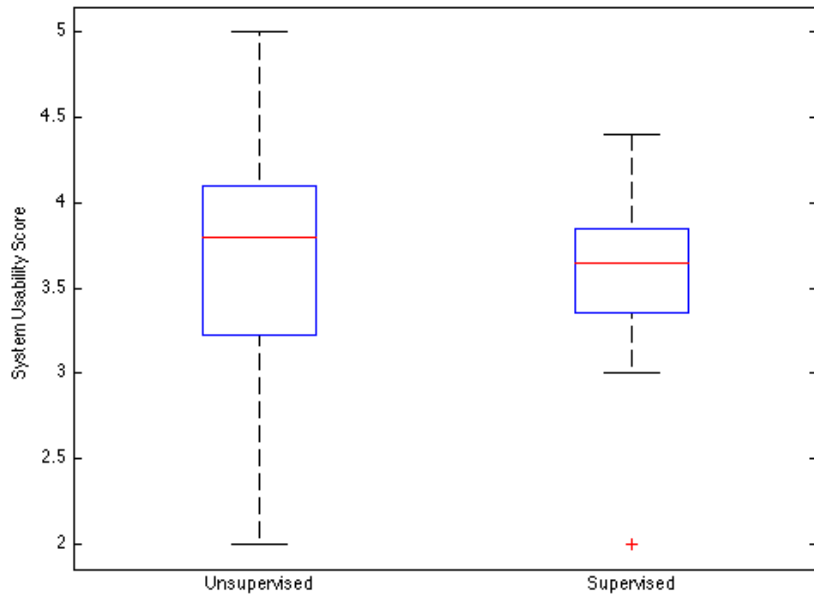


Figure 12.5: Box plots showing system usability scores for both conditions.

self-efficacy, learning goal orientation, performance demonstration, failure demonstration, failure avoidance, achievement goals and computer skill), students' attitudes who interacted with JavaTutor-U and JavaTutor-S were not significantly different. Because there were no differences found on incoming characteristics, we have slightly higher confidence in concluding that the promising results with JavaTutor-U are attributable to the unsupervised dialogue act model, which was the only systematic difference between the two treatment conditions.<sup>7</sup>

## 12.5 Discussion

Taking all evaluation criteria into account, the results showed that students in the JavaTutor-U condition interacted slightly more with the system and learned slightly more (Table 12.1). In terms of students' evaluation of the system, conditions were not significantly different. Although the results show that JavaTutor-U did not significantly outperform JavaTutor-S, there appears to be a trend suggesting that the unsupervised dialogue act models could support better learning, and higher levels of tutorial interaction, than the supervised models. As originally hypothesized,

<sup>7</sup>We cannot rule out that some incoming characteristics that were not measured could display systematic differences. However, the suite of metrics was informed by seven years of prior tutorial dialogue studies and were selected to be thorough, given the current set of available psychometric instruments.

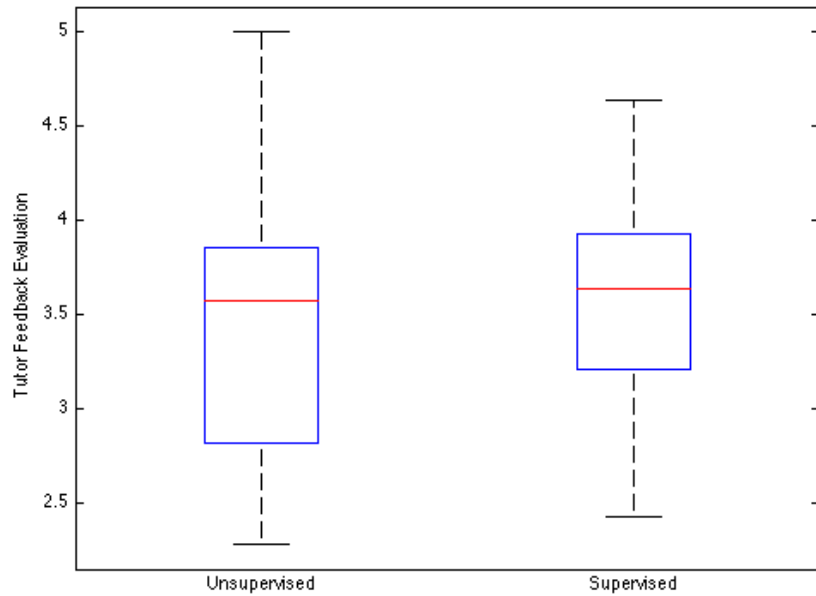


Figure 12.6: Box plots showing tutor feedback evaluation scores for both conditions.

this great promise could be due to several factors. First, the unsupervised models represent directly data-driven groupings of student utterances into the most natural clusters, rather than manually labeled tags that were created based upon dialogue taxonomies. Second, the unsupervised dialogue act classification may have supported better tutorial responses even given the same number of unsupervised clusters as there were manual tags. Finally, it is possible that this improvement in both regards resulted in students' increased participation either because they received higher quality help or because the interactions felt more natural to the students. As discussed in the next chapter, these findings point the way to several important directions for future work.

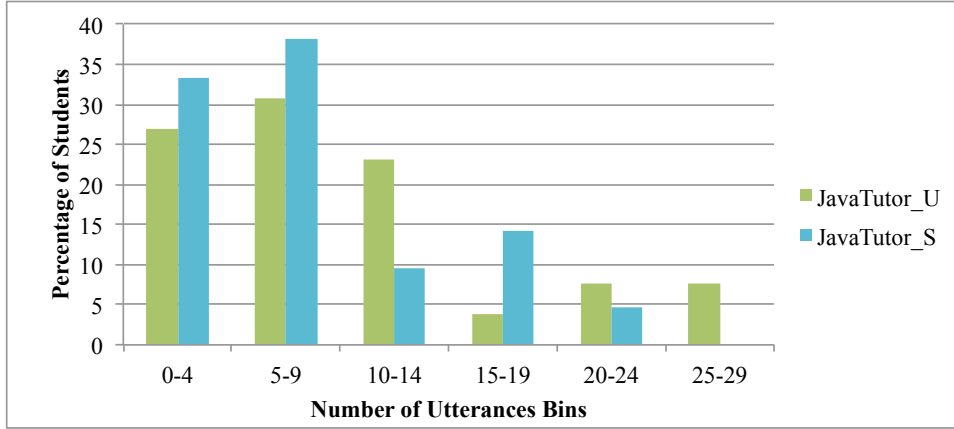


Figure 12.7: Percentage of utterances binned with 5 utterance increments.

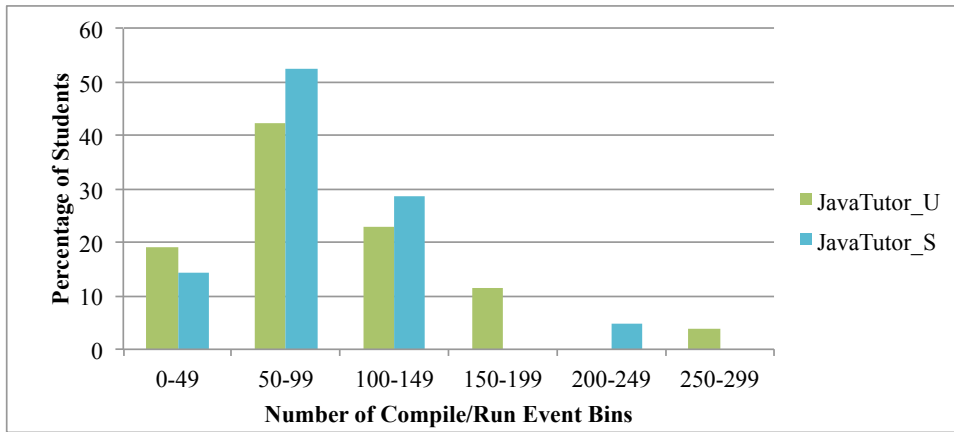


Figure 12.8: Percentage of compile/run events binned with 50 event increments.

Table 12.1: Results of the voluntary study for both versions of the system considering various metrics.

Metric	JavaTutor-S	JavaTutor-U
Number of utterances written	8.24	10.08
Number of compile/run events	89.90	95.50
Number of content changes	609.76	639.19
Number of compile errors	11.52	10.38
Number of tutor moves received	39.57	43.77
Pre-test score	52.38	43.44
Post-test score	69.75	68.55
Average normalized learning gain	0.29	0.37
System usability	3.59	3.59
Tutor feedback evaluation	3.57	3.47

# Chapter 13

## Conclusion

Building effective tutorial dialogue systems is a very challenging task. In order to overcome the bottleneck introduced by supervised dialogue act classifiers, this dissertation project created a task-oriented tutorial dialogue system based on data-driven dialogue act models and unsupervised classification. This system advanced the state of the art in tutorial dialogue systems by eliminating the manual annotation effort for engineering dialogue act taxonomies and providing the first tutorial dialogue system that implements unsupervised dialogue act classification in real time. Additionally, derivation and evaluation of tutorial policies using groupings of utterances showed the applicability and practicality of unsupervised dialogue act classifiers within tutorial dialogue systems. The system presented in this dissertation stands as a proof-of-concept for utilizing unsupervised dialogue act models that are built using human-human dialogues as a promising venue for real-time dialogue systems.

### 13.1 Hypotheses Revisited

This dissertation project revolved around the research question: “**How can we derive automatic dialogue act classification models that do not rely on manually labeled corpora, but that facilitate effective tutorial dialogue?**”. To answer this research question, several unsupervised dialogue act classifiers were built in this project and their evaluation was conducted with the following hypotheses:

**Hypothesis 1:** *A tutorial dialogue system that relies on unsupervised dialogue act models will support significant student learning.*

**Hypothesis 2:** *A tutorial dialogue system that relies on unsupervised dialogue act models can provide students with a satisfying user experience.*

User studies with the tutorial dialogue system were conducted to investigate these hypotheses. The results demonstrate that a tutorial dialogue system that relies on an unsupervised dialogue

act classifier can support significant learning. The students who interacted with JavaTutorU achieved significant improvements from their pre-test scores to post-test scores. In addition, the students in JavaTutorU group showed positive user satisfaction and evaluated the system’s feedback positively.

The sub-hypotheses, that JavaTutor-U would support significantly *better* learning and user satisfaction than JavaTutor-S were not supported by the evidence; however, there was a slight improvement with JavaTutor-U than with JavaTutor-S. We believe that this slight improvement is indeed due to the highly data-driven nature of the dialogue policies in JavaTutor-U, and it is possible that further exploration will uncover significant benefits of the unsupervised policy. Importantly, even if there is no significant improvement in learning or satisfaction with unsupervised dialogue act classification, the fact that we could maintain comparable learning and satisfaction while tremendously reducing manual annotation effort is extremely promising.

## 13.2 Summary

Improving natural language interactions within tutorial dialogue systems is a promising venue for increasing their effectiveness and making these systems more broadly available to support a wide variety of students and topic areas. Due to the engineering efforts required within several aspects of natural language modeling, it has traditionally been extremely time consuming and labor intensive to build tutorial dialogue systems. To greatly reduce the manual labor required for natural language understanding, this dissertation project presented a data-driven approach that utilized human-human corpora of tutorial dialogue to model interactions and used the learned models in real-time within a deployed tutorial dialogue system.

The machine-learned models learned groupings of student utterances that made sense qualitatively and quantitatively performed well when evaluated as a separate module outside of a system. To improve the performance of unsupervised dialogue act classifiers, we have proposed novel classification algorithms adapting information retrieval techniques of query-likelihood modeling and Markov random field graphs, as well as incorporated a rich set of features such as learner characteristics, dialogue history, task context and multimodal features. We showed that by the addition of these features and the improved classification algorithms, it is possible to achieve better performances with unsupervised dialogue act models. In addition, the groupings also suggested important distinctions that were not present in the manual tags. Motivated by the preliminary in-context evaluation we conducted, we deployed a tutorial dialogue system with an unsupervised dialogue act classifier in the backend.

This project has seen the creation of the end-to-end first tutorial dialogue system to rely on unsupervised dialogue act classification. The results show that the unsupervised model supports students significantly. In addition, students showed satisfaction with the system. The results are

promising not only for tutorial dialogue systems, but also for all dialogue systems as to the best of our knowledge, this dissertation work was the first to deploy a dialogue system in any domain that relies on an unsupervised dialogue act classifier in real time. The results showed that it is practical to utilize unsupervised models in real-time dialogue systems without requiring manual annotations or dialogue act taxonomy creation and labeling.

### **13.3 Threats to Validity**

This dissertation work proposed several novel unsupervised dialogue act classification techniques for the natural language processing community and presented the first dialogue system that relies on an unsupervised dialogue act classifier. While accomplishing these goals, there were notable limitations that constituted threats to validity.

First, the different conditions of the system were developed by the same person, the author of this dissertation. Although the best-performing supervised dialogue act classifier for the training corpus was utilized, as part of the dissertation the author had to implement the dialogue policy in the supervised condition as well. We made every effort to make both policies (supervised and unsupervised) to be as similar as possible, the groupings of utterances in clusters and the manual dialogue act tags did not exactly match (which was expected), therefore causing us to use slightly different tutor moves.

The participants of the user studies were also an important factor affecting the results of this dissertation. Learner characteristics, perceptions and attitudes toward the learning task are related to how much students participate in the study and how much they like the system. We took this into account by comparing incoming characteristics and attitudes of students in two conditions of the system. Also, we mitigated the effect of students' willingness on the evaluation results by conducting two studies, one in-class and one voluntary out-of-class setting.

### **13.4 Future Work**

This dissertation research has investigated machine-learned models based on introductory computer science dialogue and utilized these models for teaching introductory computer science in a tutorial dialogue system. To be able to generalize findings of this research, it would be a promising direction to apply these techniques to other domains and other tasks. Our research started investigating how the learned models transfer to other educational domains, particularly to Massively Open Online Courses (MOOC). Our preliminary experiments showed that the dialogue act classification approaches presented in this dissertation for tutorial dialogue is applicable to MOOC discussion forums with slight changes like addition of topic models on top

of the clusters (Ezen-Can, Boyer, Kellogg, & Booth, 2015). Further analysis for generalizing the learned models to other domains would be promising.

Dialogue management in dialogue systems consist of two crucial modules: natural language understanding and dialogue planning according to the output of understanding phase. This dissertation project focused on natural language understanding, particularly modeling dialogue acts of utterances. To this end, we have utilized a fixed set of tutor responses that are selected in real time based on the dialogue act classification result. Although the pre-defined set of tutor moves facilitated the evaluation process, it was somewhat limiting the effectiveness of the system. To be able to provide a wider variety of tutor moves in a tutorial dialogue system, it is promising to utilize natural language generation as the next step. Classifying dialogue acts in an unsupervised manor and generating tutor responses on-the-fly would significantly decrease the manual labor for the dialogue management module in dialogue systems.

## 13.5 Concluding Remarks

The novel contributions of this dissertation work include:

- A novel dialogue act classification technique, *query-likelihood clustering*, showing the applicability of information retrieval methodologies to the dialogue act classification task (Ezen-Can & Boyer, 2013b)
- Results showing the role of learner characteristics such as gender, self-efficacy and skill perception for adaptive dialogue act modeling (Ezen-Can & Boyer, 2014b, 2014c)
- Incorporation of task-context features derived from complex task-oriented tutorial dialogue into unsupervised dialogue act classification (Ezen-Can & Boyer, 2014a)
- Preliminary evaluation of unsupervised dialogue act models within their intended usage domain (Ezen-Can & Boyer, 2013a)
- The system built as part of this dissertation is the first to implement unsupervised dialogue act classification within tutorial dialogue systems (Ezen-Can & Boyer, 2015b).
- Results showing the role of learner characteristics and attitudes for tutorial dialogue system participation (Ezen-Can & Boyer, 2015a)
- The dialogue act models derived from human-human corpora can be used in other domains, eliminating the need to re-engineer dialogue act taxonomies for each corpora.
- The evaluation showed that machine learning models learned using human-human tutoring corpora transfer to human-computer tutoring.

## REFERENCES

- Aleven, V., Popescu, O., & Koedinger, K. R. (2001). Towards Tutorial Dialog to Support Self-Explanation: Adding Natural Language Understanding to a Cognitive Tutor. *Proceedings of Artificial Intelligence in Education*, 246–255.
- Allen, J. F., Schubert, L. K., Ferguson, G., Heeman, P., Hwang, C. H., Kato, T., . . . others (1995). The TRAINS project: A case study in building a conversational planning agent. *Journal of Experimental & Theoretical Artificial Intelligence*, 7(1), 7–48.
- Ames, C., & Archer, J. (1988). Achievement Goals in the Classroom: Students' Learning Strategies and Motivation Processes. *Journal of Educational Psychology*, 80(3), 260.
- Arroyo, I., & Woolf, B. P. (2005). Inferring learning and attitudes from a bayesian network of log file data. *Proceedings of AIED*, 33–40.
- Arthur, D., & Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms*, 1027–1035.
- Austin, J. L. (1975). *How to do things with words* (Vol. 1955). Oxford university press.
- Azarnoush, B., Bekki, J. M., & Bernstein, B. L. (2013). Toward a Framework for Learner Segmentation. *Journal of Educational Data Mining*, 5(2), 102–126.
- Baker, R. S., de Carvalho, A., Raspat, J., Aleven, V., Corbett, A. T., & Koedinger, K. R. (2009). Educational software features that encourage and discourage “gaming the system”. *Proceedings of AIED*, 475–482.
- Bandura, A. (2006). Guide for Constructing Self-Efficacy Scales. *Self-efficacy Beliefs Of Adolescents*(5), 307–337.
- Bangalore, S., Di Fabbrizio, G., & Stent, A. (2008). Learning the structure of task-driven humanhuman dialogs. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(7), 1249–1259.
- Beal, C. R., Qu, L., & Lee, H. (2008). Mathematics motivation and achievement as predictors of high school students' guessing and help-seeking with instructional software. *Journal of Computer Assisted Learning*, 24(6), 507–514.

- Beck, J. E. (2005). Engagement tracing: Using response times to model student disengagement. *Proceedings of AIED*, 88–95.
- Becker, L., Basu, S., & Vanderwende, L. (2012). Mind the gap: Learning to choose gaps for question generation. *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 742–751.
- Blikstein, P. (2011). Using learning analytics to assess students' behavior in open-ended programming tasks. *Proceedings of the International Conference on Learning Analytics Knowledge (LAK)*, 110–116.
- Blikstein, P. (2013). Multimodal learning analytics. *Proceedings of the International Conference on Learning Analytics Knowledge (LAK)*, 102–106.
- Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational researcher*, 4–16.
- Boyer, K. E., Grafsgaard, J., Ha, E. Y., Phillips, R., & Lester, J. C. (2011). An affect-enriched dialogue act classification model for task-oriented dialogue. *Proceedings of the International Conference of the Association for Computational Linguistics*, 1190–1199.
- Boyer, K. E., Ha, E. Y., Phillips, R., Wallis, M. D., Vouk, M. A., & Lester, J. C. (2010). Dialogue act modeling in a complex task-oriented domain. *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 297–305.
- Boyer, K. E., Phillips, R., Ingram, A., Ha, E. Y., Wallis, M., Vouk, M., & Lester, J. (2011). Investigating the relationship between dialogue structure and tutoring effectiveness: A hidden markov modeling approach. *International Journal of Artificial Intelligence in Education*, 21(1), 65–81.
- Boyer, K. E., Phillips, R., Wallis, M., Vouk, M., & Lester, J. (2008). Balancing Cognitive and Motivational Scaffolding in Tutorial Dialogue. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 239–249.
- Boyer, K. E., Vouk, M. A., & Lester, J. C. (2007). The influence of learner characteristics on task-oriented tutorial dialogue. *Proceedings of the 13th International Conference on Artificial Intelligence in Education (AIED)*, 365–372.

- Callaway, C., Dzikovska, M., Farrow, E., Marques-pita, M., Matheson, C., & Moore, J. (2007). The Beetle and BeeDiff tutoring systems. *In Proceedings of the SLaTE2007 Workshop*.
- Chen, G., Gully, S. M., & Eden, D. (2001). Validation of a new general self-efficacy scale. *Organizational Research Methods, 4*(1), 62–83.
- Chen, L., & Eugenio, B. D. (2013). Multimodality and dialogue act classification in the RoboHelper project. *Proceedings of the Annual Meeting of Special Interest Group on Discourse and Dialogue*, 183–192.
- Chen, S. S., & Gopalakrishnan, P. S. (1998). Clustering via the Bayesian information criterion with applications in speech recognition. *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, 2*, 645–648.
- Chi, M. T., Siler, S. A., Jeong, H., Yamauchi, T., & Hausmann, R. G. (2001). Learning from human tutoring. *Cognitive Science, 25*(4), 471–533.
- Cohen, J. (1968). Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin, 70*(4), 213.
- Cohen, P. A., Kulik, J. A., & Kulik, C.-L. C. (1982). Educational outcomes of tutoring: A meta-analysis of findings. *American Educational Research Journal, 19*(2), 237–248.
- Core, M. G., & Allen, J. (1997). Coding dialogs with the DAMSL annotation scheme. *AAAI Fall Symposium on Communicative Action in Humans and Machines*, 28–35.
- Crook, N., Granell, R., & Pulman, S. (2009). Unsupervised classification of dialogue acts using a dirichlet process mixture model. *Proceedings of the SIGDIAL 2009 Meeting on Discourse and Dialogue*, 341–348.
- Di Eugenio, B., Xie, Z., & Serafin, R. (2010). Dialogue act classification, higher order dialogue structure, and instance-based learning. *Dialogue & Discourse, 1*(2), 1–24.
- D’Mello, S., Williams, C., Hays, P., & Olney, A. (2009). Individual differences as predictors of learning and engagement. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 308–313.
- D’Mello, S. K., Lehman, B., & Graesser, A. (2011). A motivationally supportive affect-sensitive AutoTutor. *In New perspectives on affect and learning technologies* (pp. 113–126).

- Dzikovska, M., Steinhauser, N., Farrow, E., Moore, J., & Campbell, G. (2014). BEETLE II: Deep natural language understanding and automatic feedback generation for intelligent tutoring in basic electricity and electronics. *IJAIED*, *24*(3), 284–332.
- Evens, M. W., Chang, R.-C., Lee, Y. H., Shim, L. S., Woo, C. W., Zhang, Y., . . . Rovick, A. A. (1997). CIRCSIM-Tutor: An intelligent tutoring system using natural language dialogue. *ANLC '97 Proceedings of the Fifth Conference on Applied Natural Language Processing: Descriptions of System Demonstrations and Videos*, 13–14.
- Ezen-Can, A., & Boyer, K. E. (2013a). In-Context Evaluation of Unsupervised Dialogue Act Models for Tutorial Dialogue. *Proceedings of the 14th Annual SIGDIAL Meeting on Discourse and Dialogue*, 324–328.
- Ezen-Can, A., & Boyer, K. E. (2013b). Unsupervised classification of student dialogue acts with query-likelihood clustering. *International Conference on Educational Data Mining*, 20–27.
- Ezen-Can, A., & Boyer, K. E. (2014a). Combining task and dialogue streams in unsupervised dialogue act models. *Proceedings of the 15th Annual SIGDIAL Meeting on Discourse and Dialogue*, 113–122.
- Ezen-Can, A., & Boyer, K. E. (2014b). A Preliminary Investigation of Learner Characteristics for Unsupervised Dialogue Act Classification. *Proceedings of the 7th International Conference on Educational Data Mining (EDM)*, 373–374.
- Ezen-Can, A., & Boyer, K. E. (2014c). Toward adaptive unsupervised dialogue act classification in tutoring by gender and self-efficacy. *Workshop on Non-Cognitive Factors & Personalization for Adaptive Learning (NCFPAL) in Extended Proceedings of the 7th International Conference on Educational Data Mining (EDM)*, 94–100.
- Ezen-Can, A., & Boyer, K. E. (2015a). Choosing to interact: Exploring the relationship between learner personality, attitudes, and tutorial dialogue participation. *Proceedings of the International Conference on Educational Data Mining*, in press.
- Ezen-Can, A., & Boyer, K. E. (2015b). A Tutorial Dialogue System for Real-Time Evaluation of Unsupervised Dialogue Act Classifiers: Exploring System Outcomes. *Proceedings of the International Conference on Artificial Intelligence in Education*, in press.

- Ezen-Can, A., & Boyer, K. E. (2015c). Understanding Student Language: An Unsupervised Dialogue Act Classification Approach. *International Journal of Educational Data Mining (JEDM)*, 7(1), 51–78.
- Ezen-Can, A., Boyer, K. E., Kellogg, S., & Booth, S. (2015). Unsupervised Modeling for Understanding MOOC Discussion Forums: A Learning Analytics Approach. *Proceedings of the International Conference on Learning Analytics and Knowledge (LAK)*, 146-150.
- Ezen-Can, A., Grafsgaard, J. F., Lester, J. C., & Boyer, K. E. (2015). Classifying Student Dialogue Acts with Multimodal Learning Analytics. *Proceedings of the International Conference on Learning Analytics and Knowledge (LAK)*, To appear.
- Ferschke, O., Gurevych, I., & Chebotar, Y. (2012). Behind the article: Recognizing dialog acts in Wikipedia talk pages. *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 777–786.
- Forbes-Riley, K., & Litman, D. (2005). Using bigrams to identify relationships between student certainty states and tutor responses in a spoken dialogue corpus. *6th SIGDIAL Workshop on Discourse and Dialogue*.
- Forbes-Riley, K., & Litman, D. (2006a). Correlations between dialogue acts and learning in spoken tutoring dialogues. *Natural Language Engineering*, 12(2), 161–176.
- Forbes-Riley, K., & Litman, D. (2006b). Modelling user satisfaction and student learning in a spoken dialogue tutoring system with generic, tutoring, and user affect parameters. *Proceedings of the Human Language Technology Conference*, 264–271.
- Forbes-Riley, K., & Litman, D. (2009a). Adapting to student uncertainty improves tutoring dialogues. *AIED*, 33–40.
- Forbes-Riley, K., & Litman, D. (2009b). A User Modeling-Based Performance Analysis Of A Wizarded Uncertainty-Adaptive Dialogue System Corpus. *International Speech Communication Association*, 2467–2470.
- Forbes-Riley, K., & Litman, D. (2012). Adapting to multiple affective states in spoken dialogue. *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 217–226.

- Forbes-Riley, K., & Litman, D. (2013). When does disengagement correlate with performance in spoken dialog computer tutoring? *International Journal of Artificial Intelligence in Education*, 22(1), 39–58.
- Forbes-Riley, K., Litman, D., & Rotaru, M. (2008). Responding to student uncertainty during computer tutoring: An experimental evaluation. *Proceedings of the International Conference of Intelligent Tutoring Systems*, 60–69.
- Forbes-Riley, K., Rotaru, M., Litman, D., & Tetreault, J. (2007). Exploring affect-context dependencies for adaptive system development. *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, 41–44.
- Fossati, D., Di Eugenio, B., Brown, C., & Ohlsson, S. (2008). Learning linked lists: Experiments with the iList system. *Intelligent tutoring systems*, 80–89.
- Glass, M. S., & Evens, M. W. (2008). Extracting information from natural language input to an intelligent tutoring system. *Far Eastern Journal of Experimental and Theoretical Artificial Intelligence*, 1(2).
- Glass, M. S., & Evens, M. W. (2015). Extracting information from natural language input to an intelligent tutoring system. *Proceedings of International Conference on Artificial Intelligence in Education*.
- Graesser, A., Chipman, P., Haynes, B. C., & Olney, A. (2005). AutoTutor: An intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, 48(4), 612–618.
- Graesser, A., Person, N. K., & Magliano, J. P. (1995). Collaborative dialogue patterns in naturalistic one-to-one tutoring. *Applied Cognitive Psychology*, 9(6), 495–522.
- Grafsgaard, J. F., Fulton, R. M., Boyer, K. E., Wiebe, E. N., & Lester, J. C. (2012). *Multimodal Analysis of the Implicit Affective Channel in Computer-Mediated Textual Communication*, 145–152.
- Grafsgaard, J. F., Wiggins, J. B., Boyer, K. E., Wiebe, E. N., & Lester, J. C. (2013). Embodied affect in tutorial dialogue: Student gesture and posture. *Proceedings of the International*

- Conference on Artificial Intelligence in Education*, 1–10.
- Grosz, B. J., & Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3), 175–204.
- Ha, E. Y., Grafsgaard, J. F., Mitchell, C. M., Boyer, K. E., & Lester, J. C. (2012). Combining verbal and nonverbal features to overcome the 'information gap' in task-oriented dialogue. *Proceedings of the 13th SIGDIAL Meeting on Discourse and Dialogue*, 247–256.
- Hall, M., National, H., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter*, 11(1), 10–18.
- HersHKovitz, A., & Nachmias, R. (2011). Online Persistence In Higher Education Web-Supported Courses. *The Internet and Higher Education*, 14(2), 98–106.
- Higashinaka, R., Kawamae, N., Sadamitsu, K., Minami, Y., Meguro, T., Dohsaka, K., & Inagaki, H. (2011). Unsupervised clustering of utterances using non-parametric bayesian methods. *INTERSPEECH*, 2081–2084.
- Hirschberg, D. S. (1975). A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18(6), 341–343.
- Jackson, G. T., Graesser, A. C., & McNamara, D. S. (2009). What students expect may have more impact than what they know or feel. *Proceedings of AIED*, 73–80.
- Jain, A., Nandakumar, K., & Ross, A. (2005). Score Normalization In Multimodal Biometric Systems. *Pattern Recognition*, 38(12), 2270–2285.
- Jordan, P., Albacete, P., Ford, M. J., Katz, S., Lipschultz, M., Litman, D., . . . Wilson, C. (2013). Interactive event: The Rimac tutor—a simulation of the highly interactive nature of human tutorial dialogue. *Artificial Intelligence in Education*, 928–929.
- Joty, S., Carenini, G., & Lin, C.-Y. (2011). Unsupervised modeling of dialog acts in asynchronous conversations. *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 1807–1813.
- Jurafsky, D., Shriberg, E., Fox, B., & Curl, T. (1998). Lexical, prosodic, and syntactic cues for dialog acts. *Proceedings of ACL/COLING-98 Workshop on Discourse Relations and*

*Discourse Markers*, 114–120.

- Kanungo, T., Member, S., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., & Wu, A. Y. (2002). An Efficient k-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 881–892.
- Kim, S. N., Cavedon, L., & Baldwin, T. (2010). Classifying dialogue acts in one-on-one live chats. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 862–871.
- Klein, D., & Manning, C. D. (2003). Accurate Unlexicalized Parsing. *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, 423–430.
- Landauer, T. K., & Dumais, S. T. (1997). A solution to plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2), 211.
- Landis, J. R., & Koch, G. G. (1994). The Measurement of Observer Agreement for Categorical Data Data for Categorical of Observer Agreement The Measurement. *International Biometric Society*, 33(1), 159–174.
- Lane, H. C., & VanLehn, K. (2004). A dialogue-based tutoring system for beginning programming. *Proceedings of the FLAIRS Conference*, 449–454.
- Lane, H. C., & VanLehn, K. (2005). Teaching the tacit knowledge of programming to novices with natural language tutoring. *Computer Science Education*, 15(3), 183–201.
- Lee, C., & Bobko, P. (1994). Self-efficacy beliefs: Comparison of five measures. *Journal of Applied Psychology*, 79(3), 364.
- Lee, D., Jeong, M., Kim, K., Ryu, S., & Lee, G. (2013). Unsupervised spoken language understanding for a multi-domain dialog system. *IEEE Transactions On Audio, Speech, and Language Processing*, 21(11), 2451–2464.
- Litman, D., Moore, J. D., Dzikovska, M., & Farrow, E. (2010). Using natural language processing to analyze tutorial dialogue corpora across domains and modalities. *Proceedings of the 14th International Conference on AI in Education*.
- Litman, D., & Pan, S. (2002). Designing and evaluating an adaptive spoken dialogue system.

- User Modeling and User-Adapted Interaction*, 12(2-3), 111–137.
- Litman, D., & Silliman, S. (2004). ITSPROKE : An Intelligent Tutoring Spoken Dialogue System. *Demonstration Papers at HLT-NAACL 2004. Association for Computational Linguistics*, 5–8.
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction To Information Retrieval* (Vol. 1). Cambridge University Press.
- Marineau, J., Wiemer-Hastings, P., Harter, D., Olde, B., Chipman, P., Karnavat, A., . . . others (2000). Classification of speech acts in tutorial dialog. *Proceedings of the Workshop on Modeling Human Teaching Tactics and Strategies at the Intelligent Tutoring Systems 2000 Conference*, 65–71.
- Meece, J. L., & Holt, K. (1993). A Pattern Analysis Of Students' Achievement Goals. *Journal Of Educational Psychology*, 85(4), 582.
- Merceron, A., & Yacef, K. (2003). A Web-Based Tutoring Tool With Mining Facilities to Improve Learning and Teaching. *Proceedings of the 11th International Conference on Artificial Intelligence in Education*, 201–208.
- Merceron, A., & Yacef, K. (2005). Clustering Students To Help Evaluate Learning. *Technology Enhanced Learning*, 171, 31–42.
- Metzler, D., & Croft, W. B. (2005). A markov random field model for term dependencies. *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and development in information retrieval*, 472–479.
- Midgley, T. D., Harrison, S., & MacNish, C. (2009). Empirical verification of adjacency pairs using dialogue segmentation. *Proceedings of the 7th SIGDIAL Workshop on Discourse and Dialogue*, 104–108.
- Ng, R. T., & Han, J. (1994). Efficient and effective clustering methods for spatial data mining. *Proceedings of the 20th International Conference on Very Large Data Bases*, 144–155.
- Nye, B. D., Graesser, A. C., & Hu, X. (2014). AutoTutor and family: A review of 17 years of natural language tutoring. *IJAIED*, 24(4), 427–469.
- Paquette, L., Baker, R., Sao Pedro, M., Gobert, J., Rossi, L., Nakama, A., & Kauffman-Rogoff,

- Z. (2014). Sensor-free affect detection for a simulation-based science inquiry learning environment. *Proceedings of the International Conference on Intelligent Tutoring Systems*, 8474, 1–10.
- Pekrun, R., Goetz, T., Daniels, L. M., Stupnisky, R. H., & Perry, R. P. (2010). Boredom in achievement settings: Exploring control–value antecedents and performance outcomes of a neglected emotion. *Journal of Educational Psychology*, 102(3), 531.
- Pelleg, D., & Moore, A. (2000). X-means: Extending K-means with Efficient Estimation of the Number of Clusters. *Proceedings of the Seventeenth International Conference on Machine Learning*, 727–734.
- Quarteroni, S., Ivanov, A. V., & Riccardi, G. (2011). Simultaneous dialog act segmentation and classification from human-human spoken conversations. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 5596–5599.
- Rangarajan Sridhar, V. K., Bangalore, S., & Narayanan, S. (2009). Combining lexical, syntactic and prosodic cues for improved online dialog act tagging. *Computer Speech & Language*, 23(4), 407–422.
- Ricardo, B.-Y., & Ribeiro-Neto, B. (1999). *Modern Information Retrieval* (Vol. 463). New York: ACM press.
- Ritter, A., Cherry, C., & Dolan, B. (2010). Unsupervised modeling of twitter conversations. *Proceedings of the Association for Computational Linguistics*, 172–180.
- Rosé, C. P., Alevén, V., & Torrey, C. (2004). CycleTalk: Supporting reflection in design scenarios with negotiation dialogue. *CHI Workshop on the Designing for the Reflective Practitioner*.
- Rosé, C. P., Kumar, R., Alevén, V., Robinson, A., & Wu, C. (2006). Cycletalk: Data driven design of support for simulation based learning. *International Journal of Artificial Intelligence in Education*, 16(2), 195–223.
- Rus, V., D’Mello, S., Hu, X., & Graesser, A. C. (2013). Recent advances in conversational intelligent tutoring systems. *AI Magazine*, 34(3).
- Rus, V., Moldovan, C., Niraula, N., & Graesser, A. C. (2012). Automated discovery of speech act categories in educational games. *International Educational Data Mining Society*, 25–32.

- Sadohara, K., Kojima, H., Narita, T., Nihei, M., Kamata, M., Onaka, S., . . . Inoue, T. (2013, August). Sub-lexical dialogue act classification in a spoken dialogue system support for the elderly with cognitive disabilities. *Proceedings of the Fourth Workshop on Speech and Language Processing for Assistive Technologies*, 93–98.
- Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4), 288–297.
- Samei, B., Li, H., Keshtkar, F., Rus, V., & Graesser, A. C. (2014). Context-based speech act classification in intelligent tutoring systems. *Proceedings of the International Conference on Intelligent Tutoring Systems*, 236–241.
- Schegloff, E. A., & Sacks, H. (1973). Opening up closings. *Semiotica*, 8(4), 289–327.
- Searle, J. R. (1969). *Speech acts: An essay in the philosophy of language*. Cambridge University Press.
- Serafin, R., & Di Eugenio, B. (2004). FLSA: Extending latent semantic analysis with features for dialogue act classification. *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, 692–699.
- Shriberg, E., Stolcke, A., Jurafsky, D., Coccaro, N., Meteer, M., Bates, R., . . . Van Ess-Dykema, C. (1998). Can prosody aid the automatic classification of dialog acts in conversational speech? *Language and Speech*, 41(3-4), 443–492.
- Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., . . . Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3), 339–373.
- Strohman, T., Metzler, D., Turtle, H., & Croft, W. B. (2005). Indri: A language model-based search engine for complex queries. *Proceedings of the International Conference on Intelligent Analysis*, 2(6), 2–6.
- Tavafi, M., Mehdad, Y., Joty, S., Carenini, G., & Ng, R. (2013). Dialogue act recognition in synchronous and asynchronous conversations. *Proceedings of SIGDIAL Meeting on Discourse and Dialogue*, 117–121.
- Traum, D. R. (1999). Speech acts for dialogue agents. *Foundations of Rational Agency*, 169–201.

- Vail, A. K., & Boyer, K. E. (2014a). Adapting to Personality Over Time: Examining the Effectiveness of Dialogue Policy Progressions in Task-Oriented Interaction. *Proceedings of the 15th Annual SIGDIAL Meeting on Discourse and Dialogue*, 41–50.
- Vail, A. K., & Boyer, K. E. (2014b). Adapting to personality over time: Examining the effectiveness of dialogue policy progressions in task-oriented interaction. *Proceedings of the Annual SIGDIAL Meeting on Discourse and Dialogue*, 41–50.
- Vail, A. K., & Boyer, K. E. (2014c). Identifying effective moves in tutoring: On the refinement of dialogue act annotation schemes. *Proceedings of the International Conference on Intelligent Tutoring Systems (ITS)*.
- VandeWalle, D., Cron, W. L., & Slocum Jr, J. W. (2001). The role of goal orientation following performance feedback. *Journal of Applied Psychology*, 86(4), 629.
- VanLehn, K. (2008). The interaction plateau: Answer-based tutoring step-based tutoring=natural tutoring. *Intelligent Tutoring Systems*, 7–7.
- VanLehn, K., Graesser, A. C., Jackson, G. T., Jordan, P., Olney, A., & Rosé, C. P. (2007). When are tutorial dialogues more effective than reading? *Cognitive Science*, 31(1), 3–62.
- VanLehn, K., Jordan, P. W., Rosé, C. P., Bhembe, D., Böttner, M., Gaydos, A., ... others (2002). The architecture of why2-atlas: A coach for qualitative physics essay writing. *Intelligent Tutoring Systems*, 158–167.
- VanLehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R., Taylor, L., ... Wintersgill, M. (2005). The Andes physics tutoring system: Lessons learned. *IJAIED*, 15(3), 147–204.
- Walker, M. A., Litman, D., Kamm, C. A., & Abella, A. (1997). PARADISE: A framework for evaluating spoken dialogue agents. *Proceedings of the European Chapter of the Association for Computational Linguistics*, 271–280.
- Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems*. Morgan/Kaufmann Publishing Co.
- Zhai, C., & Lafferty, J. (2001). A study of smoothing methods for language models applied to ad-hoc information retrieval. *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 334–342.

## APPENDICES

# Appendix A

## Tutoring Tasks

**JAVATUTOR LESSON 1** During this research study you will write an interactive text-based adventure game. Writing this computer program, in the Java programming language, will teach you to apply some of the most important computer programming concepts. To get started, you will play a game that is similar to the one you will create this semester.

The JAVA CODE window displays a completed version of the game's code. Before the computer can "do" what the code instructs and let someone play the game, you must compile it. Compiling translates the human-readable code into a machine language.

Task 1 of 13: Compile the game code.

---

**AN ADVENTURE GAME** Now that the code is compiled, the computer is ready to run the program. Running the program makes the computer execute the code, doing what the program instructs on each line in sequence.

When you run this program, the computer will let you play the game. This will give you a sense of the game that you will write!

Task 2 of 13: Run the program.

---

**COMMENTS** Let's get started writing your game. As you write your game you will want to use comments, which are notes in English about your program. Java ignores these comments when you compile the code and run the program. Two forward-slashes ( // ) are a special character in the Java language used to write a single line comment.

Example: // This is a comment.

Task 3 of 13: Write a comment that includes your name.

---

**OUTPUT** During the game you played earlier, the program displayed output to you on the screen. Your program will need to display the scenes and various requests for information to

the player.

Example: `System.out.println("Hello");`

Any text within quotations is called a String literal. It is a long-standing tradition for a new programmer's first output to be the String literal, "Hello World!"

Task 4 of 13: Output: Hello World!

---

**TESTING** Testing your game involves first compiling your code and then running the program.

Task 5 of 13: Test your game.

---

**VARIABLES AND DATA TYPES** Your game needs to use values that the player gives you. The player will type in his or her name, and later, will type responses to prompts during the game. Your game also needs to use values that you provide inside the program, such as the name of your game.

Variables are named storage locations that contain these values (called data). Declaring a variable creates a named storage location in Java. You need to provide both a name and data type to declare a variable. A variable can hold values that match its data type. One data type is String, which allows a variable to hold a sequence of text characters.

Example: `String aStringVariable;`

Task 6 of 13: Declare a variable, using the String data type, to hold the name of your game.

---

**ASSIGNMENT** Storing a value inside a variable is called assignment.

Example: `aStringVariable = "A String literal";`

Task 7 of 13: Assign the name of your game to the variable you declared.

---

**OUTPUT** You can display the information stored inside a variable to the player. The player will see the currently assigned value of the variable.

Example: `System.out.println(aStringVariable);`

Task 8 of 13: Output the name of your game using the variable that stores your game's name. Then test your program.

---

**INPUT** Just as your game outputs information to the player it will also need to get information, called input, from the player. The first piece of information you will get is the player's name.

Task 9 of 13: Declare a String variable that will hold the player's name.

---

**INPUT** To get input from your player, you can use a Java Scanner. You will use `System.in` with your Scanner as the means for getting input from your player.

Task 10 of 13: Type the following code into your program: `Scanner playerInput; playerInput = new Scanner(System.in);`

---

**ASSIGNMENT THROUGH INPUT** You will use `nextLine()` with your Scanner variable named `playerInput` to get the player's name from the player. Once you get this value you will assign it to the player name variable you created earlier.

Example: `aStringVariable = playerInput.nextLine();`

Task 11 of 13: Assign the player's name to the player name variable you declared, using `playerInput.nextLine()`. Then test your program.

---

**PROMPTS** You may have noticed that your program was waiting for input from the player but didn't tell the player what to type. It is helpful to output a request, called a prompt, for the information.

Example: `System.out.println("Please give me some information:");`

Task 12 of 13: Output a prompt for the player's name. Test your program.

---

**OUTPUT OF TEXT AND VARIABLES TOGETHER** There are many other cases where your game will need to output a String literal along with the value assigned to a variable.

Example: `System.out.println("Information in example variable: " + aStringVariable);`

Task 13 of 13: Output a message with the player's name. For example, if the player inputs the name Mary, the output message would be: Player's name is Mary. Then test your program.

---

**CONGRATULATIONS!** You've created the first part of a text-based adventure game.

Today you learned: - What a variable is - How to create (or declare) a variable - How to store (or assign) information to a variable - How to send output to your player - How to get input from your player - How to output the value of a variable

---

## Appendix B

# Main Pre-survey

## JavaTutor Main Presurvey

Please indicate the extent to which you agree or disagree with each statement.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Generally I have felt secure about attempting computer programming problems.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am sure I could do advanced work in computer science.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am sure that I can learn programming.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I think I could handle more difficult programming problems.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I can get good grades in computer science.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I have a lot of self-confidence when it comes to programming.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I'll need programming for my future work.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I study programming because I know how useful it is.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Knowing programming will help me earn a living.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Computer science is a worthwhile and necessary subject.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I'll need a firm mastery of programming for my future work.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I will use programming in many ways throughout my life.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I like writing computer programs.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Programming is enjoyable and stimulating to me.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
When a programming problem arises that I can't immediately solve, I stick with it until I have the solution.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Once I start trying to work on a program, I find it hard to stop.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
When a question is left unanswered in computer science class, I continue to think about it afterward.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am challenged by programming problems I can't understand immediately.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Prev

Next

Figure B.1: Computer Science Attitudes Instrument

## JavaTutor Main Presurvey

Please indicate the extent to which you agree or disagree with each statement.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I will be able to achieve most of the goals that I have set for myself.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
When facing difficult tasks, I am certain that I will accomplish them.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
In general, I think that I can obtain outcomes that are important to me.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I believe that I can succeed at most any endeavor to which I set my mind.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I will be able to successfully overcome many challenges.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am confident that I can perform effectively on many different tasks.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Compared to other people, I can do most tasks very well.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Even when things are tough, I can perform quite well.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am willing to select a challenging work assignment that I can learn a lot from.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I often look for opportunities to develop new skills and knowledge.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I enjoy challenging and difficult tasks at work where I'll learn new skills.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
For me, further development of my work ability is important enough to take risks.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I like to show that I can perform better than my coworkers.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I try to figure out what it takes to prove my ability to others at work.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I enjoy it when others at work are aware of how well I am doing.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I prefer to work on projects where I can prove my ability to others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I would avoid taking on a new task if there was a chance that I would appear rather incompetent to others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Avoiding a show of low ability is more important to me than learning a new skill.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I'm concerned about taking on a task at work if my performance would reveal that I had low ability.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I prefer to avoid situations at work where I might perform poorly.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Prev

Next

Figure B.2: New General Self-Efficacy Instrument (8 items) followed by Goal Orientation Instrument (12 items)

## JavaTutor Main Presurvey

### I See Myself as Someone Who...

	Disagree strongly	Disagree a little	Neither agree or disagree	Agree a little	Agree strongly
... is talkative.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... tends to find fault with others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... does a thorough job.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is depressed, blue.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is original, comes up with new ideas.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is reserved.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is helpful and unselfish with others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... can be somewhat careless.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is relaxed, handles stress well.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is curious about many different things.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is full of energy.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... starts quarrels with others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is a reliable worker.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... can be tense.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is ingenious, a deep thinker.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... generates a lot of enthusiasm.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... has a forgiving nature.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... tends to be disorganized.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... worries a lot.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... has an active imagination.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... tends to be quiet.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is generally trusting.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... tends to be lazy.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is emotionally stable, not easily upset.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is inventive.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... has an assertive personality.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... can be cold and aloof.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... perseveres until the task is finished.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... can be moody.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... values artistic, aesthetic experiences.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is sometimes shy, inhibited.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is considerate and kind to almost everyone.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... does things efficiently.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... remains calm in tense situations.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... prefers work that is routine.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

... prefers work that is routine.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is outgoing, sociable.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is sometimes rude to others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... makes plans and follows through with them.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... gets nervous easily.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... likes to reflect, play with ideas.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... has few artistic interests.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... likes to cooperate with others.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is easily distracted.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
... is sophisticated in art, music, or literature.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Prev Next

[Progress Bar]

Figure B.3: Big Five Inventory

## JavaTutor Main Presurvey

Please indicate your opinion about each of the statements below in reference to your current situation.

	Not at all true		Somewhat true			Very true of me	
It is important for me to do better than other students.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It is important for me to do well compared to others completing this exercise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My goal in completing this exercise is to get a better score than most of the other students.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I worry that I may not learn all that I possibly could while completing this exercise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Sometimes I'm afraid that I may not understand the content of things as thoroughly as I'd like.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am often concerned that I may not learn all that there is to learn.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I want to learn as much as possible from this exercise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
It is important for me to understand the content of this exercise as thoroughly as possible.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I desire to completely master this exercise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I just want to avoid doing poorly while completing this exercise.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My goal during this exercise is to avoid performing poorly.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My fear of performing poorly is often what motivates me.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Prev

Next

Figure B.4: Achievement Goals Questionnaire



## JavaTutor Main Presurvey

What is your gender?

- Male
- Female
- Prefer not to say

How old are you?

What is your undergraduate class standing?

- Freshman
- Sophomore
- Junior
- Senior
- Other (please specify)

How skilled are you with computers, compared to the average person?

- Much more skilled
- Somewhat more skilled
- Average
- Somewhat less skilled
- Much less skilled

Please indicate your race (ethnicity):

- Native American
- White
- Latino/a
- African American
- Middle Eastern
- Asian
- Pacific Islander
- Other (please specify)

What is your major?

Survey Powered By [Qualtrics](#)

Figure B.5: Demographics Questionnaire

## Appendix C

### Lesson 1 Student Pre-test

## JavaTutor Lesson 1 Pretest

- 1) "Jan 8"
- 2) var s;
- 3) //Jan 8
- 4) s = "Jan 8";
- 5) System.out.println(s);
- 6) System.out.println("s");
- 7) String s;
- 8) System.out.println("Jan 8");
- 9) //s is a String
- 10) system.out.println("Jan 8");

Next to each explanation below, type the number of the matching Java code from the above lines.

Outputs the String literal "Jan 8" to the user.

Is a comment with the date Jan 8.

Is a String literal.

Declares a String variable named s.

Assigns a value to String variable s.

Outputs the value stored in String variable s.

- A) makes space in the computer to store a value.
- B) translates human-readable code into machine language.
- C) gets information from the user to the program
- D) stores a value in a variable.
- E) displays information from the program to the user.
- F) pauses the computer to wait for user input.
- G) makes computer programs more readable to humans.
- H) allows the programmer to see if the program behaves as expected.

Next to each phrase below, type the letter of the matching phrase from the above lines that completes the sentence.

Compiling a program	<input type="text"/>
A comment	<input type="text"/>
An output statement	<input type="text"/>
Testing a program	<input type="text"/>
Declaring a variable	<input type="text"/>
An assignment statement	<input type="text"/>
An input statement	<input type="text"/>

Fill in the missing four code statements in a correct order.

- 1) `Scanner playerInput;`
- 2) `playerInput = new Scanner(System.in);`
- 3) \_\_\_\_\_
- 4) \_\_\_\_\_
- 5) \_\_\_\_\_
- 6) \_\_\_\_\_

	Line 3:	Line 4:	Line 5:	Line 6:
<code>age = playerInput.nextInt();</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<code>int age;</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<code>System.out.println("Your age is: " + age);</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<code>System.out.println("Enter your age:");</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Next

Survey Powered By [Qualtrics](#)

Figure C.1: JavaTutor Lesson 1 pre-test.

## Appendix D

# Lesson 1 Student Post-test and Post-survey

## JavaTutor Lesson 1 Postsurvey and Posttest

Please indicate the extent to which you agree with each statement.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I forgot about my immediate surroundings while working on the learning task.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I was so involved in my learning task that I ignored everything around me.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I lost myself in this learning experience.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I was so involved in my learning task that I lost track of time.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I blocked out things around me when I was working.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
When I was working, I lost track of the world around me.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The time I spent working just slipped away.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I was absorbed in my task.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
During this learning experience I let myself go.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt frustrated while using the tutoring system.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the tutoring system confusing to use.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt annoyed while using the tutoring system.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt discouraged while using the tutoring system.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Using the tutoring system was mentally taxing.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
This learning experience was demanding.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt in control of my learning experience.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I could not do some of the things I needed to do on the tutoring system website.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The tutoring system was attractive.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The tutoring system was aesthetically appealing.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I liked the graphics and images used in the tutoring system.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure D.1: User Engagement Scale

The tutoring system appealed to my visual senses.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The screen layout of the tutoring system was visually pleasing.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Working on this learning task was worthwhile.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I consider my learning experience a success.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My learning experience did not work out the way I had planned.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My learning experience was rewarding.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I would recommend using this tutoring system to my friends and family.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I would continue to learn with this tutoring system out of curiosity.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The content of the tutoring system incited my curiosity.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt interested in my learning task.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I was really drawn into my learning task.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt involved in this learning task.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
This learning experience was fun.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure D.2: User Engagement Scale continued

Please choose a value from 1 to 100, where 1 indicates less and 100 indicates more.

How mentally demanding was the task?

How physically demanding was the task?

How hurried or rushed was the pace of the task?

How successful were you in accomplishing what you were asked to do?

How hard did you have to work to accomplish your level of performance?

How insecure, discouraged, irritated, stressed, and annoyed were you?

Survey Powered By [Qualtrics](#)

Figure D.3: NASA-TLX workload index

## Appendix E

# User Satisfaction Survey

## JavaTutor Lesson 1 Postsurvey and Posttest

Please indicate the extent to which you agree with each statement.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
My tutor was knowledgeable about programming.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor was supportive.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor's input was helpful.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt encouraged by my tutor.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor responded to me promptly.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor's advice intruded on my concentration.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor helped me finish the programming exercises more quickly than I would have on my own.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor helped me understand the most important concepts better than I would have on my own.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My tutor encouraged me to be part of the conversation.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I was more motivated doing this exercise with the tutor than I would have been on my own.	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Prev

Next

Survey Powered By [Qualtrics](#)

Figure E.1: Usability questions for the proposed tutorial dialogue system.