User Simulation for Spoken Dialog System Development

by

Hua Ai

B.S., Shanghai Jiao Tong University, 2004

M.S., University of Pittsburgh, 2006

Submitted to the Graduate Faculty of

Intelligent Systems Program in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2009

UNIVERSITY OF PITTSBURGH

SCHOOL OF ARTS AND SCIENCES

This dissertation was presented

by

Hua Ai

It was defended on

September 21, 2009

and approved by

Dr. Diane Litman, Professor, Department of Computer Science

Dr. Rebecca Hwa, Associate Professor, Department of Computer Science

Dr. Pamela Jordan, Ph.D., Department of Biomedical Informatics

Dr. Janyce Wiebe, Professor, Department of Computer Science

Dr. Maxine Eskenazi, Associate Teaching Professor, LTI, Carnegie Mellon University

Dissertation Advisor: Dr. Diane Litman, Professor, Department of Computer Science

User Simulation for Spoken Dialog System Development

Hua Ai, PhD

University of Pittsburgh, 2009

A user simulation is a computer program which simulates human user behaviors. Recently, user simulations have been widely used in two spoken dialog system development tasks. One is to generate large simulated corpora for applying machine learning to learn new dialog strategies, and the other is to replace human users to test dialog system performance. Although previous studies have shown successful examples of applying user simulations in both tasks, it is not clear what type of user simulation is most appropriate for a specific task because few studies compare different user simulations in the same experimental setting.

In this research, we investigate how to construct user simulations in a specific task for spoken dialog system development. Since most current user simulations generate user actions based on probabilistic models, we identify two main factors in constructing such user simulations: the choice of user simulation model and the approach to set up user action probabilities. We build different user simulation models which differ in their efforts in simulating realistic user behaviors and exploring more user actions. We also investigate different manual and trained approaches to set up user action probabilities. We introduce both task-dependent and task-independent measures to compare these simulations. We show that a simulated user which mimics realistic user behaviors is not always necessary for the dialog strategy learning task. For the dialog system testing task, a user simulation which simulates user behaviors in a statistical way can generate both objective and subjective measures of dialog system performance similar to human users.

Our research examines the strengths and weaknesses of user simulations in spoken dialog system development. Although our results are constrained to our task domain and the resources available, we provide a general framework for comparing user simulations in a task-dependent context. In addition, we summarize and validate a set of evaluation measures that can be used in comparing different simulated users as well as simulated versus human users.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

I have enjoyed a lot during my five years at Pitt. I was very lucky to be in this productive and friendly environment. I am indebted to many people for the successful completion of this document. Most importantly, I would like to thank my advisor Diane Litman, who has been an unending source of advice for me. I am very grateful for her generous support throughout these years.

I would also want to thank the other members of my committee, Maxine Eskenazi, Rebecca Hwa, Pamela Jordan, and Janyce Wiebe for their invaluable suggestions, critical eye, and enduring support throughout my Ph.D. study. Special thanks go to David Traum and Fuliang Weng, who offered me the wonderful opportunities to work in different research groups.

I have learned a lot from an amazing group of fellow students and collaborators. Here is an incomplete list: Antonie Raux, Arthur Ward, Beatriz Maeireizo Tokeshi, Behrang Mohit, Cem Akkaya, Colin Lynch, Greg Nicholas, Joanna Drummond, Joel Tetreault, Kate Forbes-Riley, Min Chi, Mihai Rotaru, Michael Lipschultz, Scott Silliman, Swapna Somasundaran, and Wenting Xiong. I would like to thank all of them for their support and friendship.

Finally, a big Thank-You goes to my beloved family. I thank my parents for life and the strength and determination to live it. Most of all, I thank my beloved husband Hui, who shares my burdens and joys everyday.

1.0 INTRODUCTION

1.1 MOTIVATIONS

Recent advances in spoken language understanding have made it possible to develop spoken dialog systems for many applications in information services, entertainment, education, healthcare, and so on. People are getting used to calling these dialog systems to book flight tickets, to enter their gas meter readings, or to query for weather conditions. Such systems have the potential benefits of remote or hands-free access which makes it easier and more natural for the users to interact with the system, especially for those who need to perform cognitively demanding primary tasks while manipulating the system. Large-scale deployment of spoken dialog systems in call centers also helps companies to cut huge labor costs.

Among the components in spoken dialog systems, the dialog manager plays a principal role in managing the state of the dialog in a natural way so that the system can interact with users to complete the tasks that the system is designed to support. More specifically, the dialog manager decides when to talk to the user and what to talk about based on a dialog strategy. The performance of the dialog manager and its dialog strategy has a direct impact on task completion rates (Janarthanam & Lemon, 2008).

Typically, dialog strategies are manually designed by domain experts. Although expert experiences are very valuable in system design, these manually summarized rules often only cover some of the situations that a dialog system needs to deal with when interacting with users. What is more, the policy writing process is very intensive and takes a long time to accomplish since the experts have to make many nontrivial design choices.

A recent trend in spoken dialog system design is to use Machine Learning techniques to learn dialog strategies automatically. The main motivation is to learn these strategies from data rather than having to rely on handcrafted rules. However, it is very rare that enough data is available for the automatic learner to sufficiently explore the vast space of possible dialog states and strategies. One solution is to use user simulation to generate a large dialog corpus. While human subject experiments are usually expensive and time-consuming, simulated users can generate large amounts of training data in a low-cost and time-efficient way. The large simulated corpus also makes it possible to explore dialog strategies which are not present in the training corpus, so that new and potentially better strategies can be found. Many previous studies have demonstrated that simulation models of real user behaviors can be successfully trained from small corpora (e.g., (Schatzmann et al., 2005a), (Georgila et al., 2006)). (Rieser et al., 2006) show that a simple user simulation model can generate user behaviors that do not differ significantly from human user behaviors. In addition, research studies ((Schatzmann et al., 2007a), (Lemon et al., 2006)) show that dialog strategies learned from the simulated dialog corpora can even outperform the hand-written policies.

Although these results show promising applications of user simulations, several questions need to be answered to better use user simulations in spoken dialog system development. First, since user simulations are built to replace human users, one intuition is that a good user simulation should be able to mimic realistic human user behaviors. Therefore, how to measure the human-likeliness of simulated behaviors is an important problem to solve. Second, since user simulations in our research are built to facilitate dialog system development, it is also important to evaluate user simulations in a task-dependent context. In particular, we are interested in whether a simulation model which mimics realistic user behaviors is always needed. As we are going to show that building a realistic user simulation is not a trivial task, we are interested in building other user simulation models which can perform as good as a realistic user simulation but are easier to construct.

In the next section, we describe how we formulate these two questions into a set of research studies and present a summary of our main results.

1.2 GENERAL APPROACH AND RESULTS

We explore how to construct user simulations for spoken dialog system development by comparing the quality of different user simulations. The quality of a user simulation is usually measured by the quality of the dialog behaviors generated by the user simulation. Many studies (e.g., (Schatzmann et al., 2005a), (Georgila et al., 2006)) use evaluation measures that can be automatically extracted from the simulation logs to evaluate the qualities of simulation models. These evaluation measures are designed to compare a simulated user corpus with a human user corpus at different levels. These measures assume that the more similar the user simulation behaviors are to human user behaviors, the more human-like the user simulation is. Previous studies show that the evaluation results using such automatic measures match the researchers' expectations and therefore consider these measures to be valid. However, researchers' own expectations can still be biased since the researchers design the studies themselves with pre-existing beliefs. A more convincing approach would be to collect a large number of opinions regarding the performance of different user simulations from human judges who are not involved in the design of the study. Other communities (e.g., summarization, machine translation) use human judgments collected in this way to validate the automatic evaluation measures. We believe that it is also necessary to validate the previously used user simulation evaluation measures by human judgments. Therefore, in our study, we first collect human judgments to validate previously used evaluation measures. We show that prediction models of human judgments can be built using those evaluation measures. Furthermore, we look into the differentiating power of those evaluation measures, i.e., to what extent the measures can tell the differences between two simulated corpora. We find that the previously used evaluation measures are not always able to capture the differences between simulated and realistic user behaviors in our task domain. Thus, we explore a domain-dependent constraint that can be used as a more powerful evaluation measure.

While the above approach measures how human-like user simulations are, in this research we are more interested in comparing user simulations in a task-dependent context since the user simulations here are built to facilitate dialog system development. The user simulations we investigate in this research are built on probabilistic models. Generally speaking, these user simulations choose a user action given the current dialog context based on a pre-existing user action probability distribution. When comparing user simulations, we consider two factors: the choice of user simulation models and the approach to set up user action probabilities in the simulation models. The choice of user simulation model defines the mechanism of deciding the next user actions given the current dialog context. Taking into account more context features in the simulation models helps to present a more complete context, but can add more complexities in the simulation models as well as increase computational costs (Williams and Young, 2007b).

User action probabilities can be either be assigned manually or be estimated from observed human user data. The choice depends on the size of the available training data and the complexity of the simulation model (Janarthanam and Lemon, 2008).

We compare different user simulations on two dialog system development tasks: one is a dialog strategy learning task in which user simulations generate large simulated corpora for applying machine learning techniques to learn new dialog strategies; the other is a dialog system testing task in which user simulations are used to test dialog system performance instead of human users. In the first task, user simulations are used to generate a training corpus for applying Reinforcement Learning to design new dialog strategies. We observe that for this task a simulation model which randomly generates user behaviors with certain constraints outperforms models which generate more human-like behaviors statistically. This result suggests that we do not always need to build a realistic user simulation for the dialog strategy learning task. In the second task, user simulations are used to test the performance of the dialog system during interactions with the system. It is intuitive that user simulations that can mimic realistic user behaviors are needed for this task since here we depend on the user simulations to generate the reactions that human users will have when interacting with the dialog system. While previous studies (e.g., (Georgila et al., 2006), (Schatzmann et al., 2007a)) already show that the state-of-the-art user simulations are capable of testing dialog systems by providing objective measures that can be extracted from system-user interaction logs, no study to our knowledge has used user simulations to predict subjective user satisfaction scores. We believe that the subjective measures are as important to dialog system development as the objective measures. Therefore, in our research, we show that a state-of-the-art user simulation model can produce testing dialogs with user satisfaction scores comparable to human users.

To sum up, we investigate how to construct user simulations for different dialog system development tasks by comparing the performance of different user simulations. We first examine how to compare user simulation behaviors by using human judges as well as by applying automatic evaluation measures on simulated dialogs. Then, we compare the performance of different user simulations in a task-dependent context using a dialog strategy learning task and a dialog system testing task. We examine how to choose user simulation models and how to set up user action probabilities in probabilistic user simulations in the above tasks. We provide a methodology to evaluate the effectiveness of user simulations in assisting dialog system development.

1.3 CONTRIBUTIONS OF THIS WORK

This work contributes to user simulation evaluation. Previous research studies use evaluation measures that can be automatically computed from simulation logs to show that the simulated user behaviors do not differ significantly from human user behaviors. However, the validity of these measures is not fully examined. One contribution of our study is to use human judgments as the gold standard to validate the previously used evaluation measures. We also predict human judgments using the automatic measures. We find that it is hard to predict human judges' ratings on the dialogs while dialog rankings can be predicted correctly. We further use task-dependent features to improve the prediction model.

Our work also contributes to spoken dialog system development. Previous research shows many promising applications of user simulations in spoken dialog system development. However, most of those studies only conclude that a certain type of user simulation is helpful in their own task settings. Less information is given on comparisons among different simulations. Our study explores which type of user simulation is suitable for specific dialog system development tasks. We identify two main factors in constructing probabilistic user simulations and compare different user simulations which differ in the two factors in specific tasks. Our comparison results can be used as a reference in constructing user simulations. In addition, we summarize a set of evaluation measures to compare user simulation performance in a task-dependent context. Our experimental design and evaluation measures provide an empirical framework for constructing user simulations given specific dialog system development tasks.

1.4 A GUIDE FOR READERS

The rest of the dissertation is organized as follows.

In Section 2 – Previous Work, we review previous work on using user simulations for spoken dialog system development. Section 2.1 reviews previously constructed user simulation models. Section 2.2 reviews previously used simulation evaluation measures. Section 2.3 and Section 2.4 describe how user simulations are used in dialog strategy learning and dialog system testing tasks respectively.

In Section 3 – Testbed Systems, Corpora and User Simulations, we present different systems we used as the testbeds of this research and explain why we use different systems. We describe the corpora we collected with each of the system and annotations on those corpora. In addition, we describe the user simulations we built on the dialog systems we introduced.

In Section 4 – User Simulation Evaluation Measures, we introduce different types of evaluation measures for comparing user simulation performance. Section 4.1 presents the previously used domain-independent measures. We validate these measures by human judgments in Section 4.1.1. Then, we build prediction models of human judgments using these automatic measures (4.1.2). However, we also point out the deficiency of these domain-independent measures in 4.1.3. Section 4.2 introduces a domain-dependent constraint to evaluate simulations built for the tutoring domain (4.2.1). We also use this constraint to improve user simulation models. In Section 4.2.2, we show that this domain-dependent constraint better captures the differences between simulated and human user behaviors than the domain-independent features. We further use this domain-dependent constraint to improve the prediction model of human judgments in Section 4.2.3.

In Section 5 – User Simulation for Two Dialog System Development Tasks, we examine the performance of different user simulations in two tasks: dialog strategy learning and dialog system testing. In Section 5.1 we look into the first task. In 5.1.1, we define the dialog strategy learning task we use in different experiments in Section 5.1. Then, we show that constructing user simulations is needed to generate large training corpora for learning new dialog strategies using reinforcement learning (5.1.2). This justifies our efforts in investigating user simulations in the dialog strategy learning task in the rest of this section. Then, we look into constructing user simulations for the dialog strategy learning task from two aspects, i.e., the choice of user simulation models (5.1.3) and the approach to set up user action probabilities in the simulation models (5.1.4). We look into the second task of testing dialog systems in Section 5.2. In Section 5.2.1, we discuss what type of user simulation models should be chosen for constructing user simulations to test dialog systems. We also show that the user simulation model we choose can provide objective measures of dialog system performance as human users can. Since it is important to get subjective measures on dialog system performance besides the objective measures, in Section 5.2.2 we investigate how to predict user satisfaction scores from simulation logs. After that, we discuss the other important factor in building user simulations for dialog system testing, i.e., setting up user action probabilities in 5.2.3.

In Section 6 – Other Types of User Simulations, we discuss some other approaches to construct user simulations that are not used in our studies. We explained our interests in constructing probabilistic user simulations while pointing out some non-probabilistic user simulations in Section 6.1. We suggest that different types of user simulations should be chosen for different purposes. Even for probabilistic user simulations, there are factors other than the two main factors, i.e., probabilistic model and user action probabilities, which are less considered in research studies. In Section 6.2, we look at another factor that will impact the behaviors of a probabilistic user simulation, i.e., the data that the user simulation is trained on. Since most user simulations are trained from recruited subject corpora while aiming to represent real user behaviors, we conduct a study to investigate the differences among spoken dialog corpora collected with recruited subjects versus real users. We infer the impact of training data on user simulation behaviors based on the results from the corpus comparison study.

We conclude in Section 7 with a brief summary of our main findings and contributions and suggest several interesting directions for future work.

2.0 PREVIOUS WORK

User simulations have been used in a wide variety of research studies on human-computer interaction systems. In our research we focus on applying user simulations in spoken dialog system development. In this chapter, we review previous work on applying user simulations in different dialog system development tasks and point out important issues that remain to be solved. First, we review different user simulation models built in previous studies (2.1) and the evaluation measures which assess the quality of user simulations by comparing simulated user behaviors with human user behaviors (2.2). While these measures suggest which user simulation can mimic human-like user behaviors, we are also interested in which user simulation can perform the best in specific dialog system development tasks. Therefore, we examine two widely studied tasks, i.e., a dialog strategy learning task and a dialog system testing task. We review how user simulations are used in these two tasks in Section 2.3 and 2.4 respectively.

2.1 USER SIMULATION MODELS

In this section, we review a class of probabilistic user simulations since these simulations are widely used in different studies and are also the focus of our research. We will further talk about other types of user simulations in Chapter 6.

Current user simulations mostly work on the dialog act level. Instead of trying to simulate fully natural utterances, these user simulations simply generate the dialog act of the student's next action, which in many cases is sufficient to continue the interaction between users and dialog systems. Usually, a probabilistic model is used by the user simulations to generate user actions given a certain context representation. (Eckert et al., 1997) first suggest a bigram model to predict the next user's action based on the previous system's action. While this model is simple and domain independent, it sometimes generates actions that do not make sense in the local context. In order to improve the model, (Levin et al., 2000) add constraints to this bigram model to only accept the expected dialog acts. However, their basic assumption of making the next user's action dependent only on the system's previous action is oversimplified. Later user simulations introduce the concept of *user goal* to make sure the user actions are consistent. (Scheffler, 2002) introduces fixed goal structures to hard-code all the possible paths of users' actions into a network. He trains the parameters of the network from training data for further predictions. (Pietquin, 2004) explicitly models the dependencies between a user's actions and his/her goal by conditioning the probabilities used by Levin et al. on a representation of the user goal. Both Scheffler's and Pietquin's work involve lots of manual work and may become infeasible when there are a large number of user actions. As a large number of states can make learning intractable, (Georgila et al., 2005) try to overcome this problem by exploiting commonalities among different states. They use linear combinations of shared features to express the commonalities. (Schatzmann et al., 2007a) make use of the slot-filling structure of information providing dialogs and propose an agenda-based simulated user that updates its goal and agenda based on the changes of dialog states. In a later work (Schatzmann et al., 2007b), they further describe an Expectation-Maximization approach to estimate the model parameters from a human user dialog corpus.

Other simulation models are built to simulate user behaviors at different levels rather than on the dialog act level alone. (Chung, 2004) uses a word-level user simulation to improve dialog development as well as to train understanding components. Similarly, (Schatzmann et al., 2007c) generate simulated user utterances on the word level while simulating word errors explicitly using the context-dependent acoustic confusability of words. (Filisko and Seneff, 2005) and (López-Cózar et al., 2003) build simulations on the acoustic level to train and evaluate speech recognition components.

There are also studies which generate user behaviors on multiple levels. For example, (Fukubayashi et al., 2006) build a simulated user from three dimensions: skill level, knowledge level, and the degree of urgency in order to generate adaptive user responses. (Janarthanam and Lemon, 2008) model the knowledge of novice versus expert users. They simulate a binary value of users' frustration level according to simple hand-coded rules in addition to user actions. (Jung et al., 2009) integrate different data-driven modeling techniques on the intention, lexical, and prosodic levels to build a user simulation that mimics user behaviors at all levels.

In our research, we build several probabilistic models for our tutoring spoken dialog system in which user actions are conditioned on different dialog contexts. We generate student's utterances on the word level since generating student's dialog acts alone does not provide sufficient information for our tutoring system to decide the next system's action. However, since it is hard to generate a natural language utterance corresponding to each tutor's question, we use the answer sets from the human user corpus as the candidate answer sets for the simulated students. The answer sets are extracted from the logs of human subject experiments thus our simulated student answers include misrecognized utterances. In our simple model, we condition the student answers only on the previous tutor question, which is similar to the Levin model. However, in a more complex model, we add tutoring domain related features to make our simulated users behave more like human students. We define all our user simulations in Section 3.1 and provide more information on our complex simulation model in Section 4.2.1.

2.2 USER SIMULATION EVALUATION MEASURES

Evaluation measures that can be automatically obtained from simulated dialogs are proposed to quickly and repeatedly assess user simulation qualities without human involvement. To date, there are no generally accepted evaluation methodologies to assess how realistic or how human-like a simulated corpus is. Dialog length, goal achievement rate and goal completion length have been used in previous research (Scheffler and Young, 2001). A comprehensive set of task-independent quantitative evaluation measures is proposed by (Schatzmann et al., 2005a). They consider three groups of measures that can cover the statistical properties of dialogs regardless of the task domain. The first group investigates high level dialog features. These measures look into both how much information is transmitted in the dialog and how active the dialog participants are. The second group of measures analyzes the style of the dialog in terms of the frequency of different speech acts, the proportion of goal-directed and social dialog, and the user's degree of cooperativeness. The last group of measures examines the efficiency of the dialogs using goal achievement rates and goal completion times. As Schatzmann et al. point out, these measures are only introduced to cover a variety of dialog properties for comparing simulated dialogs against real ones but there is no specific range of values to qualify a simulated corpus to be sufficiently realistic. We will investigate these set of evaluation measures further in Section 4.1.

Since it is hard to draw a single conclusion from the evaluation results based on a group of measures, (Williams, 2007) takes up the problem of a single quality measure for a user simulation. Williams defined a performance score for each dialog. Then, divergence between the distribution of the scores on the simulated corpus and the distribution on the human user corpus is computed as the measure of the simulation quality. A table of critical values is also given to interpret the statistical significance of comparisons between different corpora. (Rieser and Lemon, 2006) propose another single measure SUPER (Simulated User Pragmatic Error Rate) to take into account the varying behaviors that the simulation model generates. They argue that the goal of evaluating the quality of a simulated corpus is not to measure how well the simulation models can resemble the behavior of an average user. Instead, the evaluation must cover aspects of naturalness and variety of all human user behaviors.

Since evaluations using automatic measures are more time and cost efficient than evaluations with human judges, automatic evaluation measures are widely used in many other natural language processing tasks besides evaluating user simulations, such as the BLEU score (Papineni et al., 2002) in machine translation, the ROUGE score (Lin, 2004) in summarization, and the intrinsic metrics in facial expression generation (Foster, 2008). More importantly, those automatic evaluation measures are validated by showing that the conclusions drawn from the automatic measures are consistent with conclusions by human judges. Both the BLEU and the ROUGE scores are shown to be highly correlated with human ratings, while the intrinsic metrics rank system outputs in the same order as human preferences. Although automatic evaluation measures are shown to be useful in assessing simulated user behaviors, these measures have not been validated using human judgments. In Section 4.1, we will validate a group of previously proposed evaluation measures in our task domain.

The user simulation evaluation studies we reviewed above assess the quality of user simulations by comparing the simulated user behaviors with human user behaviors, assuming user simulations which generate more human-like behaviors are better. In Section 5, we also evaluate user simulations based on how helpful they are in specific dialog system development tasks because our goal of constructing user simulations is to assist dialog system development.

2.3 USER SIMULATION FOR DIALOG STRATEGY LEARNING

Dialog strategies are very important in dialog systems since they define how dialog systems behave. Besides manually coding dialog strategies, several machine learning approaches are explored to learn dialog strategies automatically. For example, (Horvitz and Paek, 1999) design a hierarchical Bayesian network which handles users' uncertainties on different decision levels. (Henderson et al., 2005) explore using supervised learning to generate a new dialog strategy from an example dialog corpus. Recently, reinforcement learning is widely used in learning dialog strategies automatically due to its nature of handling problems of optimization and planning of sequences of actions given uncertain observations, which is the common context in spoken dialog systems.

Markov Decision Process (MDP) is a widely used model when using reinforcement learning to design dialog strategies. It is also the model that we use in this research in learning new dialog strategies. MDP has four main components: states (s) which specifies a finite number of conditions that the dialog system can encounter, actions (a) which specifies a group of actions the dialog system can make in a certain state, a policy π which specifies the best action to take in a state, and a reward function (R) which specifies the utility of each state and the process as a whole (Lemon and Pietquin, 2007). The goal of applying MDP is to find the best policy p for a given state and action space to maximize the delayed reward at the end of the dialog. In most of the dialogs, the exact reward for each state is not known immediately. Thus, an expected cumulative reward is calculated for each state based on the delayed final reward function. In this view, the problem of designing a dialog strategy is formalized to an optimization problem in a mathematical framework. Then, we can apply optimization techniques (e.g., dynamic programming) to find the provably optimal dialog strategy.

While MDP provides a principled way for reinforcement learning, it does not take into account the uncertainty when defining dialog states. However, uncertainty always exists in a deployed dialog system due to speech recognition errors and the ambiguous nature of human language. Therefore, it is more appropriate to associate dialog states with probabilities to indicate that the current dialog context is only partially observable. In a Partially Observable MDP (POMDP), the policy π is based on the distribution over several possible dialog contexts at time t instead of one dialog context in the case of MDP. As a result, the optimal dialog action to perform at time t automatically takes account of the uncertainty in the context. However, applying POMDP is computationally expensive because it keeps track of all possible dialog contexts at all time. Summary POMDP (Williams and Young, 2005) and point-based value iteration (Williams and Young, 2007a) is introduced to make POMDP computationally feasible. While all the above approaches aim to learn a global optimal dialog strategy, (Cuayáhuitl et al., 2009) modularize dialogs into hierarchical sequences of sub-dialogs and divide the dialog strategy learning task into sub-tasks for each sub-dialog. They show that although optimal solutions may not be guaranteed with their approach, the sub-optimality will be well worth the gains in terms of scalability to large systems.

Since reinforcement learning needs to explore a large state space in order to calculate the best action sequence, user simulation is often used in generating a training corpus for this purpose. User simulations are usually trained from a small human user corpus (e.g., (Schatzmann et al., 2007a), (Georgila et al., 2006)). In the case of offline reinforcement learning, the simulation is used to interact with the old dialog system to generate a simulated corpus. After that, reinforcement learning is applied on the simulated corpus to generate a dialog strategy. Finally, the new dialog strategy is implemented in to a new dialog system. Dialog strategies can also be learned on the fly in online reinforcement learning while the simulated user is interacting with the dialog system (Lemon and Pietquin, 2007). In this case, the system adjusts its behaviors based on the strategies learned from the dialog history. In our experiments presented in this dissertation, we use MDP on a simulated dialog corpus to learn new dialog strategies offline since it is the simplest way to try out applying reinforcement learning on our tutoring dialog system. Most reinforcement learning techniques that are developed to design new dialog strategies were tested on information providing systems. Therefore, when applying these techniques in a new domain, the tutoring domain, we choose to start with the most basic reinforcement learning infrastructure. We will explore POMDP and online reinforcement learning in the future.

Previous studies apply different types of user simulation models to learn dialog strategies for different dialog systems. It is generally believed (Georgila et al., 2006) that a user simulation that does not act identically in similar states is needed to generate a training corpus for applying reinforcement learning to learn new dialog strategies. However, this hypothesis has not been tested directly by comparing the dialog strategies learned from different simulated corpora. In our study (Section 5.1.3), we compare different user simulations on designing new dialog strategies for the same dialog system to investigate which user simulation performs the best in the dialog strategy learning task.

2.4 USER SIMULATION FOR DIALOG SYSTEM TESTING

As spoken dialog systems are widely used in our daily lives, there is an increasing need to assess these systems' performance systematically. Based on the purposes of the evaluation, (Möller et al., 2007) summarizes previous research into the performance evaluation which measures the system component performance with respect to one or more criteria, the adequacy evaluation which determines whether the system fits with the purpose it has been designed for, and the diagnostic evaluation which tests the robustness of the system with all possible inputs. Typically, a dialog system evaluation serves all these purposes. Möller et al. also point out the two types of measures that are generally used in dialog system evaluations: a set of subjective measures (also called user satisfaction scores) which are collected from human users in a survey regarding the quality of the service that the system delivers to the users and a set of objective measures (also called interaction parameters) which can be obtained from interaction logs. The SASSI survey developed by (Hone and Graham, 2000) provides a guideline for collecting user satisfaction scores in spoken dialog systems. Similarly, (Möller, 2005a) summarizes the interaction parameters that are defined in most dialog system evaluations.

Since it is important to bridge the gap between the observed performance measured by the objective evaluation measures and the perceived performance measured by the subjective evaluation measures, many studies explore the connections between the objective and subjective evaluation measures. PARADISE (Walker et al., 2000) is an evaluation framework which represents the user satisfaction as a weighted function of task based success measures and dialog based cost measures. Once a performance prediction model is trained for a dialog system, it can be used to predict new user satisfaction scores of modified systems. (Möller et al., 2008) experiment with different modeling algorithms and different input parameters on predicting the quality and usability of two spoken dialog systems. They also discuss the issue of reusing a prediction model to evaluate a new dialog system. While PARADISE can be viewed as a framework for macro-evaluation in which a dialog system is evaluated as a whole, (Edlund et al., 2009) give an overview of micro-evaluation methods which analyze how well different spoken dialog system components function.

Recently, user simulation is used to replace human users to test different components in spoken dialog systems. For example, (López-Cózar et al., 2003) use simulated users to evaluate two different recognition front-ends and two different dialog strategies to handle users' confirmations. (Schatzman et al., 2007a) and (Janarthanam and Lemon, 2008) use user simulations to test different dialog strategies and show that the objective measures extracted from user simulation logs provide the same information as the parameters obtained from human user interaction logs. In a more recent work, (Engelbrecht et al., 2009) show that user simulations can be used to predict differences between different system versions and user groups in system evaluation. In our study, we also use user simulations to test the learned dialog strategies in Section 5.1.

In addition to the objective measures, human users provide user satisfaction scores which are important subjective measures used in dialog system evaluations. Therefore, in order to completely replace human users with simulated users, we need to synthesize not only the objective measures but also the user satisfaction scores from the simulated dialogs. In Section 5.2.2, we show that user satisfaction scores can be successfully predicted using simulation data.

3.0 TESTBED SYSTEMS, CORPORA AND USER SIMULATIONS

In this section, we introduce the three dialog systems used in our experiments and the human user corpora collected with these systems. We also describe basic user simulations we built on two of the dialog systems. The majority of our experiments were conducted with the ITSPOKE system (3.1), a tutoring spoken dialog system which teaches students physics. The ITSPOKE system provides us with the opportunity to explore previously used spoken dialog system development techniques on a new task domain – the tutoring domain. Most previous research we reviewed in Section 2 is conducted on information providing dialog systems where the systems provide information upon users' requests. In the ITSPOKE system, the information provided to the users is designed to improve the users' knowledge on physics instead of simply responding to the users' requests. In addition, since the users gain new knowledge while interacting with the ITSPOKE system, the user behaviors are different from those observed in information providing dialog systems where users' knowledge does not change. We show in our experiments that the techniques developed in information-providing dialog systems (e.g., user simulation evaluation measures (4.1), dialog strategy learning techniques (5.1)) can be successfully used in tutoring dialog systems as well. However, we also investigate domain-dependent features to better construct and evaluate user simulations for tutoring dialog systems (4.2). In Section 3.1, we introduce three user simulations built for the ITSPOKE system which condition user actions on different dialog contexts using probabilistic models similar to (Levin et al., 2000). These three simulations represent different approaches to characterize dialog contexts, varying from the most naïve approach (a random approach) to a more sophisticated approach which utilizes domain knowledge. We build yet another simulation for the ITSPOKE system in Section 5.1.3. Since the approach of constructing that simulation is motivated by special needs of using user simulation in a dialog strategy learning task, we will describe that user simulation later when we talk about learning new dialog strategies for the ITSPOKE system.

One of our experiments (6.2) was conducted on the LET'S GO system (3.2). This is a system that provides bus information for the general public in the Pittsburgh area. Since this system is actually used by the public, it provides us data with real users[1] (which is not available in the ITSPOKE system) to compare with data collected with human subjects. We discuss the differences of the two types of corpora and infer their impacts on training user simulations for different dialog system development tasks in Section 6.2. No user simulations were built for the LET'S GO system due to technical constraints at the time of our experiment.

Another experiment was conducted on the CHAT system (3.3). This system provides restaurant information and is evaluated by a user satisfaction survey. User satisfaction survey is a widely-used approach to obtain users' subjective opinion on dialog system performance (Möller, 2005b). It provides useful information such as task ease, user expectations, and system output quality which can be used to improve dialog systems. High user satisfaction scores are highly desired for information providing dialog systems since these systems' main function is to provide information to satisfy users' needs. However, in tutoring dialog systems such as ITSPOKE, enabling the students to learn more knowledge is often given a higher priority than satisfying the users' needs. Sometimes, system actions which usually result in low user

[1] The "real users" here refers to users who use a dialog system due to their own needs, contrasting to "human subjects" who are recruited to use the dialog system in experiments.

satisfaction even positively correlate with the amount of knowledge that users obtain (Litman and Forbes-Riley, 2005). Since low user satisfaction does not necessarily imply poor performance of a tutoring dialog system, when we conducted the experiment in Section 5.2.2 in which we need to link user satisfaction with the dialog system performance, we use an information providing system, i.e., the CHAT system. The user simulation built for the CHAT system adopts the same approach as in (Schatzmann et al., 2007a). More details on the choice of user simulation model and simulation construction are given in Section 3.3.

3.1 THE ITSPOKE SYSTEM

ITSPOKE (Intelligent Tutoring SPOKEn dialog System) (Litman and Silliman, 2004) is a spoken dialog tutor built on top of the Why2-Atlas text-based conceptual physics tutoring system (VanLehn et al., 2002). In each tutoring session, students first read a small document of background physics material, and then work through 5 problems[2] with the system. The five physics problems are discussed following the same procedure: first a student types an essay answering a qualitative physics question; then a tutoring dialog is initiated by ITSPOKE after analyzing the essay to correct misconceptions and to elicit further explanations. After that, the student revises the essay, thereby ending the tutoring or causing another round of tutoring/essay revision. A pre-test is given before the tutoring session and a post-test is given afterwards. The student's Normalized Learning Gain (NLG) is computed using the following formula: NLG = (postTestScore-preTestScore)/(1-preTestScore). On average, the students reach an NLG of 0.4.

[2] See Appendix A for the problem statements of the 5 problems.

By comparing pretest and post test scores using 2-tailed t-tests, we observe that students learned significantly ($p<0.05$) after interacting with ITSPOKE.

Corpora and Annotations

In our research, we use three corpora of tutoring dialogs that were collected in two prior studies with two different groups of subjects (Litman et al., 2004) (Forbes-Riley et al., 2006). The two groups were recruited on the University of Pittsburgh campus in fall 2003 and spring 2005 separately. The 2003 group consists of students recruited primarily via posted flyers only, whereas the students in the 2005 group were recruited via flyers as well as via the instructor of a large introduction to psychology class. Subjects have never taken college-level physics. The main components of ITSPOKE remain the same in the 2005 experiment as in 2003, but a slightly different language model is used and some bugs are fixed as well. Also, the system uses a synthesized voice in all 2003 experiments, though in the 2005 experiments, one half of the experiments use the synthesized voice and the other half use a pre-recorded voice. Table 1 shows an overview of the collected corpora.

Table 1. Overview of ITSPOKE corpora

Corpus		Student population	System difference	Number of dialogs	Available Information
f03		2003	Synthesized voice	100	Manual & Automatic Correctness, Manual Certainty
s05	syn	2005	Synthesized voice	136	Automatic Correctness, Manual Certainty on some of the dialogs
	pre	2005	Pre-recorded voice	135	Automatic Correctness, Manual Certainty on some of the dialogs

We also use some of the annotations that were developed in prior studies[3]. In the ITSPOKE dialogs, correctness (correct(**c**), incorrect(**ic**)) can be automatically computed from the system logs. An annotator also manually tagged the correctness of student answers based on human transcripts on the f03 corpus. A comparison of the system tagged correctness and the human annotated correctness reaches an agreement of 90% with a Kappa of 0.79 (Rotaru, 2008). Dialogs collected in 2003 and part of the dialogs collected in the 2005 experiments were also manually annotated for certainty (certain, uncertain, neutral, mixed) in each student utterance based on both lexical and prosodic information. Table 2 shows an example of coded dialog excerpt[4]. To test the reliability of the certainty annotation, a second annotator tagged the same corpus for uncertain and not-uncertain. A comparison of the inter annotator's agreement based on this binary classification yields a kappa of 0.68 (Forbes-Riley et al., 2006).

Table 2. Sample coded dialog excerpt with ITSPOKE

ITSPOKE:	Which law of motion would you use?
Student:	Newton's second law? [ic, uncertain]
ITSPOKE:	The best law to use is Newton's third law. Do you recall what it says?
Student:	For every action there is an equal and opposite reaction? [c, uncertain]

User Simulations for the ITSPOKE system

We build three user simulations for the ITSPOKE system using different probabilistic models. All models are derived from (Levin et al., 2000), but vary on their efforts on mimicking human user behaviors. Our goal is not to propose new simulation models which outperform previous simulations, but to compare simple probabilistic models that can be used in different task domains on different dialog system development tasks to illustrate what type of simulation is

[3] See (Rotaru, 2008) for a complete list of available annotations on the ITSPOKE corpora.
[4] Punctuation is added in student utterances to represent the prosody in the dialog.

appropriate on a specific task as well as what the basic factors are in constructing probabilistic user simulations.

Most previous work we reviewed in Section 2.1 generates a student's move by outputting an abstract class of a student's act, i.e., a dialog act. However, we generate simulated student utterances on the word level by picking up recognized human student answers from the collected human user dialog corpora since generating student dialog acts alone does not provide sufficient information for our tutoring system to decide the next system action.

Random Model (RAN). The Random model, as a simple unigram model, randomly picks a student's utterance from the real corpus as the answer to a tutor's question, neglecting which question it is. An excerpt of the simulated corpus by the Random model is given in Table 3. We see that the student answer can be totally irrelevant to the tutor's question in the sample dialog.

Table 3. Simulated dialog by the Random Model

ITSPOKE:	The best law of motion to use is Newton's third law. Do you recall what it says?
Student:	More. [Incorrect]
ITSPOKE:	Newton's third law says …
	[Dialog goes on]
ITSPOKE:	Do you recall what Newton's third law says?
Student:	Down. [Incorrect]

Correctness Model (COR). The Correctness model is designed to give a correct/incorrect answer with the same probability as the average of real students. For each tutor question, we automatically compute the average correctness rate of real student answers from the system logs. In addition, we put all the real student answers to this tutor question in a candidate answer set. Therefore, when the Correctness model receives a tutor question, it first decides whether to give a correct or incorrect answer based on the computed correctness rate; then, a correct/incorrect answer is randomly chosen from the candidate answer set for this tutor question. We implement a back-off mechanism to count possible answers that do not appear in the real corpus. An excerpt of the simulated corpus by the Correctness model is given in Table 4. Since the simulation model decides the student answer only on the previous tutor question, we can see that the student's answer could be inconsistent across the dialog as shown in the sample dialog.

Table 4. Simulated dialog by the Correctness Model

ITSPOKE:	The best law of motion to use is Newton's third law. Do you recall what it says?
Student:	Yes, for every action, there is an equal and opposite reaction.[correct]
ITSPOKE:	This is correct ...
[Dialog goes on]	
ITSPOKE:	Do you recall what Newton's third law says?
Student:	No. [Incorrect]

Cluster Model (CLU). The Cluster model ensures that the simulated student behaves more consistently than the Correctness model. As shown in Table 5, the student gains new knowledge in the first part of the dialog and applies the knowledge in the second part of the dialog. We will describe the mechanism of this simulation model in detail in Section 4.2.

Table 5. Simulated dialog by the Cluster Model

ITSPOKE:	The best law of motion to use is Newton's third law. Do you recall what it says?
Student:	Yes, for every action, there is an equal reaction.[partially correct]
ITSPOKE:	Newton's third law says …
[Dialog goes on]	
ITSPOKE:	Do you recall what Newton's third law says?
Student:	Yes, for every action, there is an equal and opposite reaction. [correct]

We hypothesize that the ranking of the three simulations from the most realistic to the least is: CLU, COR, RAN.

3.2 THE LET’S GO SYSTEM

The Let’s Go bus information system is a telephone-based dialog system that provides schedule information for buses in the Pittsburgh area (Raux et al., 2005). When the customer service line of the Port Authority of Allegheny County (which manages buses in Pittsburgh) is not staffed by operators (i.e. from 7pm to 6am on weekdays and 6pm to 8am on weekends), callers are redirected to Let’s Go. An example dialog with Let’s Go is shown in Table 6. The interaction with the system itself starts with an open prompt (“What can I do for you?”) followed by a more directed phase where the system attempts to obtain the missing information (origin, destination, travel time, and optionally route number) from the user. Finally, the system provides the best matching bus number and time, at which point the user has the possibility of asking for the next/previous buses. Most of the dialogs are not manually transcribed or annotated. A task success rate of 67% is estimated based on a small portion of transcribed dialogs.

Table 6. Example dialog with LET’S GO

S: Welcome to the CMU Let’s Go Bus information system. What can I do for you?
U: Uh I need to get from Fifth and Bigelow.
S: Leaving from Fifth and Bigelow. Did I get that right?
U: Yes.
S: Right. What is your destination?
U: Airport.
S: Going to airport. Did I get that right?
U: Yes.
S: Okay. When are you going to take that bus?
U: Nine P UH … Seven P M
S: Leaving at 7 p.m. Is this correct?
U: Yes
S: Right Just a minute. Let me check that for you. There is a 28 X leaving Fifth avenue at Bigelow …

Corpora and Annotations

As of April 2007, the system had received more than 34,000 calls from the general public. In our study, we obtained a small user corpus of 200 dialogs by randomly choosing two weekdays (which contains 90 dialogs) and a weekend (which contains 110 dialogs). In addition, we collected another corpus with recruited subjects in a standard experimental setting (see (Ai et al., 2007b) for details). 39 Subjects (19 female and 20 male) came into our lab to complete 3 task scenarios in a fixed order, producing 102 dialogs[5]. The task success rate in the subject corpus is 80.7% according to manual annotation. Table 7 shows an overview of the two corpora.

Table 7. Overview of Let's Go Corpora

Corpus	User population	System difference	Number of dialogs	Available Information
User	Real Users	none	200	Automatic features
Subject	Recruited Subjects	none	102	Automatic features

Both the real user corpus and the recruited subject corpus were not manually transcribed or annotated. A set of automatic features (described in Section 6.2) are extracted from the system logs to compare the two corpora. Based on the corpus comparison results, we infer the impact of the two corpora on the behaviors of user simulations trained from them in Section 6.2. No user simulations were built for the LET'S GO system for direct comparisons among user simulations due to technical constraints at the time of the experiment.

[5] Some subjects mistakenly completed more than one task per dialog. Such multi-task dialogs were excluded.

3.3 THE CHAT SYSTEM

CHAT (Conversational Helper for Automotive Tasks) is a spoken dialog system that supports navigation, restaurant selection and mp3 player applications. The system is specifically designed for users to interact with devices and receive services while performing other cognitive demanding primary tasks such as driving. In our study, we focus on a dialog corpus collected in the final evaluation on the restaurant domain prior to our study (Weng et al., 2006). In the evaluation, the system reached a task completion rate of 94%. A user satisfaction survey was also given in the final evaluation. Users were asked to rate the conversation from 6 perspectives, each on a 5-point scale: whether the system was easy to use, whether the system understood the user well, whether the interaction pattern was natural, whether the system's actions were appropriate, whether the system acted as expected, and whether the user was willing to use the system on a regular base. A user satisfaction score was computed as the average of the 6 ratings. The average satisfaction score of all the users is 3.8.

Corpora and Annotations

Several corpora were collected with the CHAT system in the restaurant domain (Weng et al., 2006) but we only use the final evaluation corpus in this study. Table 8 shows an overview of the collected corpora.

Table 8. Overview of CHAT Corpora

Corpus	**User population**	**System difference**	**Number of dialogs**	**Available Information**
Final Evaluation	Recruited subjects Jan 19-31, 06	Final system	180	Manual semantic info, User satisfaction

Table 9 shows a sample dialog with CHAT. The correct words are shown in square brackets immediately after the misunderstood words (e.g., in Utterance “User1”). Nine tasks of restaurant selections were used in the final evaluation using 12 constraints in total (e.g., cuisine type, price level, location). These 12 constraints are spread across the nine tasks evenly with three constraints per task. These dialogs are manually tagged for semantic information which will be described in detail in Section 5.2.2.

Table 9. A Sample Dialog with CHAT

User1: I’m looking for a five star Thai [Italian] restaurant accept credit cards
System1: There are 7 restaurants that have a five star rating, accept credit cards, and serve Thai food in my database. Here are the first few: TeJUS, Slam Corner Cate, Setha’s Restaurant. Should I continue?
User2: I need to look for Italian restaurant
System2: Are you looking for a restaurant that serves Italian food?
User3: yes
System3: I found 38 restaurants that serve Italian food, accept credit cards and have a five star rating, for example, Toto Café …
User4: I’m done.

User Simulation for the CHAT system

As we reviewed in Section 2.1, (Schatzmann et al., 2007a) defines a user simulation which completes a set of pre-defined task-oriented goals by updating its agenda and the current goal based on the changes of the dialog states. We believe that this type of simulated user can be directly used with the CHAT system since the constraints of restaurant requests designed for human users can be mapped into a list of goals for simulated users. Table 10 shows a simulated dialog with the CHAT system. The second column shows the simulated student’s agenda. The simulated user updated the agenda after each system turn. At the beginning of the dialog, the simulated user randomly generates an agenda for the list of the goals corresponding to the three constraints in requesting a restaurant. In this dialog, the simulated user chose to express Goal1 and Goal2 in the first utterance, and then express Goal3. During the dialog, the simulated user updates its list of goals by removing the constraints that have been understood by the system. For example, after the Turn "System1", Goal1 is removed from the simulated user's agenda. New actions are added according to the last system's question (such as requesting the user to repeat the last utterance) as well as the simulated user's current goals. The actions that address the last system's question are given higher priorities then other actions in the agenda. For example, when the dialog system fails to understand the last user utterance and thus requests a clarification (Turn "System2"), the simulated user satisfies the system's request before moving on to discuss a new constraint (Turn "User3"). The simulated user we implemented interacts with the CHAT system on the word level. It generates a string of words by instantiating its current action using predefined templates derived from previously collected corpora with human users. Random lexical errors are added to simulate a word error rate of 15% and a semantic error rate of 11% based on previous experience (Weng et al., 2006).

Table 10. A simulated dialog with the CHAT system

Utterances	Agenda
User1: I am looking for an expensive Japanese restaurant	Step 1: Goal1 (cuisine type=Japanese) & Goal2 (price range=expensive) Step 2: Goal3 (location=Palo Alto)
System1: There are 15 restaurants that serve Japanese food …	
User2: I am looking for an expensive restaurant	Step 1: Goal2 (price range=expensive) Step 2: Goal3 (location=Palo Alto)
System2: Are you looking for an expensive Japanese restaurant?	
User3: yes	Step 1: Answer Yes/No Question Step 2: Goal3 (location=Palo Alto)
System3: OK. There are 7 expensive Japanese restaurants …	
User4: I am looking for an restaurant in Palo Alto	Step1: Goal3 (location=Palo Alto)
System4: I found 3 expensive Japanese restaurants in Palo Alto. They are…	

4.0 USER SIMULATION EVALUATION MEASURES

As we build different types of user simulations, there comes an increasing need to develop user simulation evaluation measures in order to check whether the quality of user simulations meet our design expectations. Since user simulations are essentially models of human user behaviors, one intuitive way to assess user simulations is to compare simulated user behaviors with human user behaviors to see how well the simulated users mimic human users. In this section, we introduce domain-independent and domain-dependent evaluation measures for such comparisons. We reviewed the automatic evaluation measures used in prior research in assessing user simulation qualities in Section 2.2. Here, we adopt some of these measures to evaluate the user simulations we built for the ITSPOKE system (3.1). These measures aim to assess how human-like the simulated dialogs are without emphasizing domain-related dialog behaviors. These evaluation measures are shown to be useful in quickly evaluating user simulations without human involvement. However, the validity of these measures has not been proven yet. In Section 4.1.1, we collect human judgments to validate these measures. In addition, we build prediction models of human judgments using these automatic measures (4.1.2). We show that these domain-independent evaluation measures can provide the same ranking of our user simulations as human judges. However, we also observe that these simple evaluation measures are not sufficient to conclude whether a user corpus is realistic or not (4.1.3). Therefore, in section 4.2, we introduce a domain-dependent constraint for constructing and evaluating user simulations for tutoring dialogs and other dialogs where user knowledge is modified during user-system interactions. In 4.2.1, we use this domain-dependent constraint to build the most realistic simulation model for the ITSPOKE system – the CLU model we briefly introduced in Section 3.1. We also show that this constraint can be used as an evaluation measure to better compare simulated corpora against real user corpora. In 4.2.2, we further show that this domain-dependent constraint can be used to capture the similarity among realistic user behaviors that the previously proposed domain-independent measures cannot capture. Finally, we use this constraint as a predictive feature to improve the prediction model of human judgments in 4.2.3.

4.1 DOMAIN-INDEPENDENT EVALUATION MEASURES

The domain-independent evaluation measures proposed by (Schatzmann et al., 2006) are summarized in Table 11. The first column shows the original measures used by Schatzmann et al. The second and the third column show the corresponding measures in the ITSPOKE dialog corpora and their abbreviations. The asterisk (*) indicates a measure that is not available in the simulated corpora. Since we are not simulating student's essays at this stage, the simulated corpora only include dialogs of discussions before the first time the tutor asks for an essay revision. As a result, dialog style measures and dialog success rate measures are not available in simulated corpora. We are not simulating student's learning gains either; correctRate is the only learning measure used.

Table 11. Mapping between evaluation measures

Schatzmann et al.	Our measures	Abbreviation
High-level dialog features		
Dialog length (number of turns)	Number of student/tutor turns	S_turn, T_turn
Turn length (number of actions per turn)	Total words per student/tutor turn	S_wordRate, T_wordRate
Participant activity (ratio of system and user actions per dialog)	Ratio of system and user words per dialog	WordRatio
Dialog style and cooperativeness		
Proportion of goal-directed actions vs. others	Proportion of dialogs that discuss physics vs instructions about using system interface	Phy/non *
Number of times a piece of information is re-asked	Number of times a physics concept is re-discussed	repeatConcept*
Dialog Success Rate and Efficiency		
Average goal/subgoal achievement rate	Average number of essay submissions	essayRevision *
Learning features		
None	Percentage of correct answers	correctRate
None	Learning gains	Learning *

Among these measures, the first group of high-level dialog measures is applicable to both tutorial and other types of dialogs. These measures are also automatically retrievable from the simulated corpus. Since there is a large potential to use such easily obtainable measures to evaluate user simulations, we want to confirm the validity of these measures through a human assessment study in Section 4.1.1.

4.1.1 Human assessment study

In this section, we collect human judgments to directly assess the quality of user simulations[6]. We believe that this is a reliable approach to assess the simulated corpora. It is also an important step towards developing a comprehensive set of user simulation evaluation measures. First, we can estimate the difficulty of the task of distinguishing real and simulated corpora by knowing how hard it is for human judges to reach an agreement. Second, human judgments can be used as the gold standard of the automatic evaluation measures. Third, we can validate the automatic measures by correlating the conclusions drawn from the automatic measures with the human judgments. There are well-known practices which validate automatic measures using human judgments. For example, in machine translation, BLEU score (Papineni et al., 2002) is developed to assess the quality of machine translated sentences. Statistical analysis is used to validate this score by showing that BLEU scores are highly correlated with human judgments. In this section, we describe the human assessment study we conducted to collect human judgments on user simulation qualities. In the next section (4.1.2), we use the collected human judgments to validate the automatic evaluation measures.

Data

We follow the design of (Linguistic Data Consortium, 2005) in obtaining human judgments. We randomly picked 15 human student data from the ITSPOKE f03 corpus (described in 3.1) to create the real user corpus. We train the RAN, COR, and CLU model (introduced in 3.1) from all f03 data. Then, each user simulation interacts with the ITSPOKE f03 system 15 times to generate the data of 15 simulated users of each type. We used three out of the five physics problems

[6] Publications: The work presented in this section was published in (Ai and Litman, 2008).

(Prob34, Prob38, and Prob58)[7] from the ITSPOKE system to ensure the variety of dialog contents while keeping the corpus size small. Therefore, we have four user corpora (one from real user (REAL) and three from simulated users) to compare. In total, the evaluation corpus consisted of 180 dialogs, in which 15 dialogs were generated by each of the 4 user simulation models on each of the 3 problems.

We decided to conduct a middle-scale assessment study that involved 30 human judges. Judges are required to be native speakers of American English to make correct judgments on the language use and fluency of the dialog. They are also required to have taken at least one course on Newtonian physics to ensure that they can understand the physics tutoring dialogs and make judgments about the content of the dialogs. We conducted a small pilot study to estimate how long it took a judge to answer all survey questions (described below in **Survey Design**) in one dialog because we wanted to control the length of the study so that judges would not have too much cognitive load and would be consistent and accurate on their answers. Based on the pilot study, we decided to assign each judge 12 dialogs which took about an hour to complete. Each dialog was assigned to two judges.

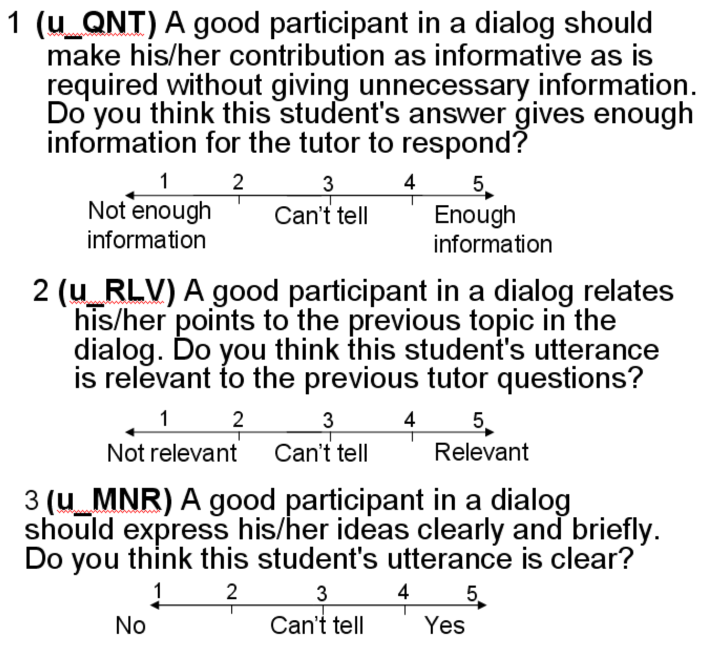
The judges involved in this study did not receive additional training on how to rate the simulated dialogs. Judges are instructed to work as quickly as comfortably possible. They are encouraged to provide their intuitive reactions and not to ponder their decisions. The same setup is adapted by (Walker et al., 2005) when they collected human judges' ratings on their natural language generator and by DUC on summarization evaluation (http://duc.nist.gov). (Harman and Over, 2004) analyze the effects of human variation in DUC summarization evaluation. They find

[7] See Appendix A for problem statements.

that despite large variations in human judgments, the ranking of the summarization systems remain constant. We observe similar results on our data.

Survey Design

We designed a web survey (See Appendix B) to collect human judgments on a 5-point scale on both utterance and dialog levels. Each dialog is separated into pairs of a tutor question and the corresponding student answer. Figure 1 shows the three questions which are asked for each tutor-student utterance pair. The three questions assess the quality of the student answers from three aspects[8] of Grice's Maxim (Grice, 1975): Maxim of Quantity (u_QNT), Maxim of Relevance (u_RLV), and Maxim of Manner (u_MNR). In the survey, we briefly describe the standards for a good conversation partner based on the three aspects of Grice's Maxim before presenting the corresponding question. This aims to serve as a quick training for the judges on how to assess the quality of a conversational dialog.



1 **(u_QNT)** A good participant in a dialog should make his/her contribution as informative as is required without giving unnecessary information. Do you think this student's answer gives enough information for the tutor to respond?

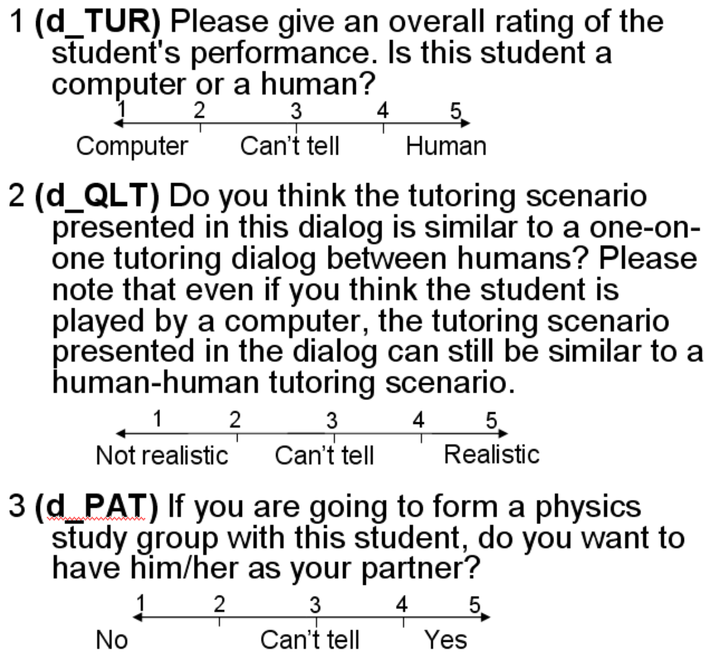
1
2
3
4
5
Not enough information
Can't tell
Enough information

2 **(u_RLV)** A good participant in a dialog relates his/her points to the previous topic in the dialog. Do you think this student's utterance is relevant to the previous tutor questions?

3 **(u_MNR)** A good participant in a dialog should express his/her ideas clearly and briefly. Do you think this student's utterance is clear?

Figure 1. Utterance level questions

[8] Another aspect in the Grice's Maxim, the maxim of quality, was not used because in a tutoring dialog a student can give both correct (good quality) and incorrect (bad quality) answers.



1 **(d_TUR)** Please give an overall rating of the student's performance. Is this student a computer or a human?

1 Computer, 2, 3 Can't tell, 4, 5 Human

2 **(d_QLT)** Do you think the tutoring scenario presented in this dialog is similar to a one-on-one tutoring dialog between humans? Please note that even if you think the student is played by a computer, the tutoring scenario presented in the dialog can still be similar to a human-human tutoring scenario.

1 Not realistic, 2, 3 Can't tell, 4, 5 Realistic

3 **(d_PAT)** If you are going to form a physics study group with this student, do you want to have him/her as your partner?

1 No, 2, 3 Can't tell, 4, 5 Yes

Figure 2. Dialog level questions

In Figure 2, we show the three dialog level questions which are asked at the end of each dialog. The first question (d_TUR) is a Turing test[9] type of question which aims to obtain an impression of the student's overall performance. The second question (d_QLT) assesses the dialog quality from a tutoring perspective. The third question (d_PAT) sets a higher standard on the student's performance. Unlike the first two questions which ask whether the student "looks" good, this question further asks whether the judges would like to partner with the particular student. We assume that the order of these three questions will not impact student answers since we believe that even if a judge thinks a student is a computer simulation, they may still think that

[9] We use the term of "Turing Test" fairly loosely here since our human judges are asked to decide whether the student presented in the dialog is a human or a computer by reading transcripts of the dialog alone. In a standard Turing Test, the judges would interact with the student in a live conversation to reach their decisions.

the simulated student behaves like a human and will be willing to partner with the simulated student if the student is an intelligent learner.

We display one tutor-student utterance pair and the three utterance level questions on each web page. After the judge answers the three questions, he/she will be led to the next page which displays the next pair of tutor-student utterances in the dialog with the same three utterance level questions. The judge reads through the dialog in this manner and answers all utterance level questions. At the end of the dialog, three dialog level questions are displayed on one webpage. We provide a textbox under each dialog level question for the judge to type in a brief explanation on his/her answer. After the judge completes the three dialog level questions, he/she will be led to a new dialog. This procedure repeats until the judge completes all of the 12 assigned dialogs.

Assessment Study Results

In the initial analysis, we observe that it is a difficult task for human judges to rate on the 5-point scale and the agreements among the judges are fairly low. Table 12 shows for each question, the percentages of pairs of judges who gave the same ratings on the 5-point scale. Therefore, in our analysis we collapse the "definitely" types of answers with its adjacent "probably" types of answers (more specifically, answer 1 with 2, and 4 with 5). We substitute scores 1 and 2 with a score of 1.5, and scores 4 and 5 with a score of 4.5. A score of 3 remains the same. The rest of our analysis is performed using the 3-point scale ratings.

Table 12. Percent agreement on 5-point scale

d_TUR	d_QLT	d_PAT	u_QNT	u_RLV	u_MNR
22.80%	27.80%	35.60%	39.20%	38.40%	38.70%

Inter-annotator agreement. Table 13 shows the inter-annotator agreements on the collapsed 3-point scale. The first column presents the question types. In the first row, "diff" stands for the differences between human judges' ratings. The column "diff=0" shows percent agreement among judges. The column "Kappa" shows the unweighted kappa agreements and the column "Kappa*" shows the linear weighted kappa. A weighted kappa is computed to take into account distances among different rating categories. For example, if Judge A's rating falls in the category "score=1.5", Judge B's rating in "score=3", and Judge C's rating in "score=4.5", although Judge A disagrees with both Judge B and Judge C, he disagrees with Judge C further than with Judge B. We can capture this difference by defining distances between the adjacent categories when calculating a weighted kappa. When calculating the linear weighed kappa, the adjacent answer distance is assigned to be one[10]. Therefore, in our example, the distance between Judge A and B is 1, while the distance between judge A and C is 2.

Note that we randomly picked two judges to rate each dialog so that different dialogs are rated by different pairs of judges and one pair of judges only worked on one dialog together. Thus, the kappa agreements here do not reflect the agreement of one pair of judges. Instead, the kappa agreements show the overall observed agreement among every pair of judges controlling for the chance agreement. We compute the kappa agreements by constructing the confusion matrix for each question. Table 14 shows an example of the confusion matrix for d_TUR. The first three rows of the first three columns show the counts of judges' ratings on the 3-point scale. For example, the first cell shows that there are 20 cases where both judges give 1.5 to the same dialog. We observe that human judges have low agreements on all types of questions, although the agreements on the utterance level questions are better than the dialog level questions. This

[10] We also calculated the quadratic weighted kappa in which the distances are squared and the kappa results are similar to the linear weighted ones. For calculating the two weighted kappas, see http://faculty.vassar.edu/lowry/kappa.html for details.

observation indicates that assessing the overall quality of simulated/real dialogs on the dialog level is a difficult task. The lowest agreement appears on d_TUR.

Table 13. Agreements on 3-point scale

Q	diff=0	diff=1	diff=2	Kappa	Kappa*
d_TUR	35.0%	45.6%	19.4%	0.022	0.079
d_QLT	46.1%	28.9%	25.0%	0.115	0.162
d_PAT	47.2%	30.6%	22.2%	0.155	0.207
u_QNT	66.8%	13.9%	19.3%	0.377	0.430
u_RLV	66.6%	17.2%	16.2%	0.369	0.433
u_MNR	67.5%	15.4%	17.1%	0.405	0.470

Table 14. Confusion Matrix on d_TUR

	score=1.5	score=3	score=4.5	sum
score=1.5	20	26	20	66
score=3	17	11	19	47
score=4.5	15	20	32	67
sum	52	57	71	180

We look further into d_TUR because it is the only question that we know the ground truth. Remember in d_TUR we asked the judges whether the student presented in the dialog is a human or a computer. This information was already known at the time we picked the dialogs from the simulated or the human user corpus. We compute the accuracy of human judgment as (number of ratings 4&5 on real dialogs + number of ratings of 1&2 on simulated dialogs)/(2*total number of dialogs). The accuracy is 39.44%, which serves as further evidence that it is difficult to discern human from simulated users directly.

We also look into which dialogs are more difficult for the human judges to reach good agreements on by dividing the dialog corpus into subsets based on the types of simulation models and the contents of physics problems separately. Table 15 shows judges' percent agreements on dialogs generated by each of the four student models. We can see that the judges' agreements on dialogs by the real students are higher than the dialogs by any simulation models.

We hypothesize that human users behave more consistently so that it is easier for judges to reach agreements on real student dialogs. This hypothesis remains to be confirmed by future studies.

Table 15. Human judge percent agreements on 3-point scale

Q	REAL	CLU	COR	RAN
d_TUR	37.8%	35.6%	31.1%	33.3%
d_QLT	57.8%	42.2%	44.4%	40.0%
d_PAT	71.1%	31.1%	37.8%	48.9%
u_QNT	71.7%	67.4%	67.6%	61.2%
u_RLV	74.2%	70.5%	66.9%	56.2%
u_MNR	71.4%	71.1%	67.8%	60.9%

Table 16 shows human judges' agreements on each of the three physics problems. Differences among judges' agreements on different dialog contents are small, although there are some systematic differences among the dialogs on different problems[11]. For example, dialogs on prob34 are significantly shorter than other dialogs. Also, dialogs on prob58 have significantly higher correctness rates. However, since our tutorial dialogs have similar and constrained dialog structures due to our dialog system design, whether human judges' agreements will vary based on the content of dialogs remains to be an open question to be investigated in the future in longer and more sophisticated dialogs.

Table 16. Human judge agreements on each physics questions on 3-point scale

Q	prob34	prob38	prob58
d_TUR	31.7%	41.7%	31.7%
d_QLT	40.0%	56.7%	41.7%
d_PAT	53.3%	46.7%	41.7%
u_QNT	67.1%	66.9%	68.7%
u_RLV	67.0%	66.4%	68.8%
u_MNR	68.2%	68.5%	67.7%

[11] See Appendix C for examples of human student dialogs on Problem 34, Problem 38, and Problem 58.

The judges we used in this study are not systematically trained to rate tutorial dialogs. Although we provided some general guidelines to the judges, different judges may still have different voting patterns, i.e., some judges are more conservative in the ratings while other judges are more generous. In order to standardize the ratings on a similar scale with comparable values across judges, we use the approach proposed by (Blatz et al., 2004) which has been applied on standardizing human judges' ratings on machine translation studies. In this approach, we first transform each rating into a quantile. For each judge J and score s, the standardized rating is:

$$x = f(s, J) = P(S < s \mid J)$$

where $P(S < s \mid J)$ is the probability that judge J would assign a score lower than s. In order to estimate this probability from the discrete 5-point scale, we accumulate all ratings that are strictly below s, and half of the ratings that are equal to s. The ratio of these accumulated ratings to the total number of ratings is the standardized score. Let n(s) be the number of ratings of judge J that are equal to s, then for judge J, the standardized rating x is approximated by

$$x' = \frac{\sum_{i<s} n(i) + n(s)/2}{\sum_{i=1}^{5} n(i)}.$$

We recalculate the inter-annotator agreements using the standardized ratings in Table 17. Both standardized rating agreements on the 5-point scale and on the 3-point scale are slightly better than the original rating agreements but are still low. Since standardized human ratings do not change our results, we continue to use the original ratings in the rest of our analysis.

Table 17. Standardized rating percent agreements

Scale	d_TUR	d_QLT	d_PAT	u_QNT	u_RLV	u_MNR
5-point	25.1%	28.3%	37.1%	38.9%	39.1%	39.3%
3-point	36.1%	48.5%	49.4%	65.3%	67.9%	68.7%

To summarize, we observe that human judges' agreements on all types of questions are low. We investigate the low agreements by looking into judges' explanations on the dialog level questions. 21% of the judges find it hard to rate a particular dialog because that dialog is too short or the student utterances mostly consist of one or two words. There are also some common false beliefs among the judges. For example, 16% of the judges think that humans will say longer utterances while 9% of the judges think that only humans will admit the ignorance of an answer. Since human judges' ratings have low agreement, next we look into an alternative way to interpret human judgments.

Ranking of the models. In Table 18, the first column shows the name of the questions; the second column shows the name of the models; the third to the fifth column present the percentages of judges who choose answer 1 and 2, can't tell, and answer 4 and 5. For example, when looking at the column "1 and 2" for d_TUR, we see that 22.2% of the judges think a dialog by a real student is generated probably or definitely by a computer; more judges (25.6%) think a dialog by the cluster model is generated by a computer; even more judges (32.2%) think a dialog by the correctness model is generated by a computer; and even more judges (51.1%) think a dialog by the random model is generated by a computer. When looking at the column "4 and 5" for d_TUR, we find that most of the judges think a dialog by the real student is generated by a human while the fewest number of judges think a dialog by the random model is generated by a human. Given that more human-like is better, both rankings support our hypothesis that the quality of the models from the best to the worst is: REAL, CLU, COR, RAN. In other words, although it is hard to obtain well-agreed ratings among judges, we can combine the judges' ratings to produce the ranking of the models. We see consistent ranking orders on d_QLT and d_PAT as well, except for a disorder of cluster and correctness model on d_QLT indicated by the underlines.

Table 18. Rankings on Dialog level Questions

Question	model	1 and 2	can't tell	4 and 5
d_TUR	REAL	22.2%	28.9%	48.9%
	CLU	25.6%	31.1%	43.3%
	COR	32.2%	26.7%	41.1%
	RAN	51.1%	28.9%	20.0%
d_QLT	REAL	20.0%	10.0%	70.0%
	CLU	21.1%	20.0%	<u>58.9%</u>
	COR	24.4%	15.6%	<u>60.0%</u>
	RAN	60.0%	18.9%	21.1%
d_PAT	REAL	28.9%	21.1%	50.0%
	CLU	41.1%	17.8%	41.1%
	COR	43.3%	18.9%	37.8%
	RAN	82.2%	14.4%	3.4%

When comparing two models, we can tell which model is better from the above rankings. Nevertheless, we also want to know how significant the difference is. We use t-tests to examine the significance of differences between every two models. We average the two human judges' ratings to get an averaged score for each dialog. For each pair of models, we compare the two groups of the averaged scores for the dialogs generated by the two models using 2-tail t-tests at the significance level of $p < 0.05$. In Table 19, the first row presents the names of the models in each pair of comparison. "Sig" means that the t-test is significant after Bonferroni correction; question mark ("?") means that the t-test is significant before the correction, but not significant afterwards, we treat this situation as a trend; "not" means that the t-test is not significant at all. The table shows that only the random model is significantly different from all other models. The correctness model and the cluster model are not significantly different from the real student given the human judges' ratings, neither are the two models significantly different from each other.

Table 19. T-test results on human judges' ratings between every two models

	real-ran	real-cor	real-clu	ran-cor	ran-clu	cor-clu
d_TUR	sig	not	not	sig	sig	not
d_QLT	sig	not	not	sig	sig	not
d_PAR	sig	?	?	sig	sig	not
u_QNT	sig	not	not	sig	sig	not
u_RLV	sig	not	not	sig	sig	not
u_MNR	sig	not	not	sig	sig	not

To summarize, although it is hard for human judges to reach good agreements on dialog ratings, the ratings give consistent ranking on the quality of the real and the simulated user models. Next, we build a prediction model using automatic evaluation features to predict the rankings of the real and simulated user models. If the predicted rankings are consistent with human rankings, we can conclude that the automatic evaluation measures are valid.

4.1.2 Validating automatic evaluation measures

Since it is expensive to use human judges to rate simulated dialogs, we are interested in building prediction models of human judgments using automatic measures[12]. If the prediction model can reliably mimic human judgments, it can be used to rate new simulation models without collecting human ratings. Here, we use a subset of the domain-dependent automatic measures proposed in (Schatzmann et al., 2005a)[13] that are applicable to our data to predict human judgments, including the number of student turns (Sturn), the number of tutor turns (Tturn), the number of words per student turn (Swordrate), the number of words per tutor turn (Twordrate), the ratio of system/user words per dialog (WordRatio), and the percentage of correct answers (cRate). The human judgment on each dialog is calculated as the average of the two judges'

[12] Publications: The work presented in this section was published in (Ai and Litman, 2008).
[13] See Table 11 for all Schatzmann's measures.

ratings on the 3-point scale. We focus on predicting human judgments on the dialog level because these ratings represent the overall performance of the student models.

We first use a stepwise multiple linear regression to predict human judges' ratings using the set of automatic measures we listed above. The stepwise procedure automatically selects measures to be included in the model. For example, d_TUR is predicted as 3.65-0.08*WordRati-3.21*Swordrate, with an R-square of 0.12. The prediction models for d_QLT and d_PAT have similar low R-square values of 0.08 and 0.17, respectively. This result is not surprising because we only include the surface level automatic measures in the prediction models. Also, the measures we use here are designed for comparisons between models instead of predictions. Thus, next we build a ranking model to utilize the measures in their comparative manner. There are also studies that evaluate different systems or system components by ranking the quality of their outputs. For example, (Walker et al., 2001) train a ranking model that ranks the outputs of different language generation strategies based on human judges' rankings.

We train three ranking models to mimic human judges' rankings of the real and the simulated student models on the three dialog level questions using RankBoost, a boosting algorithm for ranking ((Freund et al., 2003), (Mairesse et al., 2007)). We briefly explain the algorithm using the same terminologies and equations as in (Mairesse et al., 2007), by building the ranking model for d_TUR as an example. In the training phase, the algorithm takes as input a group of dialogs that are represented by values of the automatic measures and the human judges' ratings on d_TUR. The RankBoost algorithm treats the group of dialogs as ordered pairs (x, y). x and y are two dialog samples and x has a higher human rated score than y. Each dialog x is represented by a set of m indicator functions $h_s(x)(1 \leq s \leq m)$. For example:

$$h_s(x) = \begin{cases} 1 & WordRatio(x) \geq 0.47 \\ 0 & otherwise \end{cases}$$

Here, the threshold of 0.47 is calculated by Rank-Boost. α is a parameter associated with each indicator function. For each dialog, a ranking score is calculated as:

$$F(x) = \sum_s \alpha_s h_s(x) \quad \text{------ (*)}$$

In the training phase, the human ratings are used to set α by minimizing the loss function:

$$LOSS = \frac{1}{|T|} \sum_{(x,y) \in T} eval(F(x) \leq F(y))$$

The *eval* function returns 0 if (x, y) pair is ranked correctly, and 1 otherwise. In other words, LOSS score is the percentage of misordered pairs where the order of the predicted scores disagrees with the order indicated by human judges. In the testing phase, the ranking score for every dialog is calculated by Equation (*). A baseline model which ranks dialog pairs randomly produces a LOSS of 0.5 (lower is better). While LOSS indicates how many pairs of dialogs are ranked correctly, our main focus here is to rank the performance of the four student models instead of individual dialogs. Therefore, we propose another Averaged Model Ranking (AMR) score. AMR is computed as the sum of the ratings of all the dialogs generated by one model averaged by the number of the dialogs. The four student models are then ranked based on their AMR scores. The chance to get the right ranking order of the four student models by random guess is 1/(4!).

Table 20. A Made-up Example of the Ranking Model

Dialog	Human-rated Score	Predicted Score
real_1	0.9	0.9
real_2	0.6	0.4
ran_1	0.4	0.6
ran_2	0.2	0.2

Table 20 shows a made-up example to illustrate the two measures. Real_1 and real_2 are two dialogs generated by the real student; ran_1 and ran_2 are two dialogs by the random model. The second and third column shows the human-rated score as the gold standard and the machine-predicted score in the testing phase respectively. The LOSS in this example is 1/6, because only the pair of real_2 and ran_1 is misordered out of all the 6 possible pair combinations. We then compute the AMR of the two models. According to human-rated scores, the real model is scored 0.75 (=(0.9+0.6)/2) while the random model is scored 0.3. When looking at the predicted scores, the real model is scored 0.65, which is also higher than the random model with a score of 0.4. We thus conclude that the ranking model ranks the two student models correctly according to the overall rating measure. We use both LOSS and AMR to evaluate the ranking models.

First, we use regular 4-fold cross validation where we randomly hold out 25% of the data for testing and train on the remaining 75% of the data for 4 rounds. Both the training and the testing data consist of dialogs equally distributed among the four student models. However, since the practical usage of the ranking model is to rank a new model against several old models without collecting additional human ratings, we further test the algorithm by repeating the 4 rounds of testing while taking turns to hold out the dialogs from one model in the training data, assuming that model is the new model that we do not have human ratings to train on. The testing corpus still consists of dialogs from all four models. We call this approach the minus-one-model cross validation.

Table 21. LOSS scores for Regular and Minus-one-model

Cross Validation	d_TUR	d_QLT	d_PAT
Regular	0.176	0.155	0.151
Minus-one-model	0.224	0.180	0.178

Table 22. AMR Scores for Regular and Minus-One Cross Validation

	REAL			CLU			COR			RAN		
	H	R	M	H	R	M	H	R	M	H	R	M
d_TUR	0.68	0.62	0.61	0.65	0.59	0.58	0.63	0.52	0.53	0.51	0.49	0.40
d_QLT	0.75	0.71	0.68	0.71	0.63	0.62	0.69	0.61	0.59	0.48	0.50	0.43
d_PAT	0.66	0.65	0.62	0.60	0.60	0.58	0.58	0.57	0.57	0.31	0.32	0.36

Table 21 shows the LOSS scores for both cross validations. We compute a random baseline by randomly assigning orders for each dialog pairs, and then compute the LOSS score for all dialog pairs. Using 2-tailed t-tests, we observe that the ranking models significantly outperform a random baseline in all cases after Bonferroni correction ($p<0.05$). When comparing the two cross validation results for the same question, we see more LOSS in the more difficult minus-one-model case. However, the LOSS scores do not offer a direct conclusion on whether the ranking model ranks the four student models correctly or not. To address this question, we use AMR scores to re-calculate all cross validation results. Table 22 shows the human-rated scores (in column **H**), predicted AMR scores on the regular cross validation (in column **R**), and predicted AMR scores on minus-one cross validation (in column **M**). We see that the ranking model gives the same rankings of the student models as the human judges on all questions in both regular cross validations and minus-one cross validations. Therefore, we suggest that the ranking model can be used to evaluate a new simulation model by ranking it against several old models.

To summarize, we validate the previously proposed domain-independent simulation evaluation measures by showing that these measures can predict the ranking of our user simulations the same as human judges. We also observe that the ranking model built by these automatic evaluation measures can be used to assess the quality of a new simulation model on which we do not have human judgments on.

4.1.3 The deficiency of the task-independent evaluation measures

In Section 4.1.2, we show that the domain-independent evaluation measures we adapted from previous studies can be used to evaluate user simulations by ranking the simulations from the most human-like to the least. However, we are not sure about the differentiating power of these evaluation measures. In other words, if a dialog corpus is shown to be different from a real user corpus by these evaluation measures, can we conclude that this dialog corpus is not realistic? To answer this question, we use these evaluation measures to compare the three human user corpora (f03, s05pre, and s05syn) we collected with the ITSPOKE system (3.1)[14]. Remember that the ITSPOKE system used in the f03 experiment used a synthesized voice. Two versions of the ITSPOKE system were used in the s05 experiments, one with a synthesized voice (used to collect the s05syn corpus) and one with a pre-recorded voice (used to collect the s05pre corpus). The experiments in f03 and s05 use two different groups of users.

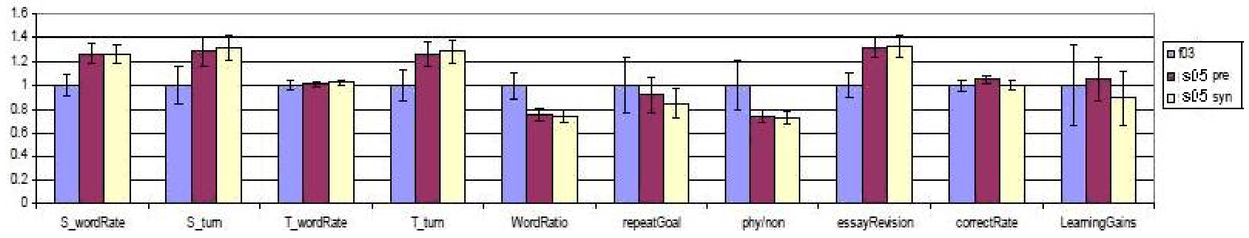


Figure 3. Comparisons between real corpora

Figure 3 illustrates the mean values of each evaluation measure for each corpus. The error bars show standard deviations of the mean values. In the graph, the x-axis shows the

[14] Publications: The work presented in this section was published in (Ai and Litman, 2006).

evaluation measures, the y-axis shows the mean for each corpus normalized to the f03 mean. For instance, when comparing S_wordRate, the mean of S_wordRate for f03 is scaled to “1”, and the means for the 2005 corpora are normalized accordingly. We can tell how different two corpora are from the overlapping between the error bars. The less overlapping are the error bars, the greater is the difference between the two corpora. We can see by studying the first eight groups of bars that the two corpora s05syn and s05pre from 2005 are very different from f03 using most of the measures. The fact that there are no clear differences between s05syn and s05pre suggests that the only difference between the two s05 systems – the type of the system’s voice does not cause systematical differences. Therefore, we consider that the f03 system and s05 systems are similar since they only differ in the type of the system’s voice which does not cause systematic differences in our experiment. Given that the system is the same and both the 05 and 03 corpora are collected with human users, the clear differences between the 2003 and 2005 corpora are most likely due to the different population of subjects. In other words, the differences caught by the above measures may be due to the different subject populations represented in the corpora, instead of the differences in how human-like the user behaviors are. As a result, the differences shown by using the above measures are not sufficient to support conclusions about whether a dialog corpus is collected with simulated users or with human users. If the human corpus used to train the simulation model represents the entire target user population for a dialog system, also a successful simulated corpus represents the same population; the differences shown by the above measures might then be interpreted as the differences in how human-like the user behaviors are. However, if the human data is skewed and only represent a small part of the entire population while a successful simulation represents the right target user population, these two corpora might be shown to be very different using the above measures. Nevertheless, these differences do not indicate that the simulated corpus does not present human-like user behaviors. Interestingly, we do not see clear differences between any two of the three corpora using learning gain features (shown by the last two groups of bars). This could be a positive sign that different groups of students do learn from the interaction with the tutoring system. However, as human learning is a complex behavior which may not be fully described with learning gains alone, we need to verify this in the future work.

To briefly summarize, we find that two human user corpora can be shown to be very different using the domain-independent evaluation measures shown in Table 11. Therefore, if we seen any difference between a dialog corpus and a human user corpus using those evaluation measures, we cannot conclude that the dialog corpus is not realistic. In Section 4.2, we explore a domain-dependent feature that can be used in tutorial dialogs to distinguish realistic versus non-realistic user behaviors.

4.2 THE KNOWLEDGE CONSISTENCY CONSTRAINT

Most studies we reviewed in Section 2.1 (e.g., (Georgila et al., 2006), (Rieser et al., 2006), (Schatzmann et al., 2007a)) emphasize consistency to be one important constraint for user simulations to generate realistic user behaviors. “User goal” is often defined in these user simulations to ensure that simulated users behave in a consistent and goal-directed manner. A commonality of these simulations is that they are built for information-providing dialog systems which help users to complete certain tasks requested by the users. It is natural for these simulations to define a user goal because the goal can be viewed as an abstraction of the task. For example, assuming a user calls the system to complete a task of booking a flight ticket, the user goal can then include providing basic information regarding departure/destination cities and constraints on dates. While this goal-directed simulation approach performs well in information-providing dialogs, it is less applicable for other dialog genres where it is harder to define clear user goals. For example, when using the ITSPOKE system, students usually do not have a clear goal of what physics concepts they want to learn. They may have a general goal to learn some physics, but this kind of goal is too general to be helpful in building a simulation model since it cannot help with deciding the next user action in a dialog. Unlike information-providing systems, which collaborate with users to accomplish pre-existing user goals, computer tutors set up some learning goals for students in a tutoring dialog. Although simulated users in tutoring dialogs do not have consistent user goals, they behave consistently on the knowledge level. In this section, we introduce a domain-dependent constraint – the knowledge consistency constraint to capture the consistency of user knowledge in tutorial dialogs. In Section 4.2.1, we define the knowledge consistency constraint. We show that when using this constraint as a measure to differentiate simulated versus human user behaviors, this measure has a greater differentiating power than the domain-independent evaluation measures described in Section 4.1. We also demonstrate how to implement this constraint in the CLU user simulation we briefly introduced in Section 3.1 to better mimic human user behaviors. In Section 4.2.2, we further use the knowledge consistency constraint to compare different human user corpora. We show that this constraint can capture the similarity among human user corpora. In Section 4.2.3, we use the knowledge consistency constraint as a predictive feature to improve the ranking model we introduced in 4.1.2. By adding this domain-dependent feature, the ranking model can rank the quality of user simulations more reliably.

4.2.1 The knowledge consistency constraint in tutorial dialogs

In a tutoring dialog, the computer tutor trains the student to master any fragment of the persistent, domain-specific information that should be used to accomplish tasks. These task domain concepts, principles, and facts are called knowledge components (VanLehn, 2006). During the dialog, the tutor helps the student to construct and apply knowledge components. Eventually, a practice effect should be observed in which the students remove the flaws in their understanding of knowledge components with more practice. Research on learning (Cen et al., 2006) suggests that the learning process proceeds smoothly without sudden gain or loss of knowledge components. In other words, once the student acquires certain knowledge components, his/her performance on similar problems that require that knowledge component will become stable. Based on this theory, we model student knowledge consistency in the CLU model (3.1) by constraining student performance on similar problems that require the same knowledge component[15]. We also use the knowledge consistency constraint as a measure to evaluate how human like a simulation model is.

Knowledge component representation

We use the f03 ITSPOKE corpus which consists of 210 different tutoring questions to develop knowledge components. The author of this document defined knowledge components for this data by manually clustering tutor questions that discuss the same physics concepts together. For example, the author read the two system questions in the dialog shown in Table 23 and judged that both of them talk about Newton’s third law. Thus, both these tutor questions were tagged as “3rdLaw” and were added into a mapping table, as illustrated in Table 24. In this table, the first

[15] Publications: The work presented in this section was published in (Ai and Litman, 2007).

column shows the names of two example knowledge components (KC). *3rdLaw* stands for Newton's third law, and *acceleration* stands for acceleration. The second column shows examples of some of the tutoring questions associated with these knowledge components. 20 knowledge components were created from the 210 tutor questions in our tutoring dialogs. We call these clusters of questions associated with knowledge components the manual clusters.

Table 23. Sample coded dialog excerpt

ITSPOKE1:	Do you recall what Newton's third law says? [3rdLaw]
Student1:	No. [ic]
ITSPOKE2:	Newton's third law says … If you hit the wall harder, is the force of your fist acting on the wall greater or less? [3rdLaw]
Student2:	Greater. [c]
Dialog goes on…	

Table 24. Examples of knowledge components

KC	Tutoring Question
3rdLaw	Do you recall what Newton's third law says?
	If you hit the wall harder, is the force of your fist acting on the wall greater or less?
acceleration	What is the definition of acceleration?
	Acceleration is the rate of change of what quantity

Knowledge component learning curve.

Learning literature (VanLehn, 2006) points out that student behaviors in tutoring dialogs can be visualized using a **learning curve**, which represents that error rate of student answers decreases according to a power function as the amount of practice increases. Here, we follow the standard way adopted by learning researchers (Cen et al., 2006) to plot the learning curves on simulated as well as human user corpus. Later, we measure how well a learning curve plotted on the simulated data can fit with the learning curve plotted on the human user data to evaluate how well the user simulation mimics human user behaviors.

In a learning curve figure, the x-axis stands for the i-th opportunity the student has to practice a certain knowledge component; the y-axis stands for the error rate, which is the percentage of students who failed to use the knowledge correctly. For example, in the dialog shown in Table 23, this student failed to use 3rdLaw at the first opportunity, but was successful at the second opportunity. Assume there is another student who uses 3rdLaw correctly at both of the opportunities. Given these two students, on the learning curve for 3rdLaw, the error rate would then be 50% for the first opportunity and 0% for the second opportunity.

Following the learning curve plotting procedure defined by (Cen et al., 2006), we first compute separate learning curves for each of the 20 knowledge components. Then, we get an overall learning curve (as in Figure 4) by computing the average error rates of all the knowledge components for each practice opportunity. When using all data from the 20 students, we did not observe a decreasing power curve. However, when we split the 20 students into 10 high learners and 10 low learners according to the median of the normalized learning gain, we observe a decreasing power learning curve from the high learner data which is shown in Figure 4. The equation of this curve is $ErrorRate = 0.409 * i^{th}Opportunity^{(-1.50)}$. The adjusted R^2 value is 0.631, which represents that 63% of the variance in the data is explained by the curve. We do not see a learning curve when using the low learner data. This is not surprising since previous research also reports that learning occurs differently among high/low learners (VanLehn, 2006). Therefore, we confirm that in our ITSPOKE data from real users, the performance of high-learners can be represented by a learning curve in terms of our representation of knowledge components. Since the focus of this study is to model student behaviors while observable learning is taking place, we train the simulation models on the high-learner data only and compare the simulation models only with the high-learners.

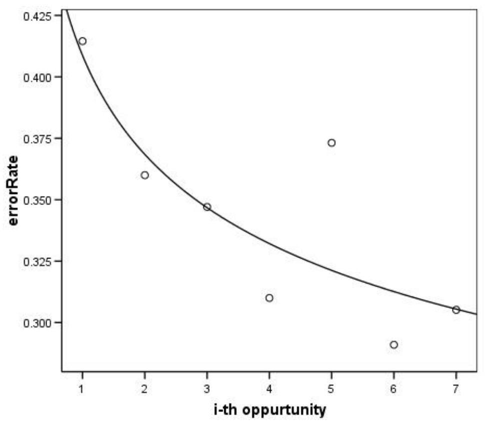


Figure 4. Learning curve for high learners

Using the knowledge consistency constraint to build a user simulation

We briefly mentioned in Section 3.1 that the CLU model is designed to solve the problem of inconsistent user behaviors that may happen with the COR model. The CLU model generates a student answer based on both the content of the tutor's question and the student's previous answer to a similar question. The answer selection probability can be represented as: P(A|KC, c). A stands for the simulated student's action of picking a correct/incorrect answer from the candidate answer set; KC stands for the knowledge component of the tutor's question; and c stands for the correctness of the student's answer to the last previous question that requires the same knowledge component. When there is no previous student answer, the answer selection probability is computed as P(A|KC). In general, this model assumes that a student will have a higher chance to give a correct answer to the question of a cluster in which he mostly answers correctly before, and a lower chance to do so otherwise. Besides the manual clusters we described before, we also automatically cluster the tutor questions into 20 clusters based on the lexical items of the questions in order to investigate how well machine clustering can replace the manual clustering. We use the RBR clustering algorithm provided by CLUTO (Karypis, 2002).

We use automatic clusters to refer to these clusters created by machine, and **auto_Clu** to refer to the Cluster simulation model based on the automatic knowledge component clusters. We use **man_Clu** to refer to the Cluster model based on the manual clusters.

Using the knowledge consistency constraint to evaluate user simulation

Since we believe student knowledge consistency is an important feature to characterize realistic user behaviors and learning curves can visualize this feature, we plot the learning curves for the simulated corpora to compare with the curve observed in the real corpus. If the simulation can model the student learning exactly as what we observed in real data given the knowledge components we defined, the simulated learning curve should mirror the observed learning curve. We use the R^2 to measure the goodness-of-fit between each simulated curve and the real curve. We also report the adjusted R^2 since it is believed to be more accurate for allowing the degrees of freedom to be associated.

We compare the CLU models which feature the knowledge consistency constraint and the COR model which does not feature this constraint. We compare using both the domain-independent measures (4.1) and the knowledge consistency constraint. We let the CLU models and the COR model interact with the ITSPOKE system, generating 500 dialogs for each model. This provides us with simulated corpora of comparable size to previous studies (Schatzmann et al., 2005a) which also compare simulated and real corpora. We first evaluate with respect to the previously used domain-independent measures. In Figure 5, the x-axis shows the evaluation measures; the y-axis shows the mean for each corpus normalized to the mean of the real corpus. The error bars show standard deviations of the mean values. We can see that all the models do not differ from the real students on all the measures, which suggests that they can all simulate realistic high-level dialog behaviors.

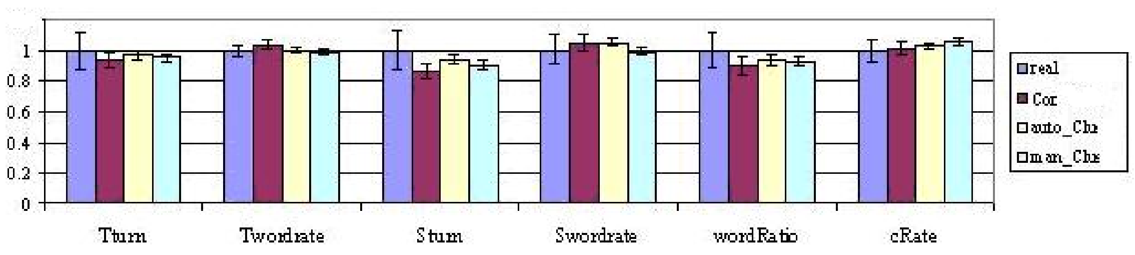


Figure 5. Evaluation of real and simulated dialogs measured by high-level dialog features

Then, we evaluate the simulation models with respect to our new knowledge consistency constraint. Table 25 shows the R^2 and adjusted R^2 value of the simulated curves. The first row lists the name of the models. A higher R^2 or adjusted R^2 implies that the simulation models student knowledge consistency more similar to human users. We can see that both the man_Clu model and the auto_Clu model outperform the correctness model. When using automatic clusters, the adjusted R^2 performance of the Cluster model decreases relatively 53.2%.

Table 25. The goodness-of-fit of simulated and observed learning curves measured by R^2

Model	cor	auto_clu	man_clu
R^2	0.252	0.352	0.564
adjusted R^2	0.102	0.223	0.477

In summary, although the COR model and the two CLU models using different knowledge components perform equally well when measured by high-level dialog features, they show quite different abilities in modeling user knowledge consistency. This implies that the knowledge consistency constraint can be used as a better measure than the previously used domain-independent measures to distinguish different simulation models on the tutoring domain. However, using the knowledge consistency constraint to evaluate user simulation behaviors requires extra effort to construct reasonable knowledge components and plot the learning curve. Constructing knowledge components is not a trivial task. As we find in our study, learning curves may not always be observed even given manually constructed knowledge components. In the future, we will investigate how to construct the appropriate knowledge components in order to apply the knowledge consistency constraint in evaluating user simulation. In this dissertation, when using the knowledge consistency constraint as a simulation evaluation measure, we only consider the high learner data where learning curves can be observed. As a result, when applying the knowledge consistency constraint to compare human user corpora in Section 4.2.2, we only consider high learner data. However, when using knowledge consistency constraint as a feature in building the CLU simulation, we can train the simulation from both high and low learner data. Therefore, the CLU simulations we use in Chapter 4 and in Chapter 5 are all trained from the whole human user corpus.

4.2.2 Using knowledge consistency to distinguish real user corpora

Remember in Section 4.1.3 we showed that human user corpora collected with different user populations were significantly different when comparing using the domain-independent measures we introduce in Table 11. However, we hypothesize that there should be similarities among real user behaviors that distinguish the real users from the simulated users since in the human assessment study (4.1.1) the human judges can agree on human user behaviors much easier than on the simulated user behaviors (Table 15). Here, we use the knowledge consistency constraint to plot the learning curves from real user behaviors generated from ITSPOKE f03 and s05 corpora[16]. We apply the knowledge components we developed on the f03 data to high learners in the s05 data. We observe that a learning curve can be fitted on this subset of the s05 corpus using these knowledge components (R^2=0.562, adjusted R^2=0.457). We further compute

[16] Since users from s05pre corpus and s05syn corpus behave similarly (see Section 4.1.3), we collapse the two corpora into one s05 corpus here.

the goodness-of-fit of the f03 and the s05 learning curves and get an R^2 of 0.583 (adjusted R^2 =0.504). Comparing with the best results in Table 25 (Column “man_clu”) we see that the learning curve plotted on s05 corpus fit better with the curve on f03 corpus than the best curve plotted on the simulated corpus does. Therefore, we hypothesize that the knowledge consistency constraint is a better measure than the domain-independent evaluation measures in capturing the similarity among real users while distinguishing real users from simulated users. However, this preliminary result needs to be confirmed in the future work using more human and simulated corpora.

4.2.3 Using knowledge consistency to improve the prediction model of simulation rankings

Although the original ranking model in Section 4.1.2 is already able to give correct rankings of the four student models, further improvements can ensure the ranking model to be more reliable when being applied to rank more student models. Thus, in this section we add the knowledge consistency as a predictive feature into the original ranking model to improve its performance. Since the knowledge consistency constraint we used in this section is computed on all students over the whole corpus, it is hard to apply this constraint directly as a dialog level feature in the ranking model. Therefore, we will define a simplified feature to represent the knowledge consistency constraint: the value of this feature is one if the user answers the tutor’s question incorrectly while the user was able to answer the tutor’s question correctly the last time; the value of the feature is zero in all other cases. We use the same training-testing data and the same experimental settings as in Section 4.1.2 to test the improved ranking model. Table 26 shows the LOSS scores from the regular and the minus-one-model cross-validation results. Under each dialog level question, we show the LOSS scores for the old model and the new model "addKC" using two cross validations. Recall the LOSS score indicates the percentage of dialog pairs that are mis-ranked by the ranking model with regard to the human gold standard (4.1.1). We can see that by adding this simple feature which represents knowledge consistency, the new ranking model can predict the ranking of the four simulation models more robustly by reducing LOSS scores.

Table 26. Ranking model performance using the knowledge consistency feature

	d_TUR		d_QLT		d_PAT	
crossValidation	old model	addKC	old model	addKC	old model	addKC
regular	0.176	0.094	0.155	0.088	0.151	0.089
minus-one-model	0.224	0.112	0.180	0.098	0.178	0.097

To summarize, the knowledge consistency constraint can be used as a useful feature to capture the similarity among human user corpora as well as to better distinguish human user corpora from simulated user corpora. This constraint can also be used to predict the rankings of several user simulations more reliably. Although this constraint is proposed for our tutoring dialog domain, it can be generalized to other types of dialogs where user knowledge is gradually modified during dialog system interactions, such as trouble shooting dialogs (Janarthanam and Lemon 2008) or collaborative task dialogs (Foster et al., 2009).

5.0 USER SIMUALTION FOR TWO DIALOG SYSTEM DEVELOPMENT TASKS

In Chapter 4, we assess the quality of user simulations by comparing simulated user behaviors to human user behaviors. We assume that a user simulation which can mimic human-like behaviors is better. However, remember the initial goal of constructing user simulations in our research is not to simply build some computer programs that can mimic human user behaviors. The goal is to use user simulations to help dialog system development. Therefore, in this section we look into how useful user simulations are when being used in two dialog system development tasks: generating a training corpus for applying Markov Decision Processes to learn new dialog strategies, and testing dialog system performance.

In Section 5.1, we look into using user simulations for dialog strategy learning. We introduce the experimental setups of our dialog strategy learning task in Section 5.1.1. We use similar setups in all experiments we conduct in Section 5.1. In Section 5.1.2, we first confirm that user simulations are indeed needed for the dialog strategy learning task by showing that a dialog strategy learned from a simulated corpus is better than one trained from a human user corpus on the same dialog strategy learning task. Then, we further look into what type of user simulation is better for the dialog strategy learning task by constructing and comparing different user simulations. We explore different probabilistic user simulation models (5.1.3) and different ways to set up user action probabilities in these models (5.1.4). In Section 5.2, we look into using user simulations for dialog system testing. We first discuss what type of user simulation model is desired to test dialog system performance (5.2.1) by considering which simulation can provide objective measures of dialog system performance similar to human users. However, since it is also important for the simulated users to provide subjective measures of dialog system performance as human users do, we predict human user satisfaction scores using user simulation logs in Section 5.2.2. We discuss how to set up user action probabilities for the dialog system testing task in Section 5.2.3.

5.1 USER SIMULATION FOR DIALOG STRATEGY LEARNING

In this section, we address the issue of using user simulations for dialog strategy learning. Previous studies have examined different machine learning techniques in learning new dialog strategies automatically as reviewed in Section 2.3. In this section, we focus on using user simulations for applying Markov Decision Processes (MDP) to learn a new dialog strategy for the ITSPOKE system. Figure 6 shows using a user simulation in the dialog strategy learning process. All experiments conducted in this section use similar MDP settings for the same dialog strategy learning task. We explain the main experimental setups in Section 5.1.1. Differences in individual experiments will be specified as we reach the details in each subsection. In Section 5.1.2 we show that a large simulated user corpus is needed for dialog strategy learning using MDP. This result justifies our efforts in all following sections on investigating using user simulations for dialog strategy learning. Then, we discuss the two factors of constructing probabilistic user simulations in the context of the dialog strategy learning task: the choice of user simulation model (5.1.3) and the approach to set up user action probabilities in the simulation model (5.1.4).

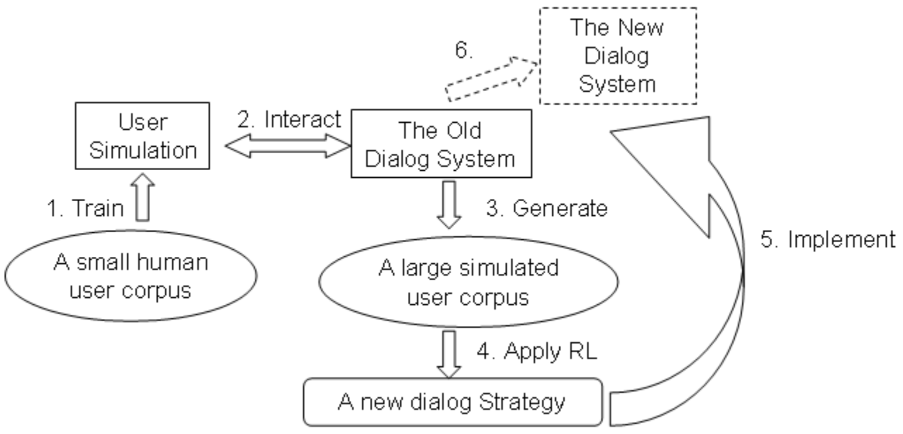


Figure 6. User simulation for dialog strategy learning

5.1.1 The dialog strategy learning task for ITSPOKE

In this section, we introduce our task of learning a new dialog strategy to handle student certainty in the ITSPOKE system (introduced in 3.1). We assess the quality of different user simulations by comparing the qualities of the dialog strategies learned from each simulated corpus. A user simulation is considered better if the dialog strategy trained from the simulated corpus generated by that user simulation is better.

The current ITSPOKE system[17] can only respond to the correctness of a student's utterances; the system thus ignores other underlying information, for example certainty, which is believed to provide useful information for the tutor. In a previous study (Forbes-Riley and Litman, 2005), each student utterance was manually annotated as certain, uncertain, neutral, and mixed based on both lexical and prosodic information. In this study, we use a two way classification of certainty: certain (*cert*) and not-certain (*ncert*), where we collapse uncertain,

[17] Here we refer to the ITSPOKE system that was available at the time of our study in 2007. Now, a new version of the system that handles uncertainty is available (Forbes-Riley et al., 2008).

neutral, and mixed to be ncert to balance our data. In addition, each student utterance is automatically judged as correct (c) and incorrect (ic) by the system and kept in the system's logs. Percent incorrectness (ic%) is also automatically calculated and logged. Remember in Section 4.2.1, we manually clustered tutor questions into 20 clusters. Therefore, each tutor utterance is associated with a cluster (e.g., 3rdLaw). An example coded dialog is shown in Table 27.

Remember that our user simulations work on the word level by using the student answers in the human corpus as the candidate answers for the simulated students (Section 3.1). Here, we simulate student certainty in a very simple way: the simulation models output the certainty originally coded with that utterance.

In the s05pre corpus we collected with the ITSPOKE system (described in Section 3.1), the strength of the tutor's minimal feedback (defined below) is strongly correlated with the percentage of student certainty (chi-square test, $p<0.01$). Strong Feedback (SF) is when the tutor clearly states whether the student's answer is correct or incorrect (i.e., "This is great!"); Weak Feedback (WF) is when the tutor does not comment on the correctness of a student's answer or gives slightly negative feedback such as "well". Therefore, we want to develop a new dialog strategy which manipulates the strength of the tutor's minimal feedback in order to maximize student's overall certainty in the entire dialog. We keep the other parts of the tutor feedback (e.g. explanations, questions) so the system's original design of maximizing the percentage of student correct answers is utilized. A sample coded dialog is shown in Table 27.

Table 27. Sample Coded Dialog

ITSPOKE1:	Do you recall what Newton's third law says? [**3rdLaw**]
Student1:	Force equals mass times acceleration. [**ic, c%=0, ncert**]
ITSPOKE2:	Well, Newton's third law says … If you hit the wall harder, is the force of your fist acting on the wall greater or less? [**3rdLaw, WF**]
Student2:	Greater. [**c, c%=50%, cert**]
	dialog goes on

MDP configuration. Remember in Section 2.3 we reviewed that MDP has four main components: states, actions, a policy, and a reward function. In this study, the actions allowed in each dialog state are **SF** and **WF**; the policy we are trying to learn is in every state whether the tutor should give **SF** or **WF** in order to maximize users' percent certainty in the dialog. In the experiments in Section 5.1.2, 5.1.3, and 5.1.4, we use a simple state space representation (referred as **SSR1**) which is described by the correctness of the current student turn and percent incorrectness so far. The reward function (referred as **RF1**) is to assign +100 to every dialog that has a percent certainty higher than the median from the training corpus, and -100 to every dialog that has a percent certainty below the median. Another state space representation **SSR2** and another reward function **RF2** are introduced in the experiments in Section 5.1.3 to explore the influence of different MDP configurations on the quality of learned dialog strategy. Other MDP parameter settings are the same as described in (Tetreault et al., 2006).

Evaluating learned strategies

We learn dialog strategies from the simulated dialog corpora. In our experiments, we run a user simulation with the dialog system 8,000[18] times to simulate 8000 simulated users, each of which complete 5 dialogs with the system to generate a simulated corpus of 40,000 dialogs in total. We use the dialog strategy learned from a simulated corpus to represent the quality of the simulated corpus. There are different ways to evaluate the learned new dialog strategy. One way is to implement the learned strategy into the original system and then test the effectiveness of the new system in maximizing student certainty. In our study, we introduce an evaluation measure (referred as **EM1**) to evaluate the new dialog strategy by counting the number of dialogs that would be assigned +100 according to RF1. A policy is considered better if it increases the

[18] We empirically discovered that the dialog strategies learned from this size (8,000*5=40,000 dialogs) of simulated corpora is stable.

number of dialogs that will be assigned +100. Similar to previous studies (e.g., (Schatzmann et al., 2007b), (Lemon et al., 2006)), we test the new dialog system with a user simulation that can generate similar behaviors as human users, i.e., the CLU model, since it is the most human-like user simulation (shown in Section 4.1) we built for the ITSPOKE system. In our experiments, we simulate a group of 100 CLU simulated users, each of which interacts with the ITSPOKE system to generate 5 dialogs, to create an evaluation corpus that is of comparable size to (Schatzmann et al., 2005b). The baseline of EM1 is 250, since half of the 500 dialogs will be assigned +100 using a median split.

In the experiments in Section 5.1.2 and 5.1.3, we implement the learned dialog strategies into the original dialog system to evaluate the learned strategies using EM1. Since we used another reward function RF2 in Section 5.1.3, we also introduce a corresponding evaluation measure EM2 in that section. However, since first implementing a new dialog strategy and then testing it can be a complicated process, (Williams and Young, 2007b) use Expected Cumulated Reward (ECR) as an estimation of the quality of the dialog strategy learned by reinforcement learning (in our case MDP). (Tetreault et al., 2007) further introduce an approach to construct the confidence interval for ECR so that we get a sense of how reliable the learned dialog strategy is. We use the ECR to evaluate learned dialog strategies in 5.1.4.

Previous studies (e.g., (Paek, 2006)) have pointed out that the MDP configuration has a strong impact on the quality of learned dialog strategies. Therefore, the quality of the learned dialog strategies does not totally depend on the quality of the user simulations, but also on the MDP configuration. Since our focus here is to compare the effectiveness of different simulated corpora in the dialog strategy learning task, ideally we would like to factor out the impact of MDP configuration. As we already mentioned, in Section 5.1.3, we explore the impact of different MDP configuration by experimenting with different state space representations and different reward functions. In Section 5.1.4, we apply another approach to factor out the impact of MDP configuration. We introduce an evaluation measure which compares the simulated corpora directly by calculating the transitional probabilities that are represented in the corpora. The transitional probability distribution in a user corpus has a direct impact on the quality of the dialog strategy trained on the corpus. Therefore, we can expect to see differences in learned dialog strategies trained from user corpora with different transitional probability distributions, although the difference we observe does not help us to figure out which strategy is better and which is worse.

Table 28 summarizes the MDP configurations and evaluation measures for experiments in Section 5.1 as we explained above.

Table 28. Summary of Experimental Configrations

Configuration \ Experiments		Section 5.1.2	Section 5.1.3	Section 5.1.4
MDP Configuration	action	SF/WF	SF/WF	SF/WF
	state space representation	SR1	SR1, SR2	SR1
	reward function	RF1	RF1, RF2	RF1
Evaluation Measure		EM1	EM1, EM2	ECR, transitional probability

5.1.2 The need of using user simulation for dialog strategy learning

Before we get into the discussion of how to choose user simulations for dialog strategy learning, we first conduct an experiment to show that constructing user simulations and generating a large simulated user corpus are necessary for learning dialog strategies using reinforcement learning. This experiment aims to justify our efforts on investigating using user simulations in dialog strategy learning in later studies.

Training dialog strategies using reinforcement learning is more effective when using a large training corpus since the reinforcement learner needs to explore a vast dialog state space to find the optimal dialog strategy. It is generally believed (e.g., Schatzmann et al., 2006) that the size[19] of the human user corpus that is collected in a standard lab setting (Ai et al., 2007b) is not sufficient to generate reliable dialog strategies using reinforcement learning. Therefore, simulated corpora are used for dialog strategy learning since large amounts of data can be generated easily with user simulation. Nevertheless, although simulated users are built to mimic human user behaviors, state-of-the-art simulation techniques cannot generate fully human-like behaviors as shown in Chapter 4. Therefore, when choosing between using simulated or human user corpus for dialog strategy learning, we face a tradeoff between the amount of the data available and the human-likeness of data.

In this section, we use an empirical approach to investigate the choice of user data for dialog strategy learning using reinforcement learning. Here, we use offline Markov Decision Process (MDP) to learn dialog strategies from three human dialog corpora of different sizes. We compare these learned strategies with dialog strategies learned on a simulated corpus generated by the COR model (defined in Section 3.1). We choose this user simulation from the three user simulations (RAN, COR, and CLU) we built for the ITSPOKE system (see Section 3.1 for more details) because this simulation can generate a corpus more similar to a human user corpus (shown in Table 18) than the RAN model. In other words, the simulated corpus generated by the COR model can be considered as human-like data in large quantity. Here we do not use the CLU

[19] Human user corpora collected in standard lab settings vary from tens to hundreds of dialogs, but are rarely above thousands of dialogs.

model which is even better than the COR model in simulating human-like behaviors because we will test the learned dialog strategies using the CLU model.

We learn dialog strategies directly from the f03 and the s05pre corpus since they represent the sizes of human user corpora that are commonly available in standard lab settings. We use MDP configuration (SSR1+RF1) described in Section 5.1.1 to learn dialog strategies directly from these two human user corpora and also the combined corpus. The new dialog strategies are implemented into new dialog systems and evaluated by EM1 (defined in Section 5.1.1). We use a CLU model trained on the s05pre corpus to interact with each new system to generate 500 dialogs to test each learned strategy.

Table 29. Comparing learned strategies trained from different real and simulated corpora on EM1

Tested on \ Trained on	f03 (100 dialogs)	s05pre (130 dialogs)	s05pre+f03 (230 dialogs)	COR(trained on s05pre) (40,000 dialogs)
CLU trained on s05pre	137	182	152	227

Table 29 summarizes our results. The first row presents different training data (with its size in the parenthesis): using f03, using s05pre, using the two real user data together, and using the data generated from a COR model which is trained from the s05pre corpus. The second row shows the results evaluated by EM1, i.e., the number of dialogs which have a percent student certainty higher than the median student certainty in the original corpus generated by the CLU model and the old ITSPOKE system. We observe that the dialog strategies trained from the simulated corpus significantly ($p<0.05$) outperforms the strategies trained from any of the real user corpora using t-tests with Bonferroni corrections. It’s interesting to note that the quality of learned strategies measured by EM1 does not grow monotonically with the size of the dialog strategy training corpus. We see that the dialog strategy trained from the combined corpus (s05pre+f03) is outperformed by the strategy trained on s05pre only. We believe this is because the dialog strategy trained from the combined corpus is tested on a different group of users than the group of users that it is optimized for. Recall in Section 4.1.3, we find that the s05pre corpus represents a different user population than the f03 corpus.

This result confirms the previous belief that a large training corpus is needed for using reinforcement learning to learn dialog strategies. Since large (in the size of 40,000 dialogs in our setting) real user training corpora is typically not available, it is necessary to build user simulations in order to generate enough training data.

5.1.3 Choosing simulation models in dialog strategy learning

In this section, we examine how the choice of simulation model will impact user simulation performance in dialog strategy learning[20]. When applying reinforcement learning to learn dialog strategies, it is not clear how realistic versus how exploratory (Singh et al., 2002) the training corpus should be. A training corpus needs to be exploratory with respect to the chosen dialog system actions because if a certain action is never tried at certain states, we will not know the value of taking that action in that state. In (Singh et al., 2002), their system is designed to randomly choose one action from the allowed actions with uniform probability in the training phase in order to explore possible dialog state spaces. In contrast, we use user simulations to generate exploratory training data because in the tutoring system we work with, reasonable tutor actions are largely restricted by student performance. If certain student actions do not appear, this system would not be able to explore a state space randomly.

[20] Publications: The work presented in this section was published in (Ai et al., 2007a).

Simulation Models. We compare the performance of four simulation models in the dialog strategy learning task defined in Section 5.1.1, i.e., to generate training data for applying MDP to learn a dialog strategy which maximizes student certainty in the ITSPOKE system. Three of the user simulations we use, the CLU, COR, and RAN models, have been described in Section 3.1. In this study, when the user simulations choose to give a correct (**c**) or incorrect (**ic**) human student answer, these simulation models output the words in the student answer together with the annotated certainty (certain (**cert**) and not certain (**ncert**)) in the original student utterance (see example user utterances in Table 27). The CLU and the COR model represent simulation models which mimic realistic student behaviors in a probabilistic way. In contrast, the RAN model randomly explores all possible dialog states, ignoring the distribution of dialog states observed in human user training data. Here, we propose a new simulation model for the dialog strategy learning task – the Restricted Random Model (RRM) which compromises between exploring dialog state space and generating realistic user behaviors. Given a certain tutor question and a tutor feedback, the RRM model chooses to give a **c** and **cert**, **c** and **ncert**, **ic** and **cert**, or **ic** and **ncert** answer with equal probability. All four simulation models are trained on the s05pre[21] (described in 3.1) corpus.

Methodology. We first let the four simulation models interact with the original system to generate different training corpora. Then, we learn four MDP policies in a fixed configuration (described below) from the four training corpora separately. Since different state space representations and reward functions have a strong impact on the MDP policy learning (Paek, 2006), we investigate different configurations to avoid possible bias introduced by certain

[21] We choose s05pre corpus because it has the largest number of dialogs that are annotated for certainty among the three dialog corpora we collected with the ITSPOKE system (f03, s05pre, and s05syn). We believe that simulated models trained from a larger human user corpus behave more stable.

configurations. In addition to the state space representation **SSR1** we introduced in Section 5.1.1 which represent dialog state space using the correctness of current student turn and percent incorrectness so far, we explore another state space representation **SSR2** which adds in the certainty of the current student turn on top of **SSR1**. We also explore another reward function in addition to the reward function **RF1** we introduced in Section 5.1.1. Remember in **RF1**, we assign +100 to every dialog that has a percent certainty higher than the median of the training corpus, and -100 to every dialog that has a percent certainty below the median. Here in another reward function **RF2**, we assign different rewards to every different dialog by multiplying the percent certainty in that dialog with 100. Other MDP parameter settings remain the same as described in Section 5.1.1.

For each configuration, we run the simulation models until we get enough training data such that the learned policies on that corpus do not change anymore (40,000 dialogs are generated by each model). After that, the learned new policies are implemented into the original system respectively[22]. Finally, we use our most realistic model, the CLU model, to interact with each new system 500 times to evaluate the new systems' performances. We create evaluation corpora that are of comparable size to (Schatzmann et al., 2005b).

In Section 5.1.1, we define an evaluation measure **EM1** which is designed based on **RF1**. **EM1** counts the number of dialogs that would be assigned +100 using the old median split. Here, we add another evaluation measure **EM2** which corresponds to **RF2**. **EM2** computes the average of percent certainty in every single dialog from the newly generated corpus. A policy is considered better if it has more dialogs that will be assigned +100, or can improve the percentage of certainty more than other policies. As we explained in Section 5.1.1, the baseline for **EM1** is

[22] For example, the policy learned from the training corpus generated by the RRM with SSR1 and RF1 is: give SF when the current student answer is ic and ic%>50%, otherwise give WF.

250. The baseline for **EM2** is 33.48%, which is obtained by calculating the percent certainty in the corpus generated by the 40,000 interactions between the CLU and the original system.

Results

Table 30. Evaluation of the new policies trained with the four simulation models

Model	SSR1+RF1		SSR2+RF1		SSR1+RF2		SSR2+RF2	
	EM1	EM2	EM1	EM2	EM1	EM2	EM1	EM2
CLU	240	37.76%	228	37.74%	192	42.67%*	198	41.79%*
COR	227	37.28%	224	37.48%	186	40.78%*	194	40.03%*
RAN	183	34.30%	197	36.52%	175	41.57%*	179	41.45%*
RRM	**413***	**49.31%***	**356***	39.29%*	**303**	42.21%*	**297**	42.21%*

*: results that significantly outperform the corresponding baselines. In bold: results that significantly outperform the other three models.

Table 30 summarizes our results. There are two columns under each "state representation+reward function" configuration, presenting the results using the two evaluation approaches. Although **EM1** measures exactly what **RF1** tries to optimize and **EM2** measures exactly what **RF2** tries to optimize, we show the results evaluated by both **EM1** and **EM2** for all configurations since the two evaluation measures have their own practical values and can be deployed under different design requirements.

All results that significantly[23] outperform the corresponding baselines are marked with *. In these cases, the learned new dialog strategies are found to be significantly more effective than the strategy in the original system. We can also say that the dialog strategy learning tasks are successful in these cases. It is interesting to see that when optimized for RF1, the dialog strategies learned from the RRM simulated corpus outperform the baselines when measured by both EM1 and EM2. However, when optimized for RF2, the dialog strategies learned from the RRM corpus are successful only when measured by the corresponding evaluation measure EM2.

[23] Using 2-sided t-test with Bonferroni correction, $p<0.05$.

When evaluating using **EM1**, the RRM significantly outperforms the other three models in all configurations (in bold in Table 30). Also, both the CLU and the COR models perform better (but not statistically significantly) than the RAN. When evaluating on **EM2**, the RRM significantly outperforms the other three when using **SSR1** and **RF1** (in bold in Table 30). In all other configurations, the four models do not differ significantly. We observe that the RRM outperforms the COR and the CLU models in most of the cases even when we test on the CLU. The two strategies trained from the simulated corpus generated by the two more realistic simulations (CLU and COR) do not differ significantly in any cases. Because we test the learned strategies using the CLU model, we focus on the results of the strategies trained on the COR model to avoid the problem of training and testing using the same simulation. Training dialog strategies from the simulated corpus generated by the COR model and testing using the CLU model[24] also resemble in the real case when we train on a simulated corpus (the less human-like model) while testing on human users.

Discussion

We are interested in why the more realistic user simulations (the CLU model and the COR model) are outperformed by user simulation that is not designed to mimic realistic user behaviors (the RRM model). Since we test the learned dialog strategies with the CLU model, we mainly look into the performance of the COR model. We hypothesize that the performance of the COR model is harmed by the data sparsity issue in the human user corpus that we trained the model on. Consider the case of SSR1: 25.8% of the potentially possible dialog states do not exist in the real corpus. Although we implement a back-off mechanism, the COR model will still have much less chance to transition to the states that are not observed in the real corpus. Thus, when we

[24] The CLU model generates more human-like behaviors than the COR model according to human judges' ranking (Section 4.1.1).

learn the MDP policy from the corpus generated by this model, the actions to take in these less-likely states are not fully learned. In contrast, the RRM model transitions from one state to each of the next possible states with equal probability, which compensates for the data sparsity problem.

To support our hypothesis, next we eliminate the possibility that the RRM model performs well only because that it mimics the low performers. (Lemon and Liu, 2007) show that dialog strategies that are trained on expert user data do not perform well when tested with novice users (the low performers). However, dialog strategies that are trained on novice user data show acceptable performance when tested with expert users, while performing better with novice users. It could be possible that the RRM model outperforms the more realistic model only because it mimics students with low knowledge, since the RRM model answers tutor questions correctly 50% of the time while the COR model give correct answers 60% of the time on average. We will eliminate this possibility by constructing a COR model which gives correct answers 50% of the time. If the RRM model still outperforms the COR model of 50% correctness rate, then the reason that the RRM model outperforms the COR model is not that the RRM model simulates low performers.

In addition, we compare the RRM model with the COR models of different low correctness rates to examine how different correctness rates in the simulation models affect the learned dialog strategies. More specifically, we compare the RRM model with the COR Models with correctness rates of 40%, 30%, and 20%. If the RRM model outperforms all these COR Models, we can conclude that it is more important for the simulated users to explore the possible dialog state spaces more uniformly than to simulate users of low performance.

We observe in Table 30 that in our experimental setting, the state space representation does not impact evaluation results as much as reward functions and evaluation measures. When using RF2 and EM2, the differences among learned dialog strategies become less significant than using RF1 and EM1. Therefore, here we only use the MDP configuration (SSR1+RF1) and evaluate the learned dialog strategies on EM1.

Table 31. Comparing RRM and COR for different correctness rate (c%)

RRM (c%=50%)	COR (c%=60%)	COR (c%=50%)	COR (c%=40%)	COR (c%=30%)	COR (c%=20%)
413	227	215	233	210	202

Table 31 presents the EM1 values of the RRM model and the COR models of different correctness rates. The only significant differences after Bonferroni correction are found between RRM and each of the COR models. We observe that there are 77.9% frequent states[25] in the testing corpus that are seen frequently in the RRM training corpus, but only 68.7% of the frequent states are seen frequently in the COR training corpus with 40% correctness rate (the best performing COR model). This result indicates that the difference between the RRM and the COR corpora which results in different qualities of the learned dialog strategies is not in the student correctness rates that the training corpora represent, but in whether the training corpus contains enough frequent states that are going to be seen in the testing phase.

Our results suggest interesting points in building simulation models. We observe that when trained from a sparse data set, it is better to use a RRM model than a more realistic model (CLU or COR) or a model which randomly explores all user actions (RAN) in a dialog strategy learning task. Since a RRM model is easier to build than the more realistic models, we can keep

[25] We define frequent states to be those that comprise at least 1% of the entire corpus. These frequent states add up to more than 80% of the training/testing corpus. However, deciding the threshold of the frequent states in training/testing is an open question.

the cost of building a user simulation as low as possible. Our results suggest that a realistic user simulation is not always needed for a dialog strategy learning task. Carefully choosing a user simulation model can help to improve the quality of the learned dialog strategies as well as lower engineering costs.

5.1.4 Setting up User Action Probabilities for Dialog Strategy Learning

In Section 5.1.3, we discuss how to choose the type of user simulation models for the dialog strategy learning task. In our experimental setting, the RRM model which generates all possible user actions with the same probability performs the best. As we discussed, our result is likely impacted by the data sparsity issue in the human user data we collected to train our user simulations. In other words, the RRM model may not always be the best performing model in all dialog strategy learning tasks. If another probabilistic model is chosen for the task, how to set up user action probabilities[26] in the simulation model is an important problem to solve[27].

One general approach to set up user action probabilities is to learn the probabilities from a collected human user dialog corpus (e.g., (Schatzmann et al., 2007b), (Georgila et al., 2008)). While this approach takes advantage of observed user behaviors in predicting future user behaviors, it suffers from the problem of learning probabilities from one group of users while potentially using them with another group of users. The accuracy of the learned probabilities becomes more questionable when the collected human corpus is small. However, this is a

[26] Publications: the work presented in this section was published in (Ai and Litman, 2009).
[27] Setting up user action probabilities in the RRM model is easy since the RRM model generates all possible user actions with the same probability.

common problem in building new dialog systems, when often no data[28] or only a small amount of data is available. An alternative approach is to handcraft user action probabilities ((Schatzmann et al., 2007a), (Janarthanam and Lemon, 2008)). This approach is less data-intensive, but requires nontrivial work by domain experts. What is more, as the number of probabilities increases, it is hard even for the experts to set up the probabilities.

Since both handcrafting and training user action probabilities have their own pros and cons, it is an interesting research question to investigate which approach is better for a certain task given the amount of data that is available. In this study, we investigate a manual and a trained approach in setting up user action probabilities, applied to building the same probabilistic simulation model.

Simulation model set-up

In this experiment, we use the CLU model (described in 3.1 and 4.2.1) to explore different approaches in setting up user action probabilities[29] because it is the most complex model we built for the ITSPOKE system. The difficulty of setting up user action probabilities increases when the simulation model gets complex so that there are more probabilities to assign. In a probabilistic user simulation model,

$$P(userAction \mid feature_1, \ldots, feature_n),$$

the number of probabilities involved in this model is:

$$(NumberOfPossibleActions - 1) * \prod_{k=1}^{n} (NumberOfFeatureValues).$$

[28] When no human user data is collected with the dialog system, Wizard-of-Oz experiments can be conducted to collect training data for building user simulations.

[29] User action probabilities are also referred to as correctness rates here since our user simulations' actions are to give correct/incorrect answers.

In the CLU model, there are 40 probabilities to set up[30]. Remember in Section 4.2.1 we discussed that the CLU model simulates student learning using the knowledge consistency constraint. Since different groups of students often have different learning abilities, we examine such differences among our users by grouping the users based on Normalized Learning Gains (NLG) (defined in 3.1), which is an important feature to describe user behaviors in tutoring systems. By dividing our human users into high/low learners based on the median of NLG, we find a significant difference between the NLGs of the two groups based on 2-tailed t-tests ($p<0.05$). Therefore, we construct a simulation to represent low learners and another simulation to represent high learners to better characterize the differences in high/low learners' behaviors. Similar approaches are adopted in other studies in building user simulations for dialog systems (e.g., (Georgila et al., 2008) simulate old versus young users separately). Using separate models to represent high/low learners also makes it easier for handcrafting probabilities in the CLU model as we will describe below.

Next, we discuss how to set up user action probabilities in the CLU Model. We compare learning probabilities from human user data to handcrafting probabilities based on expert knowledge. Since we represent high/low learners using different models, we build simulation models with separate user action probabilities to represent the two groups of learners.

When learning the probabilities in the Trained CLU Models, we calculate user action probabilities for high/low learners in our human corpus separately. We use add-one smoothing to account for user actions that do not appear in the human user corpus. For the first time the student answers a question in a certain cluster, we back-off the user action probability to

[30] The CLU model is defined as $P(A|KC, c)$ in Section 4.2.1. There are 2 possible actions, 20 possible values for the first feature KC and 2 possible values for the second feature c. Therefore, the number of user action probabilities to assign is $40=(2-1)*20*2$.

P(student action | average correctness rate of this question in human user corpus). We first train a CLU model using the data from all 20 human users from the f03 corpus (defined in 3.1.1) to build the TrainedMore (**Tmore**) Model. Then, in order to investigate the impact of the amount of training data on the quality of trained simulations, we randomly pick 5 out of the 10 high learners and 5 out of the 10 low learners to get an even smaller human user corpus. We train the TrainedLess (**Tless**) Model from this small corpus.

When handcrafting the probabilities in the Manual CLU Models, the clusters of questions are first grouped into three difficulty groups (**Easy**, **Medium**, **Hard**). Based on expert knowledge, we assume on average 70% of students can correctly answer the tutor questions from the Easy group, while for the Medium group only 60% and for the Hard group 50%. Then, we assign a correctness rate higher than the average for the high learners and a corresponding correctness rate lower than the average for the low learners. For the Manual CLU model (**Man**), we assume that the high learners answer a question from any tutor clusters which belong to the same difficulty group with the same correctness rate. Similarly, a single correctness rate is assigned to the low learners when answering any questions that belong to the same difficulty group. Since a different human expert will possibly provide a slightly different set of probabilities even based on the same expert knowledge, we build another manual CLU model by randomly assigning correctness rates that varies in a small range next to the average[31]. However, the manual CLU model generated by this naïve approach does not differ much from the Man model in our experiments so we do not report the results on the second manual model here. In

[31] For the clusters in each difficulty group, we randomly assign a correctness rate that differs no more than 5% from the average. For example, for the easy clusters, we assign average probabilities of high/low learners between [65%, 75%].

the future, we will ask different human experts to design different sets of probabilities for the CLU models and compare the models' behaviors.

Although human experts may differ to some extent in assigning individual probability values, we hypothesize that in general a certain amount of expertise is required in assigning these probabilities. To confirm the hypothesis, we build a baseline simulation with no expert knowledge, which is a Random Model[32] (**Ran**) that randomly assigns values for these user action probabilities.

Measures on Dialog Strategy Learning

As we discussed in Section 5.1.1, although first implementing a dialog strategy learned from a simulated corpus into a new dialog system and then testing the effectiveness of the new system is a direct way to assess the usefulness of the user simulation, this process requires a lot of human effort. Alternatively, we can use some automatic measures to estimate the quality of the simulated corpus. The first automatic measure is computed from the simulated corpus directly. When building an MDP from a simulated user corpus, we compute the transition probabilities $P(s_{t+1} | s_t, a)$ (the probability of getting from state s_t to the next state s_{t+1} after taking action a). These transition probabilities are only determined by the states and user actions presented by the training corpus, regardless of the rest of the MDP configuration. Since the MDP configuration has a big impact on the learned dialog strategies, this measure factors out such impact by estimating the differences in learned strategies that are brought in by the training corpora alone.

As a second evaluation measure, we apply MDP on each simulated corpus separately to learn dialog strategies. We compare the Expected Cumulative Rewards (ECRs) (Williams and

[32] This random model is similar to the RAN model we introduced in 3.1, except that this random model uses two sets of probabilities for high/low learners. However, there are no differences in the random behaviors generated by the two simulations.

Young, 2007b) of the learned dialog strategies, which show the expectation of the rewards we can obtain by applying these strategies.

We use the learning task introduced in Section 5.1.1 and SSR1 and RF1 in our MDP configuration. We let all user simulations interact with our dialog system, where each simulates 500 users including 250 low learners and 250 high learners. This creates an evaluation corpus that is of comparable size to (Schatzmann et al., 2005b). Next, we report the results of applying the evaluation measures we discussed above on comparing different simulated user corpora. When we talk about significant results in the statistics tests below, we always mean that the p-value of the test is less than or equal to 0.05.

Table 32. Comparisons of MDP Transition Probabilities at State (c, lc) (Numbers in percentages)

	Tmore	Tless	Man	Ran
s->c_lc	24.82	31.42	25.64	13.25
w->c_lc	17.64	12.35	16.62	9.74
s->ic_lc	2.11	7.07	1.70	19.31
w->ic_lc	1.80	2.17	2.05	21.06
s->c_hc	29.95	26.46	22.23	10.54
w->c_hc	13.93	9.50	22.73	11.29
s->ic_hc	5.52	2.51	4.29	7.13
w->ic_hc	4.24	9.08	4.74	7.68

Remember in SSR1, dialog states are represented by the correctness of the current student turn (c or ic) and a binary value of the student's correctness rate so far (lc or hc). Thus, there are four dialog states in this representation: (c, lc), (c, hc), (ic, lc), and (ic, hc). Table 32 shows the transition probabilities starting from the state (c, lc). For example, the first cell shows in the Tmore corpus, the probability of starting from state (c, lc), getting a strong feedback, and transitioning into the same state is 24.82%. We calculate the same table for the other three states (c, hc), (ic, lc), and (ic, hc). Using paired-sample t-tests with Bonferroni corrections, the only significant differences are observed between the random simulated corpus and each of the other simulated corpora.

We also use a MDP toolkit (Tetreault et al., 2006) to learn dialog strategies from all the simulated corpora and then compute the Expected Cumulative Reward (ECR) for the learned strategies. In Table 33, the upper part of each cell shows the ECR of the learned dialog strategy; the lower part of the cell shows the 95% Confidence Interval (CI) of the ECR. We can see from the overlap of the confidence intervals that the only significant difference is observed between the dialog strategy trained from the random simulated corpus and the strategies trained from each of the other simulated corpora.

Table 33. Comparisons of ECR of Learned Dialog Strategies

	Tmore	Tless	Man	Ran
ECR	15.10	11.72	15.24	7.03
CI	± 2.21	± 1.95	± 2.07	± 2.11

In sum, the manual approach works as well as the trained approach in setting up user action probabilities for the CLU model in our dialog strategy learning task. Although this conclusion is subject to the complexity of the CLU model as well as the human user data we use in the trained approach, similar experiments can be conducted to compare manual and trained approaches for other user simulation models using the automatic measures we introduced.

This experiment takes a first step in comparing the choices of handcrafting versus training user simulations when only limited or even no training data is available, e.g., when constructing a new dialog system. As shown for our task setting, both trained and manual user simulations can be used in generating training data for learning new dialog strategies. However, we observe (as in a prior study by (Schatzmann et al., 2007b)) that the simulation trained from more user data has a better chance to outperform the simulation trained from less training data (the difference is not significant). We also observe that a handcrafted user simulation with expert knowledge can reach the performance of the better trained simulation. However, a certain level of expert knowledge is needed in handcrafting user simulations since a random simulation does not perform well. Therefore, our results suggest that if an expert is available for designing a user simulation when not enough user data is collected, it may be better to handcraft the user simulation than training the simulation from the small amount of human user data. However, it is another open research question to answer how much data is enough for training a user simulation, which depends on many factors such as the complexity of the user simulation model.

5.2 USER SIMULATION FOR DIALOG SYSTEM TESTING

In this section, we address the issue of using user simulation for dialog system testing. As we reviewed in Section 2.4, user simulation has been used to test different components of dialog systems ((López-Cózar et al., 2003), (Schatzman et al., 2007a), (Janarthanam and Lemon, 2008)). Testing dialog systems with user simulations is especially useful in the early development stage, since it would avoid conducting tests with human users when they may feel extremely frustrated due to the malfunction of the unstable systems. In this section, we focus on using user simulations to test dialog strategies. In Section 5.2.1, we argue that a user simulation model which mimics realistic user behaviors is desired for testing dialog strategies. We also show that such user simulation can generate objective measures of dialog system performance similar to human users. However, since subjective measures are also important in testing dialog system performance, in Section 5.2.2 we look into how to generate subjective measures from user simulation logs. Finally, in Section 5.2.3 we investigate how to set up user action probabilities in the user simulation models for the dialog system testing task.

5.2.1 Choose user simulation models for dialog system testing

In Section 5.1.3, we compare several user simulation models which differ in their efforts in mimicking realistic human user behaviors in the task of dialog strategy learning. We show that a good simulated corpus to use in the dialog strategy learning task should balance between simulating realistic user behaviors and exploring possible user actions. However, when using simulated users to replace human users in testing dialog systems, we believe that realistic user behaviors are highly desired because the systems are evaluated and adjusted based on the analysis of the dialogs generated in this phase. Therefore, we would expect that the user behaviors we see at the testing phase are what we will see when the system is used by real users. What is more, over-generated user behaviors may cause the system to be blamed for untargeted functions. For the above reasons, we do not consider the RRM model to be appropriate in testing dialog system performance, although it is the best model in the dialog strategy learning task we examined in Section 5.1.

In Section 4.1.1, we show that the CLU model is ranked the highest among the simulations for the ITSPOKE system in terms of its ability in simulating realistic user behaviors. Therefore when testing the dialog strategies learned for the ITSPOKE system in Section 5.1, we always use the CLU model. In Section 4.2.1 Figure 5, we show that the CLU model can generate dialogs which have similar high-level dialog features comparing to human dialogs. These high-level dialog features are a subset of the objective measures that are extracted from user-system interaction logs to assess dialog system performance when tested with human users (reviewed in

Section 2.4). Note that in Figure 5, we use the high-level dialog features to compare *user behaviors* in simulated and human user corpora while here we use some of those features to evaluate *dialog system performance*. Tturn, Twordrate, Sturn, Swordrate, wordRatio (defined in Table 11) can be used in both evaluating user and dialog system behaviors. However, cRate (defined in Table 11) and knowledge consistency constraint (defined in Section 4.2.1) are features that describe user behaviors. Therefore, those features are not used in evaluating dialog system performance here. Based on Figure 5, we observe when examined by high-level dialog features the CLU model can provide objective measures of dialog system performance similar to human users. Other research also finds that a state-of-the-art probabilistic simulation model can provide objective measures as human users can for dialog system testing (Schatzmann et al., 2007a).

Remember in Section 2.4 we mentioned that there is another type of evaluation measure important in dialog system evaluation with human users – the subjective measures. It is important to show that user simulations can also generate subjective measures (also called user satisfaction scores) like human users before we can use user simulations to replace human users in testing dialog systems. No previous study to our knowledge has addressed this issue. In Section 5.2.2, we conduct an experiment to show that user satisfaction scores can be predicted from user simulation logs.

5.2.2 Predicting user satisfaction scores from user simulation logs

Ideally we would use the CLU model to predict user satisfaction scores on the ITSPOKE system for consistency among our studies[33]. However, as we mentioned at the beginning of Section 3, user satisfaction scores collected with a tutoring system do not always reflect the dialog system performance because a tutoring system is designed to maximize users' learning gain instead of maximizing users' satisfaction. Therefore, in this experiment we use the CHAT system (introduced in Section 3.3) which is a restaurant information system. User satisfaction surveys have been collected with the CHAT system in the final evaluation stage with human users. In this study, we first define three evaluation measures which cover three basic aspects in information-providing dialog systems: understanding ability, efficiency, and the appropriateness of the system actions. Then, we apply these measures on a human user corpus to demonstrate the validity of these measures by constructing a regression model to predict human user satisfaction scores using these evaluation measures. Finally, we apply the regression model on a simulated dialog corpus trained from the above human user corpus. We show that the user satisfaction scores estimated from the simulated corpus do not differ significantly from the human users' satisfaction scores. Thus, we suggest that human users' satisfaction scores can be estimated from the simulated user corpus that trained from the human user corpus.

We use the agenda-based user simulation (Schatzmann et al., 2007a) to predict user satisfaction. This user simulation has shown to be able to generate objective measures of dialog system performance for a tourist information system in (Schatzmann et al., 2007a) and can be easily implemented (described in Section 3.3) for other information providing systems, such as

[33] Publications: the work presented in this section was published in (Ai and Weng, 2008).

CHAT. Next, we define three measures that are automatically retrievable from simulated corpora to build prediction models for user satisfaction scores.

Three Automatic Measures

We describe in detail the three measures we use to predict user satisfaction: the understanding ability measure, the efficiency measure, and the system action appropriateness measure.

Understanding Ability Measure. Human-human dialog is a process to reach mutual understanding between the dialog partners by exchanging information through the dialog. This information exchange process also takes place in the interaction between users and spoken dialog systems. In a task-oriented conversation, the dialog system's major task is to understand the users' needs in order to provide the right service. In the information-state update framework (Wei and Rudnicky, 1999), the system continuously updates its information-states during the dialog while the users are conveying their requests. If a misunderstanding occurs, there would be a mismatch between the users' requests and the system's understandings. Thus, an error recovery dialog is needed to fix the mismatch. The error recovery dialog can be initiated either by the system by asking the user to rephrase or to repeat the previous utterance, or by the user to restate the previous request.

We use the percent of agreement between the system's and the user's understandings (understandingAgreement) to measure how well the system understands the user. The computation of this measure is illustrated through the example dialog in Table 34. In this table, the first column shows the system utterances and the user utterances received by the system. The correct words are shown in square brackets immediately after the misunderstood words (e.g., in Utterance "User1"). The second column represents semantic content from the users' utterances in the form of constraint-value pairs based on the system's understandings. This information can be automatically retrieved from the system logs. The third column includes the action types of the current system/user utterances. Since the dialog manager is an information-updating dialog manager that manages information in the format of slots, this column also shows the number of slots that are exchanged in the utterance and the number of matched slots. In our task domain, the user can request information (req_info), request the same information again (rereq_info), answer an explicit confirmation (exp_confirm), and close a dialog (close_dialog). The system can provide information (pro_info) or explicitly confirms (exp_confirm) the information. Another available system action that is not shown in this example is to ask the user to repeat/rephrase (rephrase), where the user can respond by providing the information again (repro_info).

Table 34. A Sample Dialog with Annotations

Utterances	Constraints Understood by the System	Action Types & Understanding Status
User1: I'm looking for a five star Thai [Italian] restaurant accept credit cards		Action: req_info Slots conveyed: 3 slots
System1: There are 7 restaurants that have a five star rating, accept credit cards, and serve Thai food in my database. Here are the first few: TeJUS, Slam Corner Cate, Setha's Restaurant. Should I continue?	Service level=five star Cuisine type=Thai Payment type=credit card	Action: pro_info Slots accepted: 3 slots Completely matched: 2 slots
User2: I need to look for Italian restaurant		Action: rereq_info Slot conveyed: 1 slot
System2: Are you looking for a restaurant that serves Italian food?	Service level=five star cuisine Cuisine type=Italian (unconfirmed) Payment type=credit card	Action: exp_confirm Slot updated: 1 slot Completely matched: 2.5 slot
User3: yes		Action: exp_confirm Slot conveyed: 1 slot
System3: I found 38 restaurants that serve Italian food, accept credit cards and have a five star rating, for example, Toto Café …	Service level=five star Cuisine type=Italian (confirmed) Payment type=credit card	Action: pro_info Slots accepted: 3 slots Completely matched: 3 slots
User4: I'm done.		Action: close_dialog

In our experiment, we measure the understandings between the users and the system by comparing the values of the constraints that are specified by the users with their values understood by the system. In this dialog, the user specified all constraints in the first utterance:

Service level = Five star
Cuisine type = Italian
Payment type = Credit card

The first system utterance shows that the system understood two constraints but misunderstood the cuisine type, thus the percent agreement of mutual understandings is 2/3 at this time. Then, the user restated the cuisine type and the second system utterance confirmed this information. Since the system only asks for explicit information when its confidence is low, we count the system's understanding on the cuisine type as a 50% match with the user's. Therefore, the total percent agreement is 2.5/3. The user then confirmed that the system had correctly understood all constraints. Therefore, the system provided the restaurant information in the last utterance. The system understands the user 100% at this point.

The percent agreement of system/user understandings over the entire dialog is calculated by averaging the percent agreement after each turn. In this example, understandingAgreement is (2/3 + 2.5/3 + 1)/3 =83.3%. We hypothesize that the higher the understandingAgreement, the better the system performs, and thus the more the user is satisfied. The matches of understandings can be calculated automatically from the user simulation and the system logs. However, since we are building prediction models of user satisfaction scores using human user data first, we manually annotated the semantic contents (e.g., cuisine name) in the real user corpus.

Previous studies (e.g., (Walker et al., 1997)) use a corpus level semantic accuracy measure (semanticAccuracy) to capture the system's understanding ability. SemanticAccuracy is defined in the standard way as the total number of correctly understood constraints divided by the total number of constraints mentioned in the entire dialog. The understandingAgreement measure we introduce here is essentially the averaged per sentence semantic accuracy, which emphasizes the utterance level perception rather than a single corpus level average. The intuition behind our measure is that it is better for the system to always understand something to keep a conversation going than for the system to understand really well sometimes but really bad at other times. We compute both measures in our experiments for comparison.

Efficiency Measure. Efficiency is another important measure to evaluate dialog system performance. A standard efficiency measure is the number of dialog turns. However, we would like to take into account the user's dialog strategy because how the user specifies the restaurant selection constraints has a certain impact on the dialog pace. Comparing two situations where one user specifies the three constraints of selecting a restaurant in three separate utterances, while another user specifies all the constraints in one utterance, we will find that the total number of dialog turns in the second situation is smaller assuming perfect understanding. Thus, we propose to use the ratio between the number of turns in the perfect understanding situation and the number of turns in practice (efficiencyRatio) to measure the system's efficiency. The larger the efficiencyRatio is, the closer the actual number of turns is to the perfect understanding situation. In the example in Table 34, because the user chose to specify all the constraints in one utterance, the dialog length would be 2 turns in perfect understanding situation (excluding the last user turn which is always "I'm done"). However, the actual dialog length is 6 turns. Thus, the efficiencyRatio is 2/6.

Since our task scenarios always contain three constraints, we can calculate the length of the error free dialogs based on the user's strategy. When the user specifies all constraints in the first utterance, the ideal dialog will have only 2 turns; when the user specifies two constraints in one utterance and the other constraints in a separate utterance, the ideal dialog will have 4 turns; when the user specifies all constraints one by one, the ideal dialog will have 6 turns. Thus, in the simulation environment, the length of the ideal dialog can be calculated from the simulated users' agenda. Then, the efficiencyRatio can be calculated automatically. For the human user dialogs, we manually computed this measure.

Similarly, in order to compare the efficiencyRatio we defined here with other efficiency measures defined in previous studies, we also look at the total number of dialog turns (dialogTurns) proposed in (Möller et al., 2007).

Action Appropriateness Measure. This measure aims to evaluate the appropriateness of the system actions. The definition of appropriateness can vary on different tasks and different system design requirements. For example, some systems always ask users to explicitly confirm their utterances due to high security needs. In this case, an explicit confirmation after each user utterance is an appropriate system action. However, in other cases, frequent explicit confirmations may be considered as inappropriate because they may irritate the users. In our task domain, we define the only inappropriate system action to be providing information based on misunderstood user requirements. In this situation, the system is not aware of its misunderstanding error. Instead of conducting an appropriate error recovery dialog, the system provides wrong information to the user which we hypothesize will decrease the user's satisfaction.

We use the percentage of appropriate system actions out of the total number of system actions (percentAppropriate) to measure the appropriateness of system actions. In the example in Table 34, only the first system action is inappropriate in all 3 system actions. Thus, the percent system action appropriateness is 2/3. Since we can detect the system's misunderstanding and the system's action in the simulated dialog environment, this measure can be calculated automatically for the simulated dialogs. For human user dialogs, we manually coded the inappropriate system utterances.

Note that the definition of appropriate action we use here is fairly loose. This is partly due to the simplicity of our task domain and the limited possible system/user actions. Nevertheless, there is also an advantage of the loose definition: we do not bias towards one particular dialog strategy since our goal here is to find some general and easily measurable system performance factors that are correlated with the user satisfaction.

Building prediction model of user satisfaction.

Table 35 lists the correlation between the evaluation measures and the user satisfaction scores, as well as the p-value for each correlation. The previously proposed measures are shown in Italics. The correlation describes a linear relationship between these measures and the user satisfaction scores. For the measures that describe the system's understanding abilities and the measures that describe the system's efficiency, our new measures show higher correlations with the user satisfaction scores than their counterparts. Therefore, in the rest of the study, we drop the two measures used by the previous studies, i.e., semanticAccuracy and dialogTurns.

Table 35. Correlations with User Satisfaction Scores

Evaluation Measures	Correlation	P-value
understandingAgreement	0.354	0.05
semanticAccuracy	*0.304*	*0.08*
efficiencyRatio	0.406	0.02
dialogTurns	*-0.321*	*0.05*
percentAppropriate	0.454	0.01

We observe that the user satisfaction scores are significantly positively correlated with all the three proposed measures. These correlations confirm our expectations: user satisfaction is higher when the system understands better the users' requirements; when the dialog efficiency is closer to the situation of perfect understanding; or when the system's actions are mostly appropriate. We suggest that these measures can serve as indicators for user satisfaction.

We then use the three measures to build a regression model to predict user satisfaction scores. The prediction model is:

User Satisfaction =6.123*percentAppropriate+2.854*efficiencyRatio

+0.864*understandingAgreement- 4.6

The R-square is 0.655 ($p<0.05$), which indicates that 65.5% of the user satisfaction scores can be explained by this model. While this prediction model has much room for improvement, we suggest that it can be used to estimate the users' satisfaction scores for simulated users in the early system testing stage to quickly assess the system's performance.

Estimating User Satisfaction Scores for User Simulation

We train a goal and agenda driven user simulation model (described in Section 3.3) from the final evaluation dialog corpus with the real users. The simulation model interacts with the dialog system 20 times (each time the simulation model represents a different simulated user), generating nine dialogs on all of the nine tasks that the human users completed in the evaluation. The simulated corpus consists of 180 dialogs from 20 simulated users, which is of the same size as the real user corpus. The three evaluation measures we defined above are computed automatically at the end of each simulated dialog. We compute the estimated user satisfaction score using the prediction model built from the human user data for each simulated user. We then compare the user satisfaction scores of the 20 simulated users with the satisfaction scores of the 20 real users. The average and the standard deviation of the user satisfaction scores for real users are (3.79, 0.72), and the ones for simulated users are (3.77, 1.34). Using two-tailed t-test at significance level $p<0.05$, we observe that there are no statistically significant differences between the two pools of scores. Therefore, we suggest that the user satisfaction estimated from the simulated dialog corpus can be used to assess the system performance. However, these average scores only offer us one perspective in comparing the real with the simulated user satisfaction. In the future, we would like to look further into the differences between the distributions of these user satisfaction scores.

In summary, user simulation has been increasingly used in generating large corpora for using machine learning techniques to automate dialog system design. However, user simulation has not been used much in testing dialog systems since we do not get important feedback on user satisfaction when replacing human users with simulated users. In this experiment, we suggest that while the simulated users might not be mature to use in the final system evaluation stage, they can be used in the early testing stages of the system development cycle to make sure that the system is functioning in the desired way. We introduce a set of evaluation measures that can be extracted from simulation logs. We show that user satisfaction scores can be predicted using these evaluation measures by building a regression model. Therefore, we suggest that user simulations can be used to provide both objective measures (5.2.1) and subjective measures of dialog system performance as human users.

5.2.3 Setting up user action probabilities in dialog system testing

In Section 5.1.4 we discussed how to set up user action probabilities in the CLU model for learning dialog strategies in the ITSPOKE system. We compared a simulation (Man) built by a manual approach, two simulations (Tless and Tmore) built by a trained approach, and a random simulation (Ran). Since the CLU model is the best model that we have to test the ITSPOKE system, in this section, we continue to compare the manual and the trained approach in setting up user action probabilities in the CLU model for the dialog system testing task[34].

Comparing Objective Dialog System Performance

First, we compare the Man, Tmore, Tless, and Ran models on generating objective dialog system performance measures. As explained in Section 5.2.1, we use Tturn, Twordrate, Sturn, Swordrate, and wordRatio as the objective measures here. These comparisons give us a direct impression on whether the objective measures of dialog system performance we calculate from the simulated dialogs are similar to the ones we calculate from the human user dialogs. Similar to Section 5.1.4, we let all user simulations interact with the ITSPOKE system, where each simulates 250 low learners and 250 high learners.

Figure 7 shows the comparison results. The x-axis shows the evaluation measures; the y-axis shows the mean for each corpus normalized to the mean of the human user corpus. Error bars show the standard deviations of the mean values. As we can see from the figure, the objective measures of dialog system performance calculated from the Random simulated corpus are different from the measures calculated from the human and all other simulated corpora. There is no difference in objective measures calculated from the human corpus, the trained and the manual simulated corpora.

[34] Publications: the work presented in this section was published in (Ai and Litman, 2009).

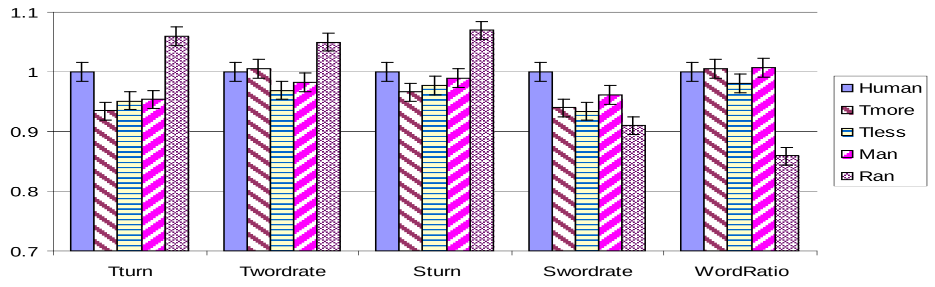


Figure 7. Comparison of Human and Simulated Dialogs by High-level Dialog Features

In sum, both the Trained CLU Models and the Manual CLU Model can provide objective measures of dialog system performance similar to humans while the Random Model cannot. Therefore, both the trained and manual CLU models can be used to generate objective measures of dialog system performance for dialog system testing.

Comparing Utterance-level Simulated User Actions

When we use simulated users to test learned dialog strategies (5.1.3), we mainly compare the testing corpora generated between the simulated users and the dialog systems with different learned strategies on the corpus level. For example, if the new dialog strategy is aimed to improve student certainty in dialogs, when testing the new dialog strategy, we count the overall number of dialogs in which student certainty is improved by using the learned new strategy. In that case, we are interested in whether the average certainty among all students is improved. Thus, we do not need to examine whether a student is certain or not in each student utterance. In Figure 7 we also compare different user simulations' capabilities in generating objective dialog system performance measures on the corpus level. However, there are also cases in which we need to closely watch every single action of a simulated user's. For example, when using a simulated user to test one particular system prompt, we are interested in how the simulated user responds after the prompt. In this case, a simulated user that can accurately predict human user behaviors is needed to replace human users in testing the dialog system. Next, we assess how accurately a user simulation can replicate human user behaviors when encountering the same dialog situation.

We randomly divide the human user dialog corpus into four parts: each part contains a balanced amount of high/low learner data. We perform four-fold cross validations by always using 3 parts of the data as our training corpus for user simulations, and the remaining one part of the data as testing data to compare with simulated user actions. Remember we built separate CLU models to represent high/low learners (5.1.4). When comparing simulated user actions with human user actions, we compare high human learners only with simulation models that represent high learners and low human learners only with simulation models that represent low learners. Comparisons are done on a turn by turn basis. Every time the human user takes an action in the dialogs in the testing data, the user simulations are used to predict an action based on related dialog information from the human user dialog. For a CLU Model, the related dialog information includes the cluster of the current tutor question (KC) and last time whether the student answered a tutor question from the same cluster correctly (c) (4.2.1). We first compare the simulation predicted user actions directly with human user actions. We define simulation accuracy as:
Accuracy = Correctly predicted human user actions / Total number of human user actions.

However, since our simulation model is a probabilistic model, the model will take an action stochastically after the same tutor turn. In other words, we need to take into account the probability for the simulation to predict the right human user action. If the simulation outputs the right action with a small probability, it is less likely that this simulation can correctly predict human user behaviors when generating a large dialog corpus. We consider a simulated action associated with a higher probability to be ranked higher than an action with a lower probability.

Then, we use the reciprocal ranking from information retrieval tasks (Radev et al., 2002) to assess the simulation performance[35]. Mean Reciprocal Ranking is defined as:

$$MRR = \frac{1}{A}\sum_{k=1}^{A}\frac{1}{rank_i}$$

In this equation, A stands for the total number of human user actions, $rank_i$ stands for the ranking of the simulated action which matches the i-th human user action.

Table 36. An Example of Comparing Simulated Actions with Human User Actions

i-th Turn	human	Simulation Model	Simulation Output	CorrectlyPredicted Actions	Reciprocal Ranking
Student1	ic	60% ic, 40% c	ic	1	1
Student2	c	70% ic, 30% c	ic	0	½
Average	/	/	/	(1+0)/2	(1+1/2)/2

Table 36 shows an example of comparing simulated user actions with human user actions in the sample dialog in Table 27. In the first turn Student1, a simulation model has a 60% chance to output an incorrect answer and a 40% chance to output a correct answer while it actually outputs an incorrect answer. In this case, we consider the simulation ranks the actions in the order of: ic, c. Since the human user gives an incorrect answer at this time, the simulated action matches with this human user action and the reciprocal ranking is 1. However, in the turn Student2, the simulation's output does not match the human user action. This time, the correct simulated user action is ranked the second. Therefore, the reciprocal ranking of this simulation action is ½.

Similarly to Section 5.1.4, we generated a simulated corpus of 250 low learners and 250 high learners. Table 37 shows the averages and Confidence Intervals (CI) in parenthesis from the four fold cross validations. The second row shows the results based on direct comparisons with

[35] (Georgila et al., 2008) use Precision and Recall to capture similar information as our accuracy, and Expected Precision and Expected Recall to capture similar information as our reciprocal ranking.

human user actions, and the third row shows the mean reciprocal ranking of simulated actions. We observe that in terms of both the accuracy and the reciprocal ranking, the performance ranking from the highest to the lowest (with significant difference between adjacent ranks) is: the Tmore Model, the Man model, the Tless Model, and the Ran Model. Therefore, we suggest that the handcrafted user simulation is not sufficient to be used in evaluating dialog systems because it does not generate user actions that are similar to human user actions. However, the handcrafted user simulation is still better than a user simulation trained with not enough training data. In sum, the Tmore simulation performs the best in predicting human user actions.

Table 37. Comparisons of Correctly Predicted Human User Actions

	Tmore	Tless	Man	Ran
Accuracy	0.78 (± 0.01)	0.6 (± 0.02)	0.7 (± 0.02)	0.41 (± 0.02)
MRR	0.72 (± 0.02)	0.52 (± 0.02)	0.63 (± 0.02)	0.32 (± 0.02)

In summary, we compare several approaches in setting up user action probabilities for the same simulation model: training from all available human user data, training from half of the available data, a manual approach which utilizes human expertise, and a baseline approach which randomly assigns all user action probabilities. We find that the two trained simulations and the manual simulation outperform the random simulation on all measures. No significant difference is observed among the trained and the handcrafted simulations when comparing their capabilities in generating objective dialog system performance measures on the corpus-level. However, the simulation trained from all available human user data can predict human user actions more accurately than the manual simulation, which again performs better than the model trained from half of the human user corpus. Our results suggest that when using simulations to test a dialog system, it is better to use the data to train a new simulation once we have a reasonable amount of human user data in order to replace the handcrafted simulation. However, deciding the amount of data that is needed to build a realistic user simulation is another open research question to solve.

To sum up, in Chapter 5 we investigate different types of user simulations given two dialog system development tasks, the dialog strategy learning task (5.1) and the dialog system testing task (5.2). We observe that the type of user simulation models (5.1.3 and 5.2.1) and the approach to set up user action probabilities in the models (5.1.4 and 5.2.3) are different based on the tasks. Our results suggest that how to construct user simulation is a task-dependent question. It is important to consider the function of a user simulation in the simulation designing phase.

6.0 OTHER TYPES OF USER SIMULATIONS

In Chapter 5, we demonstrate an empirical framework to compare different probabilistic simulation models and different approaches in setting up user action probabilities in those models given two dialog system development tasks. The user simulations we used in our study are derived from previous work on constructing probabilistic user simulations. Our goal is not to propose a new user simulation that outperforms previous simulations, but to use simulations that can be easily extended across different domains to illustrate important factors to consider in constructing user simulations in a task-dependent context. For this reason, we used the most widely used user simulation, the probabilistic user simulation, and construct the user simulation similar to previous studies, i.e., to train the simulation from human subject corpus. However, in this chapter, we would like to highlight some other approaches in constructing user simulations. In Section 6.1, we summarize some non-probabilistic user simulations that have been used in previous studies. Even for probabilistic user simulations, there are other factors that will impact simulation performance than the two main factors we discussed in Chapter 5. In Section 6.2, we show a study that explores one such factor, i.e., the resources of simulation training data. We suggest that there are many types of probabilistic and non-probabilistic user simulations and many factors to consider in constructing user simulations. How to choose a user simulation type and how to focus on the main factors that impact user simulation performance are task-dependent questions.

6.1 NON-PROBABILISTIC USER SIMULATIONS

When being used in different dialog system development tasks, user simulations are basically built to replace human users. Probabilistic user simulations are widely used for such tasks because they encode observed human user behaviors in a natural way so that the simulated users can be expected to reflect human user behaviors that the user simulations are trained from. However, user simulations can be used for a large variety of tasks in human-computer interaction and probabilistic user simulations are not necessarily always the best choice. For example, (Walker, 2005) use a user simulation environment to test hypotheses on collaborative communication strategies. In that study, two simulated users negotiate an agreement on the furniture layout of a two room house. The study is designed to test distinct parameters in multi-agent conversations, including available processing resources, communicative strategy, and goal to achieve. Since the distinct parameters are indicators of human negotiating strategies instead of estimators of observed human user behaviors, the simulated users are built on heuristic rules which represent different negotiating strategies by assigning different values to the above parameters, instead of being trained on any observed human user corpus. In this way, different negotiating strategies can be easily tested by changing the heuristic rules.

Another type of non-probabilistic user simulation is built by (Matsuda et al., 2005). This simulation is implemented in a cognitive tutor authoring toolkit to help an author build a cognitive tutor without heavy programming. The simulation is used to demonstrate to the author all types of user behaviors that the tutoring system should be able to handle at a certain stage. More specifically, the simulation uses production rules to generate answers to the tutoring system's questions given the knowledge that has been taught so far. The production rules are written to produce both correct and incorrect answers with regard to the tutor's questions so that the simulation can generate a good variety of student behaviors. Since a tutoring system should be able to handle any possible student answers, a probabilistic user simulation which mainly generates observed human user behaviors based on a small human user corpus will not be sufficient for this task.

While probabilistic user simulations utilize observed human user behaviors, they often only produce the average behaviors of human users. (Rieser et al., 2006) use clustering to build simulated users that generate complete, consistent and varying behaviors with respect to human users. Simulated behaviors are defined by a sub-group of human users who share common behaviors instead of being defined by all human users. With such a user simulation, adaptive system strategies can be learned from different group of users.

A user simulation is in essential a conversational agent and thus can be built using similar approaches used in dialog system design. Rule-based user simulations can be viewed as designing by manual efforts while probabilistic user simulations can be viewed as designing by learning from data. However, since most user simulations are used as a tool in dialog system development, we would like to keep user simulation as simple as possible. Therefore, it is important to choose the appropriate user simulation with the lowest engineering cost given the design goal.

6.2 USER SIMULATION TRAINING DATA

In this dissertation, we focus on constructing simple probabilistic user simulations in which the simulated user generates the next user action based on the previous dialog system action according to observed human user behaviors. We discuss two main factors in constructing probabilistic user simulations in Chapter 5. However, there are other factors that may impact user simulations' performance. In this section, we explore the impact of user simulation training data.

In our studies and in many previous studies, probabilistic user simulations are trained from a small corpus collected with paid human subjects (hereafter referred to as subjects). However, these user simulation models intend to mimic the behaviors of real users (hereafter referred to as users). It is not well understood whether and how a corpus of dialogs collected using subjects differs from a corpus of dialogs obtained from users. In this section, we compare a subject corpus and a user corpus collected with the same dialog system[36]. We discuss the impact of both corpora on user simulation behaviors in the dialog strategy learning and dialog system testing tasks.

Selecting a small group of subjects to represent a target population of users can be viewed as statistical sampling from an entire population of users. Thus, (1) a certain amount of data is needed to draw statistically reliable conclusions, and (2) subjects should be randomly chosen from the total population of target users in order to obtain unbiased results. While we believe that most spoken dialog subject experiments have addressed the first point, the second point has been less well addressed. Most academic and many industrial studies recruit subjects from nearby resources, such as college students and colleagues, who are not necessarily representative of the target users of the final system; the cost to employ market survey companies to obtain a better representation of the target user population is usually beyond the budget of most research projects. In addition, because subjects have either volunteered or are compensated to participate in an experiment, their motivation is often different from that of users. In fact, a study comparing spoken dialog data obtained in usability testing versus in real

[36] Publications: the work presented in this section was published in (Ai et al., 2007b).

system usage finds significant differences across conditions (e.g., the proportion of dialogs with repeat requests was much lower during real usage) (Turunen et al., 2006).

In this section, we use Let’s Go Lab (introduced in Section 3.2), a platform for experimenting with a deployed spoken dialog bus information system, to collect a human subject corpus and a real user corpus on the same task for comparisons. Our first corpus is collected by recruiting subjects to call Let’s Go in a standard laboratory setting, while our second corpus consists of calls from real users calling Let’s Go during its operating hours. We quantitatively characterize the two collected corpora using previously proposed measures from the spoken dialog literature, and then discuss the statistically significant similarities and differences between the two corpora with respect to these measures. We further discuss the potential impact of these two training data on the user simulation models trained from them.

Two Experimental Conditions.

To collect our subject corpus we used a “standard” laboratory experiment, following typical practices in the field. We surveyed a set of spoken dialog papers involving human subject experiments (namely, (Allen et al., 1996), (Batliner et al., 2003), (Bohus and Rudnicky, 2006), (Giorgino et al., 2004), (Gruenstein et al., 2006), (Hof et al., 2006), (Lemon et al., 2006), (Litman and Pan, 2002), (Möller et al., 2006), (Rieser et al., 2005), (Roque et al., 2006), (Singh et al., 2000), (Tomko and Rosenfeld, 2006), (Walker et al., 2001a), (Walker et al., 2000)), in order to define a “standard” laboratory setting for use in our own experiments with subjects. We survey the literature from four perspectives: subject recruitment, experimental environment, task design, and experimental policies.

Subject Recruitment. Recruiting subjects involves deciding who to recruit, where to recruit, and how many subjects to recruit. In the studies we surveyed, the number of subjects recruited for each experiment ranged from 10 to 72. Most of the studies recruited only native speakers. Half of the studies clearly stated that the subjects were balanced for gender. Most of the studies recruited either college students or colleagues who were not involved in the project itself. Only one study recruited potential system users by consulting a market research company.

Experimental Environment. Setting up an experimental environment involves deciding where to carry out the experiment, and how to set up this experimental environment. The location of the experiment may impact user performance since people behave differently in different environments. This factor is especially important for spoken dialog systems, since system performance is often impacted by noisy conditions and the quality of the communication channel. Although users may call a telephone-based dialog system from a noisy environment using a poor communication channel (e.g., by using a cell phone to call the system from the street), most experiments have been conducted in a quiet in-room lab setting. Subjects typically talk to the system directly via a high-quality microphone, or call the system using a land-line phone. Among the studies we looked at, only 2 studies had subjects call from outside the lab; another 2 studies used driving simulators. One study changed the furniture arrangement in the lab to simulate home versus office scenarios.

Task Design. Task design involves specifying whether subjects should use the dialog system to accomplish specific tasks, and if so, defining those tasks. All except one study asked subjects to finish a set of fixed tasks in a predefined order. In one study, subjects were asked to perform 2 open tasks after a series of 7 fixed tasks. In another study, where the system provided restaurant information, the researchers asked the subjects to ask about information for at least 4 restaurants, but did not specify the restaurant names. The number of tasks in these studies ranged from 2 to 10.

Experimental Policies. Experimental policies involve specifying additional procedures for running subjects during the course of the experiment. None of the studies mentioned that they controlled their experiments by setting any time limits for the subjects. Only 2 studies clearly declared that subjects were told to read some instructions before the experiment started. While two studies motivated subjects by offering a bonus upon task completion, the majority of studies paid subjects on the basis of their participation alone.

Based on the literature, we summarize that a standard way to carry out human subject experiments with spoken dialog systems (here "standard" means that the practice occurred in a majority of the papers surveyed), is as follows: (1) Recruit at least 10 subjects who are college students or colleagues who are native English speakers, trying to balance between genders; (2) Ask the subjects to come to the lab to generate their dialogs with the system; (3) Set up several tasks for the subjects, and ask them to complete these tasks in a certain order; (4) Pay the subjects for their participation, without a bonus.

We collected our subject data following these criteria. We recruited 39 subjects (19 female and 20 male) from the University of Pittsburgh who were native speakers of American English. We asked the subjects to come into our lab to call the system from a land-line phone. We designed 3 task scenarios and asked the subjects to complete them in a given sequence. Each task included a departure place, a destination, and a time restriction (e.g., going from the University of Pittsburgh to Downtown, arriving before 7PM). We used map representations of the places and graphic representations of the time restrictions to avoid influencing subjects' language. Subjects were instructed to make separate calls for each of the 3 tasks. Subjects were also informed that they could say "Help" at any time in the initial system prompt. We did not give any additional instructions to the subjects on how to talk to the system. Instead, we let the subjects interact with the system for 2 minutes before the experiment, to get a sense of how to use the system. Subjects were compensated for their time at the end of the experiment, with no bonus for task completion. Although we set a time limit of 15 minutes as the maximum time per task, none of the subjects reached this limit.

For our user corpus, we used 4 days of calls to Let's Go (two days randomly chosen from the weekday hours of deployment, and two from the weekend hours of deployment) from the general public. Next, we introduce the evaluation measures we used to compare the subject and the user corpus.

Evaluation Measures

For high-level dialog features and dialog style, we adapt a subset of measures introduced in (Schatzmann et al., 2005a) for our system, including the duration of the dialog in seconds (dialogLen), number of turns in the entire dialog (turn), total words per user turn (U_word), number of dialog acts for system/user turn (S_action/U_action), and ratio of system and user actions (Ratio_action). We also define individual user dialog act which enables us to compute the distribution of user actions. This distribution is used to decide user action probabilities in a probabilistic user simulation. U_provideinfo and U_yesno respectively identify actions by whether the user provides information or gives a yes/no answer, while U_unkown represents all other user actions.

In addition, we define a variety of other measures that are important in a deployed spoken dialog system based on other studies (e.g., (Walker et al., 2000; Turunen et al., 2006)). These measures were not defined in our experiments in prior sections (e.g., Section 4.2.1, Section 5.2.3) on comparing simulated and real user corpora on the ITSPOKE system because we were focusing on comparing user behaviors on the dialog act level[37]. However, in this study we want to consider all factors that may differ between user and subject corpora collected with the same dialog system since we could potentially build user simulations that work on word or speech level as well. Two of our measures capture speech recognition quality: the non-understanding rate (rejection%) and the average confidence score (confScore). In addition, we look into how frequently the users ask for help (help%), how often they use touchtone (dtmf%), how often they interrupt the system (bargein%), and how fast they speak (speechRate, number of words per second). All of the features used to compute our evaluation measures are automatically extracted from system logs. Thus, the user dialog acts and dialog behavior measures are identified based on speech recognition results. For task success rate, we consider a task to be completed if and only if the system is able to get enough information from the user to start a database query and inform the user of the result (i.e., either specific bus schedule information, or a message that the queried bus route is not covered by the system).

Results

We compare the real user corpus with the subject corpus by computing the mean value for each corpus with respect to each of the evaluation measures mentioned above. We then use two-tailed t-tests to compare the means across the two corpora. All differences reported as statistically significant have p-values less than 0.05 after Bonferroni corrections. As a sanity check we first compared the weekday and weekend parts of the user corpus with respect to our set of evaluation measures. None of the measures showed statistically significant differences between these two sub-corpora.

[37] Although the simulations built for the ITSPOKE system work on the word level, the simulation model we used to decide the next system action is defined on the dialog act level. We only use a word level instantiation of the dialog act level user actions in order to provide enough information for the ITSPOKE system to work.

In Figure 8, we observe that the user dialogs and the subject dialogs are significantly different on all of the high-level dialog features. Subjects talk significantly more than users in terms of number of words per utterance; the number of turns per dialog is also higher for subjects. U_action and S_action show that both the system and the user transmit more information in the subject dialogs. Ratio_action shows that subjects are more passive than users, in the sense that they produce relatively less actions than the system.

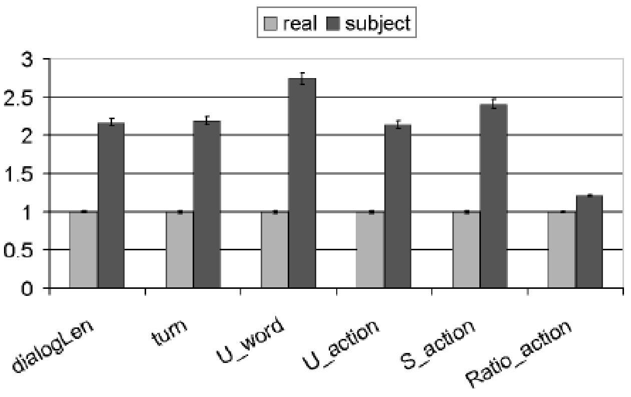


Figure 8. Comparing High-level Dialog Features

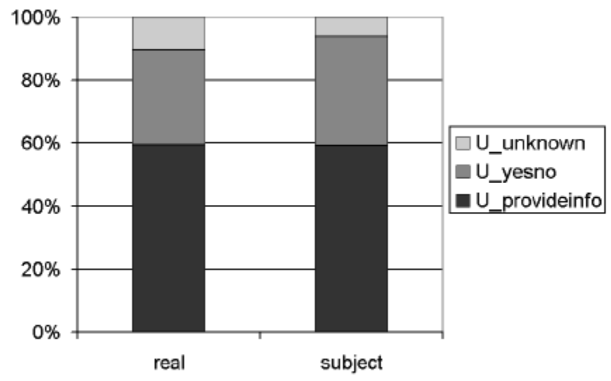


Figure 9. Comparing User Dialog Acts

When we look into the distribution of the user actions in both the user and subject corpora in Figure 9, we see that subjects give more yes/no answers and produce fewer unrecognized actions than users (these differences are statistically significant). On the other hand, there is no significant difference in U_provideinfo between users and subjects.

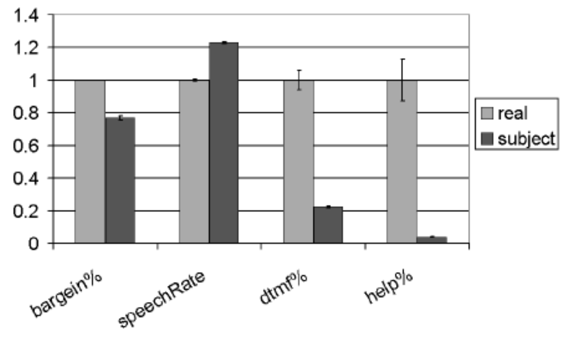


Figure 10. Comparing User Dialog Behaviors

In terms of speech recognition quality, there are no statistically significant differences between the number of rejected user turns or the average confidence scores of the speech recognizer. Recall, however, that these measures are automatically calculated using recognition results. Until we can examine speech recognition quality using manual transcriptions, we believe that it is premature to conclude that our speech recognizer performs equally well in real and lab environments. In Figure 10, we also find out that users barge in more frequently, use more DTMF inputs, and ask for more help than subjects, while subjects speak faster than users.

Finally, in addition to comparing our corpora on the dialog level, we also present a brief examination of the differences between the first user utterances from the dialogs in each corpus. (Because we are only looking at a small percentage of our user utterances, here we are able to use manual transcriptions rather than speech recognition output.) The impact of open system initial prompts on user initial utterances is an interesting question in dialog research (Raux et al., 2006). Most users answer the initial open prompt of Let’s Go (“What can I do for you?”) with a specific bus route number, while subjects often start with a departure place or destination. Subject queries may be restricted by the assigned task scenarios. However, it is interesting to note that many users call the system to obtain schedule information for a bus route they already know, rather than to get information on how to reach a destination. We also observe that there are only 2% void utterances (when only background noise is heard) in subject dialogs, while there are 20% in user dialogs. This confirms that subjects and users dialog with the system in very different environments.

To summarize, when training a simulated user on the dialog act level, the distribution of actions we learned from the subject corpus does not provide enough information to simulate behaviors of real users. However, since the most important user action (U_provideinfo) shows similar behaviors in both subject and user corpus, we hypothesize that a user simulation trained from a subject corpus can still be used in a dialog strategy learning task like ours (defined in 5.1.1) where the user simulation only needs to work on the dialog act level. Nevertheless, the learned simulated user needs to be tuned to act more like real users, for example, by barging-in more frequently or giving up earlier. On the other hand, it is hard to build a simulated user that can mimic real user behaviors on the word/speech level when trained on subject corpus. Therefore, if some other dialog strategy learning task involves learning strategies to handle users' word or speech level behaviors, a different training corpus should be chosen for building user simulations that behave at those levels.

When considering the impact of these two sources of training data in constructing user simulation for the dialog system testing task, we hypothesize that user simulations trained from a subject corpus cannot be directly used to replace real users in system testing. User simulations that work on the dialog act level need to be tuned to mimic real user behaviors before they can be used to test the dialog strategy implemented in the system. Since the simulated users behave very differently from the human users on the word and the speech level, it is not sufficient to use the simulated users to test the natural language understanding or the speech recognition component in the dialog system. Our hypothesis can be confirmed by training user simulations from the paid subject and the real user dialog corpus and comparing the simulated user behaviors using evaluation measures we introduced in Chapter 5 for both dialog strategy learning and dialog system testing tasks.

To sum up, in this chapter we briefly discuss other approaches in constructing non-probabilistic user simulations and other factors to consider in constructing probabilistic user simulations. While there are many types of user simulations and user simulation impacting factors that are not discussed in this dissertation, we can use the comparison framework we introduced to choose the most appropriate user simulation given a specific task.

7.0 CONCLUSIONS AND FUTURE WORK

In this work, we investigate constructing user simulations for spoken dialog system development. User simulation has mainly been used in two development tasks. One is to generate large training corpora for applying reinforcement learning to learn new dialog strategies; and the other is to test dialog system performance. Although previous studies have shown successful examples of applying user simulations in both of the tasks, it is not clear what type of user simulation is most appropriate for a specific task since those studies are conducted on different dialog systems of different domains. In this research, we provide an experimental framework to compare different user simulations in the same task setting. Based on a literature review, we first identify two main factors in constructing probabilistic user simulations: the choice of user simulation model and the approach to set up user action probabilities. Then, we examine each of the factors through empirical experiments. More specifically, we construct variations of user simulations which differ in one of the factors and then compare these user simulations in specific dialog system development tasks.

We introduce two groups of evaluation measures in the user simulation comparisons. The first group of measures is task-independent (4.1), i.e., these measures are not restricted to any particular dialog system development tasks. The task-independent measures assess how human-like the simulated behaviors are by comparing the simulated dialog corpus with a human user corpus. This group of measures includes domain-independent measures which are mainly adapted from previous studies and also domain-dependent measures which we designed for our tutoring dialogs. We collect human judgments to validate the domain-independent measures before applying them in our studies. We show that human judges can rank the quality of simulated users and human users consistently. In addition, we build a ranking model using the domain-independent measures to correctly predict human judges' rankings. Since the domain-independent measures can be automatically extracted from simulation logs, they can be used to quickly assess user simulation qualities.

Although these domain-independent measures can be used to distinguish simulated and human user corpora, we observe that they cannot help to explain the differences found among the corpora. We see that even two human user corpora collected with the same dialog system are shown to be significantly different using this group of measures because the two user corpora are collected from different user populations. Therefore, even if we find that a dialog corpus is different from a human user corpus, we cannot conclude that the dialog corpus is not realistic. In order to better capture the similarities among human user corpora, we propose a domain-dependent measure (4.2) – the knowledge consistency feature for dialogs in which user knowledge is gradually modified. We show that this measure can capture the similarity among human user corpora as well as better distinguish human versus simulated user corpora than the domain-independent measures. We also use this knowledge consistency feature to improve the ranking model in order to predict the rankings of user simulations more reliably.

Since our goal of constructing user simulations is to help dialog system development, we also introduce a group of task-dependent evaluation measures to assess user simulations' performance in two dialog system development tasks: dialog strategy learning and dialog system testing. When using user simulation to generate a training corpus for applying Markov Decision

Process to learning new dialog strategies, we evaluate the user simulations by examining the quality of the dialog strategies that are learned from the simulated corpora (5.1.3). We show that in our task setting it is better to choose a simulation model which generates all possible user actions in a uniform distribution than a more realistic user simulation which mimics human behaviors in a statistical way. This suggests that a realistic user simulation is not always the best to choose in a dialog strategy learning task. Although our results may be constrained to our task setting, similar experiments can be carried out to compare user simulations in the dialog strategy learning task by using our experimental setups and evaluation measures (5.1.1). In addition, we investigate how to set up user action probabilities in the user simulation models for the dialog strategy learning task (5.1.4). We observe that a manual and a trained approach both work equally well in our experiment.

When using user simulations to test a dialog system, we believe that a user simulation which mimics human-like behaviors is highly desired since we would like the simulated users to give the same reactions as the human users when using the dialog system. We evaluate the performance of different user simulations on generating objective measures of dialog system performance on the corpus level, as well as by calculating how accurately the user simulation can predict human user actions in the same dialog context. We find that not surprisingly a user simulation which mimics human user actions in a statistical way performs better than a user simulation which generates more random behaviors in producing objective measures of dialog system performance (5.2.1). Moreover, we show that subjective measures of dialog system performance can be estimated from simulation logs (5.2.2). Thus, we suggest that user simulations can be used to replace human users in testing dialog systems. We also observe that for the dialog system testing task, it is better to use a reasonable-sized human user corpus to train a simulation than handcrafting a user simulation totally based on expert knowledge (5.2.3). Therefore when constructing user simulations for a new dialog system when no human user data is available, we suggest to start with handcrafted user action probabilities but re-train these probabilities after an initial dialog system is built and human user data is available.

We focus on probabilistic user simulations in our study since our goal is to provide a general guideline for constructing user simulations for researchers and probabilistic user simulations are the most widely used simulation type. Also, we build our user simulations using similar approaches as other studies, for example, to train user simulations from human user corpus collected from lab settings, so that our results are more representative. However, there are other types of non-probabilistic user simulations that have been proven to be useful and other factors to consider in building probabilistic user simulations. In Chapter 6, we look at some issues that are less considered in constructing user simulations. In Section 6.1, we summarize some non-probabilistic user simulations and compare them against probabilistic user simulations. We suggest that different types of user simulations should be chosen regards to task and cost constraints. In Section 6.2, we investigate another factor that impact probabilistic user simulations but has been less considered in research studies, i.e., simulation training data. We compare a dialog corpus collected with paid subjects and another dialog corpus collected with real users with the same dialog system. We observe that the two corpora are different on the dialog act level in terms of user action distributions. The two groups of users differ even more in speech-level and lexical-level features. Therefore, we infer that user simulations trained from a subject corpus do not necessarily represent behaviors of real users. However, when real user corpora are not available, user simulations which work on the dialog act level may be trained from a subject corpus and then be tuned to mimic more realistic user behaviors. Nevertheless, it is hard to train user simulations which work on the word or the speech level from a subject corpus to generate real user behaviors. We point out that different factors which impact user simulation performance should be considered for different tasks and these comparisons can be done using the experimental framework we provide.

To summarize, this research contributes to user simulation development from two aspects. First, we provide an experimental framework and evaluation measures for comparing user simulation performance in specific dialog system development tasks. We suggest that a realistic user simulation is not always needed in assisting dialog system development. Second, we validate previously used domain-independent user simulation evaluation measures as well as propose a domain-dependent evaluation measure that can be used in dialogs where users' knowledge is gradually modified, such as tutoring dialogs, trouble shooting dialogs, and cooperative problem solving dialogs.

Several questions remain to be answered by future research. Our results are constrained by the task domain we worked with. As we mentioned in Section 3, the tutoring domain is different from the information-providing task domain in terms of both dialog system and user simulation behaviors. In addition, our results can be constrained by the resources that were available at the time of our studies. For example, the size of human user data would impact the trained user action probabilities. Also, the expertise of the human expert that we use in handcrafting user action probabilities will impact our results as well. Therefore, follow-up studies which replicate our experiments on different dialog systems by using richer resources will help to confirm our results. In addition, in Section 6.2 we only infer the impact of training data on simulated users' behaviors based on a corpus comparison study without comparing simulated users' behaviors directly. In the future, we will train separate user simulations from real user data and paid subject data separately and then compare the user simulations' behaviors using similar experimental setups used in Chapter 5.

Our study reveals opportunities for several future research directions. Since we show in Section 4.2 that adding domain-dependent features can improve user simulation performance, it is interesting to explore such features for each application domain, e.g., clinical reception desk, home entertainment, etc. Another direction to pursue is to model the changes of user behaviors based on the progress of dialogs. Our CLU model (4.2.1) is a simple model which mimics student learning during tutoring dialogs. The students' learning rates are fixed in this model. It will be interesting to examine how users' learning rates change after interacting with a tutoring system. More generally, user action probabilities in the probabilistic user simulation models may change during the user-system interaction. Learning these probabilities online will hopefully improve the user simulation performance. In our research, we only use user simulations in applying Markov Decision Process to design new dialog strategies. In fact, as we reviewed in Section 2.3, there are different machine learning approaches that we can use in designing dialog strategies. It is interesting to explore the relationships between the type of machine learning approach and the type of simulated corpus that is required to apply a particular approach. In addition, user simulations can be used to design other modules in a dialog system, e.g., speech recognition module, language understanding module, or language generation module. Although previous research (e.g., (Möller et al., 2006)) show successful use of user simulations for designing these modules, more systematic studies are required to provide general guidelines for constructing efficient user simulations for these tasks. Finally, user simulations can be applied to other natural language processing tasks other than dialog system development. For example, user simulations which reflect user preferences can be built to test information retrieval systems or question answering systems which are similar to dialog systems since users interact with all these systems by sending queries in natural language.

APPENDIX A

ITSPOKE PROBLEM STATEMENTS

Problem 7. Suppose a man is running in a straight line at constant speed. He throws a pumpkin straight up. Where will it land? Explain.

Problem 34. The sun pulls on the earth with the force of gravity and causes the earth to move in orbit around the sun. Does the earth pull equally on the sun? Defend your answer.

Problem 38. Suppose a lightweight car and a massive truck hit a patch of frictionless ice and have a head-on collision. Upon which vehicle is the impact force greater? Which vehicle undergoes the greater change in its motion? Defend your answers.

Problem 55. An airplane flying horizontally drops a packet when it is directly above the center of a swimming pool. Does the packet hit that spot? Explain.

Problem 58. Suppose a man is in a free-falling elevator and is holding his keys motionless right in front of his face. He then lets go. What will be the position of the keys relative to the man's face as time passes? Explain."

APPENDIX B

HUMAN ASSESSMENT STUDY QUESTIONAIRE

You are going to read the transcription of a tutoring dialog. This dialog happens between a computer tutor and a student. The tutor helps the student to solve a physics problem. The student can be either a human or a computer. The aim of this study is to investigate how a human student and a computer student can be distinguished given their performance in the dialog. You are first going to answer three questions assessing the quality of each student utterance. Then, you are asked two additional questions about your overall impression on the student.

Note: Please give your ratings as fast as you feel comfortable. You are strongly encouraged to provide your intuitive reaction. Please do not ponder your decision.

Part 1. Please read the dialog and rate EACH STUDENT utterance (i.e., turn) based on the following questions:

1. A good participant in a dialog should make his/her contribution as informative as is required without giving unnecessary information. Do you think this student's answer gives enough information for the tutor to respond?

A. Definitely NOT enough information

B. Probably NOT enough information

C. Cannot tell

D. Probably enough information

E. Definitely enough information

2. A good participant in a dialog relates his/her points to the previous topic in the dialog. Do you think this student's utterance is relevant to the previous tutor questions?

A. Definitely NOT relevant

B. Probably NOT relevant

C. Cannot tell

D. Probably relevant

E. Definitely relevant

3. A good participant in a dialog should express his/her ideas clearly and briefly. Do you think this student's utterance is clear?

A. Definitely No.

B. Probably No.

C. Cannot tell.

D. Probably Yes.

E. Definitely Yes.

Part 2. Now you've read the whole dialog. Please answer the following questions based on your impression from the dialog. Do not read the dialog again.

1. Please give an overall rating of the student's performance. Is this student a computer or a human?

A. Definitely computer

B. Probably computer

C. Cannot tell

D. Probably human

E. Definitely human

Please give your reason in one sentence: _________________________

2. Do you think the tutoring scenario presented in this dialog is similar to a one-on-one tutoring dialog between humans? Please note that even if you think the student is played by a computer, the tutoring scenario presented in the dialog can still be similar to a human-human tutoring scenario.

A. Definitely NOT realistic

B. Probably NOT realistic

C. Cannot tell

D. Probably realistic

E. Definitely realistic

Please give your reason in one sentence: ________________________

3. If you are going to form a physics study group with this student, do you want to have he/she as your partner?

A. Definitely No.

B. Probably No.

C. Cannot tell.

D. Probably Yes.

E. Definitely Yes.

Please give your reason in one sentence: ________________________

APPENDIX C

EXAMPLE ITSPOKE DIALOGS[38]

An example dialog on Problem 34 by a human student:

Tutor: To analyze this problem which law of motion would you use?

Student: two

Tutor: The best law of motion to use in this case is Newton's third law because it relates forces of interacting objects. So we will use Newton's third law to analyze this problem. Do you recall what this law says?

Student: no.

Tutor: Okay. Newton's third law says every force has an equal and opposite reaction force. That is if there is a force acting on object A due to object B then there is also a force acting on B due to A. These two forces have the same magnitudes but opposite directions. Moreover they are the same type of force. if one is a gravitational force then so is the other. If one is a frictional force then so is the other. In this case there is a gravitational force on the earth due to the sun. Is there a gravitational force on the sun due to the earth?

Student: yes

[38] The student utterances are outputs from speech recognition components.

Tutor: Fine. Now what is the relationship between the two forces magnitudes?

Student: sun's is larger

Tutor: Well... According to Newton's third law the magnitudes are the same for the force that the earth exerts on the sun and the force that the sun exerts on the earth. What is the relationship between the two forces directions?

Student: opposite.

Tutor: According to Newton's third law the force that the earth exerts on the sun and the force that the sun exerts on the earth have opposite directions. To summarize Newton's third law says that the gravitational force on the sun due to the earth is equal and opposite to the gravitational force on the earth due to the sun.

An example dialog on Problem 38 by a human student:

Tutor: The first question in this problem asks about forces applied on mutually interacting objects. Which of Newton's laws would you use to answer this question?

Student: two.

Tutor: Whenever you are asked about forces on two objects and the objects are interacting (e.g. pushing or pulling on each other) then you should use Newton's third law. What does Newton's third law say?

Student: don't know.

Tutor: Alright. Newton's third law says that every force has an equal and opposite reaction force. That is if there is a force acting on object A due to object B then there is also a force acting on B due to A. The two forces have the same magnitude and opposite directions. So in our problem upon which vehicle is the impact force greater?

Student: same neither.

Tutor: Right. According to Newton's third law both vehicles receive an impact force of the same magnitude. Now to answer the second question which asks about the change of motion in each individual vehicle what law of motion would you use?

Student: two.

Tutor: Well... You could use Newton's second law. What is the definition of Newton's second law?

Student: don't know.

Tutor: Okay. We can define Newton's second law as F=m*a where F is the net force acting on the objectmis the mass of the object andais the object's acceleration. So first we need to look at all the forces acting on the vehicles from that determine the net force and finally compare accelerations. What are the forces acting on the car?

Student: air.

Tutor: Another force acting on the car is the force of the impact which is in the horizontal direction. Another force acting on the car is the normal force which is applied by the road. What is the direction of the normal force in this case?

Student: forward.

Tutor: The normal force is always perpendicular to the surface applying it. Hence the normal force that the road exerts on the car is vertically up. One of the forces acting on the car is the force of gravity which is what we usually call weight. The force of gravity is always present when an object is in a gravitational field such as the one produced by the earth. So back to the original question. We can define Newton's second law as F=m*a where F is the net force acting on the object is the mass of the object and the object's acceleration. So first we need to look at all the forces acting on the vehicles from that determine the net force and finally compare accelerations. What are the forces acting on the car?

Student: force of the impact.

Tutor: Almost. Gravity and normal force and impact force. Just to make everything totally precise the direction of gravity is vertically down the direction of the normal force is vertically up and the impact force is horizontal. During the impact what is the NET force applied to the car equal to?

Student: horizontal.

Tutor: Well... The net force is the resultant force. We have just found that there are three forces acting on the car namely the normal force the weight and the impact force. Since there is no motion in the vertical direction there is no net force in that direction. Hence the net force on the car equals the impact force on the car. What is the NET force acting on the truck equal to?

Student: impact force.

Tutor: Correct. Hence if we apply Newton's second law to each vehicle what can we say about the acceleration of the truck compared to the acceleration of the car?

Student: same.

Tutor: If we apply Newton's second law to the car we have F = mc * ac where is the car's mass and if the car's acceleration. If we apply it to the truck we have F = mt * at where mt is the truck's mass and at is the truck's acceleration. Since the impact force is the same for both vehicles we have mc * ac = mt * at. From this you can determine how the acceleration of the car compares to that of the truck. Which one is larger?

Student: truck.

Tutor: Well... Given that the truck's mass (mt) is larger than the car's mass (mc) for the equation mc * ac = mt * at to hold the truck's acceleration (at) has to be smaller than the car's acceleration (ac). Once again on the original question. Hence if we apply Newton's second law to each vehicle what can we say about the acceleration of the truck compared to the acceleration of the car?

Student: smaller.

Tutor: Smaller than the car. So if the car has a larger acceleration than the truck which vehicle undergoes the greater change in motion?

Student: truck.

Tutor: To answer this question we will use the definition of acceleration. What is the definition of acceleration?

Student: change in velocity.

Tutor: Well... Acceleration is defined as the rate of change of velocity with time. This means that acceleration describes how rapidly the velocity changes. So if the velocity of the car is changing more rapidly than the truck's velocity then the car will undergo the greater change in motion. Let's try the original question again. So if the car has a larger acceleration than the truck which vehicle undergoes the greater change in motion?

Student: car.

Tutor: That's super. To summarize we knew to use Newton's third law for the first question because it asked about forces applied by mutually interacting objects. On the other hand because the second question asked about the objects motions we knew to use the chain of inferences: forces net force acceleration velocity and displacement. In this case we started with forces and ended up with accelerations. We didn't need to consider velocities or displacements.

An example dialog on Problem 58 by a human student:

Tutor: To analyze this problem we will first describe the motion of the person and his keys while he is holding them. Then we will look at the motion of the person and his keys after he lets go of them. Let's begin by looking at the motion of the man and his keys while he is holding them. How does his velocity compare to that of his keys?

Student: same.

Tutor: Uh huh. So while he is holding his keys they both have the same velocity. Now let's see what happens when he releases them. To do so we will first find the individual forces exerted on the man and his keys then both net forces both accelerations and finally we will describe and compare the velocities. So what are the forces exerted on the man after he releases his keys? Please specify their directions (e.g. vertically up).

Student: vertically down.

Tutor: Well... After the man releases the keys the only force acting on him is the force of gravity which is what we usually call weight. The force of gravity is always present when an object is in a gravitational field such as the one produced by the earth and it is always straight down. Because there is just one force gravity's it is trivial to determine the NET force (i.e. the vector sum of all forces). So what is the direction of the NET force on the person?

Student: down.

Tutor: Fine. Considering the net force does the person have an acceleration? If yes please specify its direction.

Student: yes down.

Tutor: Uh huh. What is the numerical value of the man's acceleration?

Student: zero.

Tutor: All objects in freefall have the same acceleration. So that would be the man's acceleration as well. So what is the value of freefall acceleration?

Student: zero.

Tutor: To deduce the numerical value of the man's acceleration we use Newton's second law and the magnitude of the gravitational force. What is the equation for calculating weight (i.e. the magnitude of the gravitational force)?

Student: weight equals.

Tutor: Okay. Weight is calculated with w=m*g where m is the mass of the object and g is the acceleration of gravity. What is the definition of Newton's second law?

Student: forces accelerates objects.

Tutor: Newton's second law says that the net force acting on an object equals the mass times the acceleration of the object. We can define Newton's second law as F=m*a where F is the net force acting on the object m is the mass of the object and a is the object's acceleration. In our case the net force is the weight. Hence F=W or m*a=m*g. Therefore the man's acceleration is g or 9.8m/s^2. So back to the original question. All objects in freefall have the same acceleration. So that would be the man's acceleration as well. So what is the value of freefall acceleration?

Student: nine point eight.

Tutor: Good job. So once again on the original question. What is the numerical value of the man's acceleration?

Student: nine point eight.

Tutor: Yeah. Let's now analyze what happens to the keys. So what are the forces acting on the keys after they are released? Please specify their directions (e.g. vertically up).

Student: down.

Tutor: Well... After the keys are released the only force acting on them is the force of gravity which is what we usually call weight. The force of gravity is always present when an object is in a gravitational field such as the one produced by the earth. What is the direction of the force of gravity?

Student: vertically down.

Tutor: Uh huh. Let's try the original question again. Let's now analyze what happens to the keys. So what are the forces acting on the keys after they are released? Please specify their directions (e.g. vertically up).

Student: gravity vertically down.

Tutor: Correct. Because gravity is the only force it is trivial to determine the NET force (i.e. the vector sum of all forces).So what is the direction of the NET force on the keys?

Student: yes downward.

Tutor: Yeah. What is the numerical value of the acceleration of the keys?

Student: nine point eight.

Tutor: Fine. So far we know that both the man and his keys have an acceleration of 9.8m/s^2 (that is g). Now we will describe their velocities. In a linear motion problem such as this one if an object has an acceleration of 9.8m/s^2 what is the change in the object's velocity in one second?

Student: zero.

Tutor: Alright. When an object moving linearly has an acceleration of 9.8m/s^2 it means that every second the object's velocity varies by 9.8m/s. For example if the object begins with a velocity of zero what will its velocity be a second later?

Student: nine point eight.

Tutor: Well... We have just said that if a falling object has an acceleration of 9.8m/s^2 that means that its velocity changes by 9.8m/s every second. So if the object began with a velocity of zero a second later it will have a velocity of 9.8m/s. What will the velocity of the object be a second after that (where the initial velocity is 9.8m/s the acceleration is 9.8m/s^2)?

Student: seventeen point six.

Tutor: As we have seen if a falling object has an acceleration of 9.8m/s^2 that means that its velocity changes by 9.8m/s every second. So if a second after it began falling the object's velocity is 9.8m/s a second later its velocity will be 9.8m/s+9.8m/s=19.6m/s. So what will its velocity be a second after it is 19.6m/s?

Student: twenty eight point four.

Tutor: A second later the object's velocity will be 19.6m/s+9.8m/s=29.4m/s. So back to the original question. What will the velocity of the object be a second after that (where the initial velocity is 9.8m/s the acceleration is 9.8m/s^2)?

Student: thirty eight point two.

Tutor: 19.6 m/s. So at every point in time during the fall of the man and his keys how do their velocities compare with each other?

Student: same.

Tutor: Right. Now we will compare the displacements of the man and his keys. Do you recall what displacement means?

Student: no.

Tutor: Okay. Displacement can be defined as the total change in position during the elapsed time. With this in mind how does the man's displacement compare to that of his keys at every point in time during the fall?

Student: greater than.

Tutor: We have just established that at every point in time while the man and his keys are falling they have the same velocity. Velocity is defined as the rate of change of position with time. Hence since they both have the same velocity they are both changing their position in the same way. Therefore at every point in time they will be at the same height. So we can conclude that the man and his keys will fall together that is the keys will remain in front of the man's face during their whole trip down. And now you know how to prove it with physics. As in many problems you start by considering the forces on the objects. This determines the net force (vector sum of forces) on each object. From the net force you can determine the acceleration of each object. From the acceleration you can determine the velocity. From the velocity you can determine the displacement. So quantities are linked in this order: forces net force acceleration velocity and displacement. In this problem we went from the forces all the way to displacements. In other problems we only need to consider part of the chain.

BIBLIOGRAPHY

Ai, H. and Litman, D. 2006. Comparing Real-Real, Simulated-Simulated, and Simulated-Real Spoken Dialog Corpora. Proceedings of the AAAI Workshop.

Ai, H., Tetreault, J., and Litman, D. 2007a. Comparing User Simulation Models for Dialog Strategy Learning. In Proc. NAACL-HLT.

Ai, H. and Litman, D. 2007. Knowledge Consistent User Simulations for Dialog Systems. In Proc. Interspeech 2007.

Ai, H., Raux, A., Bohus, D., Exkenazi, M., and Litman, D. 2007b. Comparing Spoken Dialog Corpora Collected with Recruited Subjects versus Real Users. In Proc. 8th SIGDial Workshop on Discourse and Dialog.

Ai, H. and Litman, D. 2008. Assessing Dialog System User Simulation Evaluation Measures Using Human Judges. In Proc. 46th ACL.

Ai, H. and Weng, F. 2008. User Simulation as Testing for Spoken Dialog Systems. In Proc. 9th SIGDial Workshop on Discourse and Dialog.

Ai, H. and Litman, D., 2009. Setting Up User Action Probabilities in User Simulations for Dialog System Development. In Proc. Of 47th ACL.

Allen, J. F., Miller, B. W., Ringger, E. K., and Sikorski, T. 1996. A Robust System for Natural Spoken Dialog. In Proceedings of the 34nd AnnualMeeting of the Association for Computational Linguistics (ACL).

Batliner, A., Fischer, K., Huber, R., Spilker, J., and Noth, E. 2003. How to Find Trouble in Communication. Speech Communication, 40:1-2.

Blatz, J., Fitzgerald, E., Foster, G., Gandrabur, S., Goutte, C., Kulesza, A., Sanchis, A., and Ueffing N., 2004. Confidence Estimation for Machine Translation. John Hopkins Summer Workshop on Machine Translation.

Bohus, D. and Rudnicky, A. 2003. RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. In Proceedings of Eurospeech 2003.

Cen, H., Koedinger, K., and Junker, B. 2006. Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. In Proc. the 8th International Conference on Intelligent Tutoring Systems.

Chung, G. 2004. Developing a Flexible Spoken Dialog System Using Simulation. In Proc. of ACL 04.

Cuayáhuitl, H., Renals, S., Lemon, O., and Shimodair, H. 2009. Evaluation of a hierarchical reinforcement learning spoken dialog system. Computer Speech & Language, 24:2.

Eckert, W., Levin, E., and Pieraccini, R. 1997. User modelling for spoken dialog system evaluation. In Proc. of ASRU.

Edlund, J., Gustafson, J., Heldner, M., and Hjalmarsson, A. 2008. Towards Human-Like Spoken Dialog Systems. Speech Communication, 50:8-9.

Engelbrecht, K., Quade, M., and Möller, S. 2009. Analysis of a new simulation approach to dialog system evaluation. Speech Communnication, 51:12.

Filisko, E. and Seneff, S. 2005. Developing city name acquisition strategies in spoken dialog systems via user simulation. Proc. of 6th SIGDial Workshop on Discourse and Dialog.

Forbes-Riley, K. and Litman, D. 2005. Using Bigrams to Identify Relationships Between Student Certainness States and Tutor Responses in a Spoken Dialog Corpus. Proceedings of 6th SIGdial Workshop on Discourse and Dialog.

Forbes-Riley, K., Litman, D., Silliman, S., and Tetreault, J. 2006. Comparing Synthesized versus Pre-recorded Tutor Speech in an Intelligent Tutoring Spoken Dialog System. Proceeding 19th International FLAIRS (Florida Artificial Intelligence Research Society) Conference.

Forbes-Riley, K., Litman, D., and Rotaru, M. 2008. Responding to Student Uncertainty during Computer Tutoring: An Experimental Evaluation. Proceedings 9th International Conference on Intelligent Tutoring Systems (ITS).

Foster, M. E. 2008. Automated metrics that agree with human judgements on generated output for an embodied conversational agent. In Proceedings of the 5th International Natural Language Generation Conference.

Foster, M. E., Giuliani, M., and Knoll, A. 2009. Comparing Objective and Subjective Measures of Usability in a Human-Robot Dialog System. In Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics.

Freund, Y., Iyer, R., Schapire, R. E. and Singer, Y. 2003. An Efficient Boosting Algorithm for Combining Preferences. Journal of Machine Learning Research.

Fukubayashi, Y., Komatani, K., Ogata, T., and Okuno, H. 2006. Dynamic Help Generation by Estimating User's Mental Model in Spoken Dialog Systems. In Proc. of Interspeech.

Georgila, K., Henderson, J., and Lemon, O. 2005. Learning user simulations for information state update dialog systems. Proc. of 9th European Conference on Speech Communication and Technology (Eurospeech).

Georgila, K., Henderson, J., and Lemon, O. 2006. User Simulation for Spoken Dialog Systems: Learning and Evaluation. In Proc. of Interspeech.

Georgila, K., Wolters, M., and Moore, J. 2008. Simulating the Behaviour of Older versus Younger Users when Interacting with Spoken Dialog Systems. In Proc. of 46th ACL.

Giorgino, T., Quaglini, S., and Stefanelli, M. 2004. Evaluation and Usage Patterns in the Homey Hypertension Management Dialog System. Dialog Systems for Health Communication,AAAI Fall Symposium, Technical Report FS-04-04.

Grice, H. P. 1975. Logic and Conversation. Syntax and Semantics III: Speech Acts.

Gruenstein, A., Seneff, S., and Wang, C. 2006. Scalable and Portable Web-Based Multimodal Dialog Interaction with Geographical Databases. In Proc. of ICSLP 2006.

Harman, D., and Over, P., 2004. The Effects of Human Variation in DUC Summarization Evaluation. In Proc. of ACL-04 Workshop: Text Summarization Braches Out.

Henderson, J., Lemon, O., and Georgila, K. 2005. Hybrid Reinforcement/Supervised Learning for Dialog Policies from COMMUNICATOR data. In IJCAI workshop on Knowledge and Reasoning in Practical Dialog Systems.

Hof, A., Hagen, E., and Huber, A. 2006. Adaptive Help for Speech Dialog Systems Based on Learning and Forgetting of Speech Commands. In Proc. of 7th SIGdial.

Horvitz, E. and Paek, T. 1999. A Computational Architecture for Conversation. In Proc. Of Seventh Int'l Conf. User Modeling.

Hone, K. S. and Graham, R. 2000. Towards a tool for the subjective assessment of speech system interfaces (SASSI), Natural Language Engineering 6.

Janarthanam, S. and Lemon, O. 2008. User simulations for online adaptation and knowledge-alignment in Troubleshooting dialog systems. In Proc. of the 12th SEMdial Workshop on on the Semantics and Pragmatics of Dialogs.

Jung, S., Lee, C., Kim, K., Jeong, M., and Lee, G. 2009. Data-driven user simulation for automated evaluation of spoken dialog systems. Computer Speech & Language, 23:4.

Karypis, G. 2002. CLUTO - a clustering toolkit. Technical Report 02-017, University of Minnesota.

Levin, E., Pieraccini, R., and Eckert, W. 2000. A stochastic model of human-machine interaction for learning dialog strategies. IEEE Trans. on Speech and Audio Processing, 8(1):11–23.

Lemon, O., Georgila, K., Henderson, J. 2006. Evaluating Effectiveness and Portability of Reinforcement Learned Dialog Strategies with real users: the TALK TownInfo Evaluation. In Proc. of IEEE/ACL Spoken Language Technology.

Lemon, O. and Pietquin, O. 2007. Machine Learning for Spoken Dialog Systems. Tutorial paper in Interspeech 2007.

Lemon, O. and Liu, X. 2007. Dialog Policy Learning for combinations of Noise and User Simulation: transfer results. In Proc. of SIGdial 2007.

Lin, C. Y. 2004. ROUGE: A package for automatic evaluation of summaries. In Proceedings of the ACL 2004 Workshop on Text Summarization.

Linguistic Data Consortium. 2005. Linguistic Data Annotation Specification: Assessment of Fluency and Adequacy in Translations.

Litman D. and Pan, S. 2002. Designing and Evaluating an Adaptive Spoken Dialog System. User Modeling and User-Adapted Interaction. Vol. 12, No. 2/3.

Litman, D., and Silliman, S. 2004. ITSPOKE: An Intelligent Tutoring Spoken Dialog System. In Companion Proc. of the Human Language Technology: NAACL.

Litman, D., Rośe, C., Forbes-Riley, K., VanLehn, K., Bhembe, D., and Silliman, S. 2004. Spoken Versus Typed Human and Computer Dialog Tutoring. Proceedings of the Seventh International Conference on Intelligent Tutoring Systems (ITS).

Litman, D. and Forbes-Riley, K. 2005. Speech Recognition Performance and Learning in Spoken Dialog Tutoring. Proceedings 9th European Conference on Speech Communication and Technology.

López-Cózar, R., Torre, A., Segura, J., and Rubio, A.. Assessment of dialog systems by means of a new simulation technique. Speech Communication, 40:387–407, 2003.

Mairesse, F., Walker, M., Mehl, M., and Moore, R. 2007. Using Linguistic Cues for the Automatic Recognition of Personality in Conversation and Text. Journal of Artificial Intelligence Research, Vol 30.

Matsuda, N., Cohen, W., & Koedinger, R. 2005. Building Cognitive Tutors with Programming by Demonstration. In S. Kramer & B. Pfahringer (Eds.), Technical report: TUM-I0510 (Proceedings of the International Conference on Inductive Logic Programming).

Möller, S., Engelbrecht, K. and Schleicher, R. 2008. Predicting the Quality and Usability of Spoken Dialog Services. Speech Communication. Elsevier Science Publishers B. V, 730–744.

Möller, S., Smeele, P., Boland, H. and Krebber, J. 2007. Evaluating Spoken Dialog Systems According to De-Facto Standards: A Case Study. Computer Speech and Language, 26–53.

Möller, S., Krebber, J., and Smeele, P. 2006. Evaluating the speech output component of a smart-home system. Speech Communication (48): 1-27.

Möller, S. 2005a. Quality of Telephone-Based Spoken Dialog Systems. Springer.

Möller, S. 2005b. Towards generic quality prediction models for spoken dialog systems - a case study. In Proc. INTERSPEECH.

Paek, T. 2006. Reinforcement learning for spoken dialog systems: Comparing strengths and weaknesses for practical deployment. In Proc. of Interspeech-06 Workshop on “Dialog on Dialogs – Multidisciplinary Evaluation of Advanced Speech-based Interactive Systems”.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. 2002. BLEU: A method for automatic evaluation of machine translation. In Proceedings of ACL 2002.

Pietquin, O. 2004. A Framework for Unsupervised Learning of Dialog Strategies. Ph.D. diss, Faculte Polytechnique de Mons.

Raux, A., Langner, B., Bohus, D., Black, A., and Eskenazi, M. 2005. Let’s Go Public! Taking a Spoken Dialog System to the Real World. In Proceedings of Eurospeech.

Raux, A., Bohus, D., Langner, B., Black, A., and Eskenazi, M. 2006. Doing Research on a Deployed Spoken Dialog System: One Year of Let’s Go! Experience. In Proceedings of Interspeech.

Radev, D., Qi, H., Wu, H., and Fan, W. 2002. Evaluating web-based question answering systems. In Proc. Of LREC 2002.

Rieser, V., Kruijff-Korbayova, I., and Lemon, O. 2005. A corpus collection and annotation framework for learning multimodal clarification strategies. In Proceedings of SIGdial 2005.

Rieser, V., and Lemon, O. 2006. Cluster-based User Simulations for Learning Dialog Strategies. In Proceedings of Interspeech.

Rieser, V. 2008. Bootstrapping Reinforcement Learning-based Dialog Strategies from Wizard-of-Oz data. Ph.D. thesis, Saarland University.

Roque, A., Leuski, A., Rangarajan, V., Robinson, S., Vaswani, A., Narayanan, S., and Traum, D. 2006. Radiobot-cff: A spoken dialog system for military training. In Proceedings of International Conference on Spoken Language Processing 2006.

Rotaru, M. 2008. Applications of Discourse Structure for Spoken Dialog Systems. Ph.D. Dissertation, University of Pittsburgh.

Schatzmann, J., Georgila, K., and Young, S. 2005a . Quantitative Evaluation of User Simulation Techniques for Spoken Dialog Systems. In Proc. of 6th SIGdial Workshop on Discourse and Dialog.

Schatzmann, J., Stuttle, M., Weilhammer, K. and Young, S. 2005b. Effects of the User Model on Simulation-based Learning of Dialog Strategies. In Proc. of ASRU.

Schatzmann, J., Weilhammer, K., Stuttle, M., and Young, S. 2006. A Survey of Statistical User Simulation Techniques for Reinforcement-Learning of Dialog Management Strategies. Knowledge Engineering Review, Cambridge University Press, Vol. 21(2).

Schatzmann, J., Thomson, B., Weilhammer, K., Ye, H., and Young. S. 2007a. Agenda-Based User Simulation for Bootstrapping a POMDP Dialog System. In Proc. of NAACL-HLT2007.

Schatzmann, J., Thomson B., and Young, S. 2007b. Statistical User Simulation with a Hidden Agenda. In Proc. of SigDIAL.

Schatzmann, J., Thomson, B., and Young, S. 2007c. Error Simulation for Training Statistical Dialog Systems. In Proc. of ASRU 07.

Scheffler, K. and Young, S. 2001. Corpus-based Dialog Simulation for Automatic Strategy Learning and Evaluation. In Proc NAACL-2001 Workshop on Adaptation in Dialog Systems.

Scheffler, K. 2002. Automatic Design of Spoken Dialog Systems. Ph.D. diss., Cambridge University.

Singh, S., Kearns, M., Litman, M., and Walker, M. 2000. Empirical Evaluation of a Reinforcement Learning Spoken Dialog System. Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence.

Singh, S., Litman, D., Kearns, M., and Walker, M. 2002. Optimizing Dialog Managment with Reinforcement Learning: Experiments with the NJFun System. Journal of Artificial Intelligence Research.

Tetreault, J. and Litman, D. 2006. Comparing the Utility of State Features in Spoken Dialog Using Reinforcement Learning. In Proc. NAACL2006.

Tetreault, J., Bohus, D., and Litman, D. 2007. Estimating the Reliability of MDP Policies: A Confidence Interval Approach. In Proc. NAACL2007.

Tomko, S. and Rosenfeld, R. 2006. Shaping user input in speech graffiti: a first pass. In Proc. Of CHI Extended Abstracts.

Turunen, M., Hakulinen J., and Kainulainen, A. 2006. Evaluation of a Spoken Dialog System with Usability Tests and Long-term Pilot Studies: Similarities and Differences. In Proceedings of Interspeech2006.

VanLehn, K., Jordan, P. W., Rośe, C. P., Bhembe, D., B¨ottner, M., Gaydos, A., Makatchev, M., Pappuswamy, U., Ringenberg, M., Roque, A., Siler, S., Srivastava, R., and Wilson, R. 2002. The architecture of Why2-Atlas: A coach for qualitative physics essay writing. In Proc. Intelligent Tutoring Systems Conference.

VanLehn, K. 2006. The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education.

Marilyn A. Walker. 1995. Testing collaborative strategies by computational simulation: Cognitive and task effects. Knowledge Based Systems.

Walker, M., Litman, D., Kamm, C., and Abella, A. 1997. PARADISE: A Framework for Evaluating Spoken Dialog Agents. In Proceedings of the 35th ACL.

Walker, M., Kamm, C., and Litman, D. 2000. Towards Developing General Models of Usability with PARADISE. In Natural Language Engineering, 6:3.

Walker, M., Aberdeen, J., Boland, J., Bratt, E., Garofolo, J., Hirschman, L., Le, A., Lee, A., Narayanan, S., Papineni, K., Pellom, B., Polifroni, J., Potamianos, A., Prabhu, P., Rudnicky, A., Sanders, G., Seneff, S., Stallard, D., and Whittaker, S. 2001a. DARPA Communicator dialog travel planning systems: The June 2000 data collection. In Proc. EUROSPEECH.

Walker, M., Rambow, O., and Rogati, M. 2001b. SPoT: A Trainable Sentence Planner. In Proc. of NAACL 2001.

Walker, M., Whittaker, S., Stent, A., Maloor, P., Moore, J., Johnston, M. and Vasireddy, G. 2005. Generation and Evaluation of User Tailored Responses in Multimodal Dialog. Cognitive Science, 28:5.

Wei, X. and Rudnicky, A. 1999. An agenda-based dialog management architecture for spoken language systems. In Proc. of IEEE ASRU.

Weng, F., Varges, S., Raghunathan, B., Ratiu, F., Pon-Barry, H., Lathrop, B., Zhang, Q., Bratt, H., Scheideck, T., Mishra, R., Xu, K., Purvey, M., Lien, A., Raya, M., Peters, S., Meng, Y., Russell, J., Cavedon, L., Shriberg, E., and Schmidt, H. 2006. CHAT: A Conversational Helper for Automotive Tasks. In Proc. of Interspeech2006.

Williams J. and Young, S. 2005. Scaling Up POMDPs for Dialog Management: The "Summary POMDP" Method. Proc IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU).

Williams, J. 2007. A Method for Evaluating and Comparing User Simulations: The Cramer-von Mises Divergence. In Proc IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU).

Williams J. and Young, S. 2007a. Scaling POMDPs for spoken dialog management. IEEE Trans. on Audio, Speech, and Language Processing.

Williams, J. and Young, S. 2007b. Partially Observable Markov Decision Processes for Spoken Dialog Systems. Computer Speech and Language 21:2.