A Generic Framework for Building Dialogue Games for Language Learning: Application in the Flight Domain

Yushi Xu, Stephanie Seneff

Spoken Language Systems Group
MIT Computer Science and Artificial Intelligence Laboratory, United States
{yushixu, seneff}@csail.mit.edu

Abstract

This paper presents a generic web-based dialogue game framework, "Mercurial", for language learning. The system capabilities are demonstrated within a flight domain application. The system randomly generates a flight-booking scenario according to the current difficulty level. The student accomplishes the scenario by interacting with the system through speech. The system gives real-time feedback on the dialogue progress and scores, and summarizes the whole dialogue at the end of each scenario. The system uses generic language understanding, language generation, dialogue management, and user simulation modules to conduct the dialogue and give contextual help. Preliminary experiments yielded positive feedback from the subjects.

1. Introduction

Dialogue interaction is a natural setting to use speech communicatively, and is therefore an important and effective activity for second-language learners. Dialogue activities represent an integrated exercise for both expressing meaning and comprehending meaning. It requires correct understanding of the utterances by both dialogue partners, especially in a goal-directed conversation, in order to lead the dialogue to a successful termination point. With a flexible dialogue design, the student can have many options at each turn in the dialogue, which crucially makes the task challenging and interesting.

While the student benefits significantly from dialogue with a partner, this is a high-cost activity for both traditional language learning and computer-aided language learning (CALL). For a computer to act as a dialogue partner, the system has to not only recognize what the student has said, but also understand the meaning in the context of the dialogue task, and provide appropriate feedback. The computer plays two roles in this situation: both a party for continuing the dialogue and a tutor for language learning purposes.

Many existing language learning systems use short dialogues as part of the exercise material [1] [2], but the dialogues are scripted and the students are only allowed to speak exactly what is expected. This has the advantage of high accuracy in judging the student's performance, but it sacrifices the flexibility and attractiveness of the dialogue activity itself.

The spoken dialogue community has been researching different types of dialogue management models for more than two decades. Systems using rule-based methods [3] [4] or complex statistical models [5] [6] have proved effective, and some have been released to the public [7]. These systems, however, from the standpoint of language learning, lack the second role as a learning assistant. They are designed for native speakers, and would not give students feedback on whether the dialogue is on the right track, or give hints when the student does not know how to express a certain meaning.

Our group has been working on robust speech and language processing technologies for many years. Utilizing these technologies, we are able to build different speech-enabled language learning games that understand the student's speech [8]. This paper describes our first generic framework "Mercurial" for complex task-directed dialogue games with an abundance of language tutoring features. Mercurial combines four generic modules: language understanding, dialogue management, language generation, and user simulation, as well as a game control module that assesses the student's performance and gives real-time feedback. This paper focuses on the application in the flight-reservation domain.

The rest of this paper is organized as follow: Section 2 introduces the interface and functionality of the game system. Section 3 describes the key technologies. Section 4 describes our preliminary experiments. Section 5 gives the conclusion.

2. Game System

Figure 1 shows the interface of the flight domain game system. The Web-based system can be accessed via a URL. In this paper, we will discuss the game system mainly in a Mandarin-learning setting. However, because all the modules in the system are language independent, the game can be configured into an English-learning setting simply by switching the language model, parsing grammar, and generation rules.

The goal of the game is to book a flight itinerary according to a give scenario by talking with the system. The scenario is generated randomly according to the current level: the higher the level, the more complex the scenario. The scenario is represented as a natural paragraph, which is phrased in a number of different ways. The crucial points in the scenario are gathered to form a checklist, shown in the lower left part of the figure. As the dialogue proceeds, the items in the list will be marked with a green check or a red cross according to the student's utterances. The lower right part shows the current itinerary information. In the center of the webpage, a feedback message is given for each dialogue turn. It offers praise when the student has progressed through the dialogue correctly, or it might provide a hint for correction when the student has said anything wrong. The "repeat" buttons allow the student to hear again the system's reply either in Chinese or in English. "Help" utterances are generated based on context, using a variety of sentence patterns to provide multiple ways to express appropriate actions at the current stage.

The system generates a score for the dialogue so far, which is displayed on the screen. When the scenario is successfully accomplished, or is aborted by the student, the system summarizes the complete dialogue between the system and the student, and shows the student the detailed scores in each aspect. When enough points have been accumulated, the student will be advanced to the next level, and can then practice via a more complex scenario.

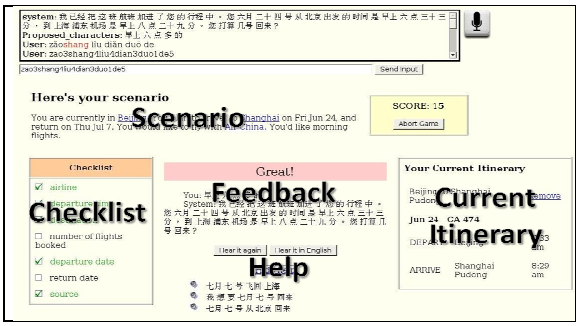

Figure 1. *Game Interface during a dialogue.*

3. Technical Components

3.1. Overview

The Mercurial system uses a client-server schema, where the client side simply provides a graphical/speech user interface and all the processing is done on the server side. The system uses the WAMI toolkit [9] to handle the web interface, including capturing and transmitting the audio waveform. The waveform is sent to the recognizer to produce an N-best hypothesis list. A control script dictates the order and parameters of the language processing and game progressing steps. The final outputs, *i.e.*, the updated webpage content and the synthetic waveform, are presented to the user via WAMI.

The system has access to a number of language- and domain-independent generic modules such as language understanding, language generation, dialogue management, and user simulation. After the appropriate parameters, such as the language for parsing and generation, and the domain for dialogue management and simulation, are specified, a dialogue game can be built by adding just one more module to handle the game progress. Figure 2 illustrates the system diagram. We will discuss these modules in more detail in the following subsections.

3.2. Scenario Generation

At the beginning of each dialogue, a scenario is shown to the student as his/her task. The scenario describes the source, destination, date, type of itinerary and other constraints that the student needs to fulfill when booking the flights. The scenario can be expressed linguistically in a number of different ways, randomly selected for each dialogue. The complexity of the scenario grows with difficulty level. As the student graduates to a higher level, new types of constraints and/or more constraints are added.

The random scenario is obtained by initializing the user simulator, which creates a user intention frame. This frame uses compact key-value representations, in which some keys have a deterministic value, such as source and destination, and others have a probability distribution over a set of values, such as preferred airlines and departure time.

Each possible key in the intention frame is assigned a difficulty level, and a scenario frame is formed by drawing the keys from the intention frame in a random order, as many as possible until the difficulty level of the scenario exceeds the specified level. The difficulty level of a scenario is defined in Equation (1), where d_{base} is the base difficulty level, d_i is the difficulty level of each key, and D is the difficulty of a

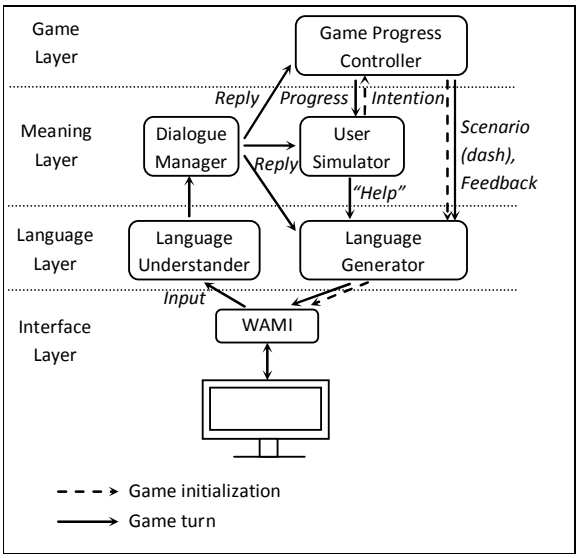

Figure 2. *The Mercurial system diagram.*

scenario. Thus, D is dominated by the highest difficulty level among all the keys, and is also affected by the number of keys.

$$D = \left\lceil \log_2(2^{d_{base}} + \sum_i 2^{d_i}) \right\rceil \quad (1)$$

After the scenario frame is formed, it is simplified so that the probabilistic values are determinized by random selection, and then converted into a natural paragraph in the desired language using the randomized language generation module.

The generation of the scenario is independent from our flight database. This will sometimes result in an unsatisfiable scenario. In this situation, when the system notices a zero-match search result during the dialogue, it will adjust the scenario by changing the constraints or dropping some constraints, and notify the student of this change via a natural message. This also increases the scenario difficulty, producing a bonus score when the scenario is successfully completed. This feature can partition the system into two separate roles: system A helps the student find some flights, and system B, which is ignorant of the flight information, constantly listens to the dialogue between the student and system A, notices what is happening, and makes appropriate comments.

3.3. Language Processing and Dialogue Management

The game has two input modalities: speech as the primary and recommended modality, and text as the back-off modality. For both modalities, an N-best list of Chinese characters is produced from the audio waveform or the typed Pinyin string. Each utterance in the list is parsed by an extended CFG parser [10] [11]. The top-most utterance with a full parse is chosen as the best input, and the parse result is converted to a key-value meaning representation, and sent to the dialogue manager.

For text inputs, students type using toned Pinyin strings. The system then expands the string into a word graph to account for possible tone errors. An N-best character list is obtained by using an n-gram language model. After parsing, the best selected character list is aligned with the original text input, and any detected tone errors will be highlighted.

We use an entity-constraint based dialogue manager [12], where a domain-dependent specification defines the entity

types and the knowledge sources (*e.g.* database) constraints. In this game, we supply the dialogue manager a specification that is exactly the same as the one that would be used in a normal flight reservation dialogue system. Statistical inference is incorporated for certain decisions. In this game, the statistical engine uses the models obtained from [12]. A simulated flight database is used for simplicity. A small number of airports and airlines are defined. Random flights are generated between each possible pair of airports at initialization.

The dialogue manager produces a reply in the meaning representation format, which is then converted into a natural sentence via a language generation module. As described in [13], generation rules specify the order of constituents, and an associated lexicon maps words into proper surface forms in the target language. In order to avoid generating the same reply pattern given a certain reply type, the language generation module is enhanced by introducing randomness. Both the rules and the lexicon can include multiple instances for each entry, which represent different ways to phrase an intended meaning. The module randomly chooses one rule or lexicon mapping from multiple instances. The combination of randomized rules and a randomized lexicon gives a fairly rich variety of different generation results.

3.4. Game Scoring and Level Assessment

After a dialogue turn is finished, the progress assessment module evaluates the student's performance, and posts a credit or deduction toward the total score. This module works at the meaning representation level, and therefore the whole system maintains the property of language independence.

The student's performance is assessed in four aspects.

(1) Sentence wellness. The student earns some points when his/her utterance goes through the language understanding component and a meaning representation is successfully obtained. The points are then proportional to the complexity of the sentence, which is currently based on sentence length.

(2) Dialogue progress. From the generated scenario frame, a scenario checklist is produced and shown to the student. Whenever a piece of information is correctly conveyed to the dialogue manager, the corresponding item gets a check, and the student earns a point. On the other hand, if the wrong information is conveyed, a point is deducted.

(3) Context wellness. The student should conform to the dialogue manager's prompts, as opposed to talking without understanding the system's speech. Whenever the student's answer does not match the dialogue manager's question, points are deducted. We do, however, allow the student to correct any information that was spoken incorrectly without penalty, regardless of the dialogue manager's prompt.

(4) Independence and scenario bonuses. At the end of a successfully completed scenario, the student is awarded a bonus score. The bonus is proportional to the scenario difficulty, which includes any scenario changes in the middle of the dialogue. It is also proportional to the student's independent performance; *i.e.*, the fewer times he/she asked for help, the more points are awarded.

According to the turn assessment, the module updates the total score, and generates a feedback message to praise the student or to alert him/her of a mistake. When the scenario is finished or aborted, a dialogue record together with a detailed score report is displayed. The student can review what has been said and what mistakes were made.

The score earned from one scenario will be accumulated to the next one. When enough points are accumulated, the student will be advanced to the next level.

3.5. Help Function through User Simulation

For a language learning system, it is very important to provide various clues when the student needs assistance. In this game system, help is provided in three ways. First, proper noun phrases appearing in the scenario are accompanied with their Chinese translation and pronunciation, and can be accessed by mouse moves and clicks. Second, the system's reply can be replayed in either Chinese or English. Third, "help" sentences are generated dynamically according to the specific scenario and the current dialogue context. The student can expand the "help" section, and see and hear the "help" sentences.

The generation of the "help" sentences is done by simulating a very cooperative user. The user simulator loads a set of conditioned rules to produce user responses in the meaning representation format. The syntax is simple yet powerful to describe the condition and various types of responses. The simulator matches the conditions against a state frame, and assigns all the responses a score. To represent high cooperativeness, the penalty for not matching the condition is set to 1, so that only the responses for which the conditions are strictly satisfied may survive.

The state frame that is sent to the user simulator contains the scenario information, the dialogue manager's reply, and the progress assessment result. The simulator is controlled by 20 rules which cover both the possible user responses from the current dialogue context and the possible ways to correct a mistake made in previous turns.

4. Preliminary Experiments

4.1. System Configuration

Some preliminary experiments were performed in a Mandarin-learning setting. The system accepts spoken Chinese input or typed Pinyin input, and speaks in Chinese. The scenario and the feedback message, however, are shown in English, to assure student understanding of the task.

We use the SUMMIT [14] speech recognizer, trained on an *n*-gram language model on a flight-domain corpus, to constrain the recognizer search space. Parsing is based on a generic Chinese grammar, augmented with a domain-specific lexicon, such as airline names and city names. The Chinese synthesizer is provided by the Chinese Academy of Science.

Five subjects, three learners and two native Chinese speakers, interacted with the system, following some verbal instructions. Afterwards, they were asked to fill out a questionnaire. Table 1 shows the average scores of the three learners' Mandarin proficiency according to the questionnaire.

Reading	Writing	Speaking	Listening
2.3	2.7	3.3	3.0

Table 1. *Average scores of self-ranked Mandarin proficiency (1-very poor, 5-native-like).*

4.2. Recognition Performance

During the development of the system, we noticed that the recognition accuracy is one of the major issues that cause system failures. The acoustic model of our recognizer is trained on native data, to encourage better pronunciation from the students. This compromises the recognition accuracy on

accented speech. However, assuming the student acts cooperatively, the scenario and the dialogue context contains information to predict the input speech. Thus, we adopted a global N-best selection strategy similar to [15].

For each hypothesis on the recognizer's output N-best list, language understanding and dialogue management are performed. The progress assessment module then evaluates the resulting dialogue state as described in Section 3.4. The hypothesis that maximizes the sum of the progress score and the context score is selected.

We collected 305 utterances from the three learners, and 134 utterances from the two native speakers. To compare the global N-best re-ranking method with conventional methods, we removed the ungrammatical utterances from the learners' utterances, as well as the native utterances that were not covered in the recognizer's language model, *i.e.*, utterances that can never be recognized correctly. This results in 278 non-native utterances, with average length of 7.1 characters, and 112 native utterances with average length of 8.7 characters.

	Native WER	Non-native WER	Native CER	Non-native CER
1-best	13.89%	15.47%	37.37%	47.21%
Top full parse	13.99%	15.37%	36.30%	45.72%
Global	12.89%	14.06%	33.45%	39.41%

Table 2. *WER and CER of different N-best selection methods.*

$$CER = \frac{\#slots_{ins} + \#slots_{del} + \#slots_{sub}}{\#slots_{ref}} \quad (2)$$

The character-based word error rates and concept error rates (CER) are shown in Table 2. The first column shows the results of using 1-best recognition output. The second column shows the results of selecting the topmost hypothesis that produces a full parse from a 10-best recognition output. The third column shows the results of using global selection on a 10-best recognition output. The CER is calculated from flattened meaning representations using Equation (2). The global selection method achieved significant improvements for both WER and CER, for both native and non-native data.

4.3. Game Survey

The subjects responded positively about the game. The average scores for interest value and helpfulness of the game on a 5-point basis are 4.3 and 4.2 respectively. All subjects would recommend the game to other learners of Chinese.

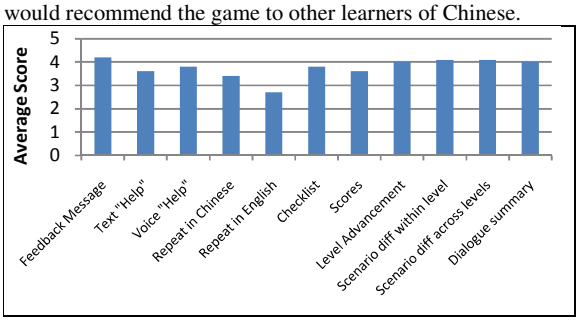

Figure 3. Average scores of helpfulness of various tutoring features.

We also surveyed the helpfulness of various features provided in the game. Figure 3 shows the result. Most features received high scores and were favored by the users.

5. Conclusions and Future Work

This paper presented a generic dialogue framework, Mercurial, demonstrated in a flight reservation domain. The framework incorporates generic language understanding, language generation, dialogue management and user simulation modules to generate random scenarios, handle conversation, and provide students instant feedback.

We conducted some preliminary experiments with three learners and two native speakers. The results indicated that using a global N-best selection strategy can significantly improve the recognition results. The subjects also gave positive feedback about the game.

In the future, we plan to set up another version of the game for learners of English. A more detailed evaluation will be conducted with a larger group of users.

6. Acknowledgements

This research is partially supported by Quanta Computers, Inc.

7. References

[1] (2008) NTU Chinese. [Online]. http://chinese.ntu.edu.tw/

[2] (2006) Active Chinese. [Online]. http://www.activechinese.com

[3] S. Seneff, "Response Planning and Generation in the Mercury Flight Reservation System," *Computer Speech and Language*, vol. 16, pp. 283-312, 2002.

[4] D. Bohus and A. I. Rudnicky, "RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda," in *Proc. Eurospeech*, Geneva, Switzerland, 2003.

[5] M. Frampton and O. Lemon, "Learning more effective dialogue strategies using limited dialogue move features," in *Proc. ACL*, Sidney, Australia, 2006, pp. 185 - 192.

[6] J. D. Williams and S. Young, "Partially observable Markov decision processes for spoken dialog systems," *Computer Speech & Language*, vol. 21, no. 2, pp. 393-422, 2007.

[7] A. W. Black, S. Burger, B. Langner, G. Parent, and M. Eskenazi, "Spoken Dialog Challenge 2010," in *Proc. SLT*, Berkeley, CA, USA, 2010.

[8] Y. Xu and S. Seneff, "Speech-Based Interactive Games for Language Learning: Reading, Translation, and Question-Answering," *International Journal of Computational Linguistics and Chinese Language Processing*, vol. 14, no. 2, 2009.

[9] A. Gruenstein, I. McGraw, and I. Badr, "The WAMI Toolkit for Developing, Deploying, and Evaluating Web-Accessible Multimodal Interfaces," in *Proc. ICMI*, Chania, Crete, Greece, 2008.

[10] S. Seneff, "TINA: A Natural Language System for Spoken Language Applications," *Computational Linguistics*, vol. 18, no. 1, pp. 61 - 86, March 1992.

[11] Y. Xu, J. Liu, and S. Seneff, "Mandarin Language Understanding in Dialogue Context," in *Proc. ISCSLP*, Kunming, China, 2008.

[12] Y. Xu and S. Seneff, "Dialogue Management Based on Entities and Constraints," in *Proc. SIGDIAL 2010*, Tokyo, Japan, 2010.

[13] L. Baptist and S. Seneff, "Genesis-II: A Versatile System for Language Generation in Conversational System Applications," in *ICSLP*, Beijing, China, 2000, pp. 271-274.

[14] J. Glass, "A probabilistic framework for segment-based speech recognition," *Computer Speech and Language*, vol. 17, no. 2-3, pp. 137-152, April/July 2003.

[15] A. Gruenstein, "Response-Based Confidence Annotation for Spoken Dialogue Systems," in *Proc. SIGDial*, Columbus, Ohio, USA, 2008.