International Workshop on Cooperative Information Agents (CIA-2001),
M. Klusch and F. Zambonelli (Eds.), *Cooperative Information Agents V,*
Lecture Notes in Artificial Intelligence, Vol. 2182, pp. 20-25, 2001.

Wizard of Oz Method for Learning Dialog Agents

Masayuki Okamoto[†], Yeonsoo Yang[†⋆], and Toru Ishida[†‡]

[†]Department of Social Informatics, Kyoto University
Yoshida Hommachi, Sakyo-ku, Kyoto, 606-8501 Japan
[†]CREST, Japan Science and Technology Corporation
{okamoto,soo}@kuis.kyoto-u.ac.jp, ishida@i.kyoto-u.ac.jp
http://www.lab7.kuis.kyoto-u.ac.jp/

Abstract. This paper describes a framework to construct interface agents with example dialogs based on the tasks by the machine learning technology. The Wizard of Oz method is used to collect example dialogs, and a finite state machine-based model is used for the dialog model. We implemented a Web-based system which includes these functions, and empirically examined the system which treats with a guide task in Kyoto through the experimental use.

1 Introduction

There are many synthetic interface agents which introduce some Web sites through a text-based dialog. For example, Jennifer in Extempo [1] sells cars virtually, and Luci in Artificial Life [2] introduces her own Web site. They play the role of guides, and prevent users from clicking all links of these sites. The number of such kind of interface agents is increasing.

However, when a designer decides to make an agent, he/she should design an internal model per domain, and he/she should implement it. It costs very much to construct each agent in different way.

In this paper, we propose a framework to construct task-oriented dialog agents from example dialogs semi-automatically. For the dialog model of each agent, we use a finite state machine (FSM). For collecting the example dialogs, we use the Wizard of Oz (WOZ) method [4] which is originally used for prototyping of dialog systems.

2 Learning Process

This section describes the learning process of the agent.

For constructing a learning dialog agent, the following elements are needed:
(a) The internal model and learning mechanism of which the agent consists.
(b) The environment and method for collecting dialog examples of good quality.

⋆ She is currently working at Toshiba Corporation.
[1] http://www.extempo.com/
[2] http://www.artificial-life.com/

We use a finite state machine for the dialog model or structure of the agent, and the WOZ method for collecting dialogs.

There are some FSM-based systems. In particular, the VERBMOBIL uses a layered architecture including FSM [1]. Our approach differs from usual approaches. We try to provide a simple method to construct systems with limited scenarios and with the assist by human, though each agent is used in a narrow domain or task.

Figure 1 shows the conceptual process of constructing agents.

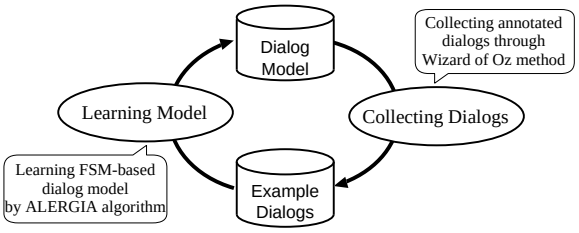


Fig. 1. Conceptual process

Finite State Machine-based Dialog Model

We consider each dialog as a series of utterances. Each utterance consists of (a) the speaker, (b) the content, and (c) the utterance tag decided based on the task design.

The dialog model is constructed by the *ALERGIA* algorithm [2], which was originally used for the grammatical inference area. We try to use it for an actual dialog systems. The algorithm works as follows.

First, a tree-formed FSM is constructed from the examples. Then, the compatibility of each pair of two states is examined. If the probability of any their suffixes are compatible, the two states are merged. Finally, a generalized FSM is constructed.

In this paper, each utterance tag means an FSM symbol, the learned FSM means a dialog model, and the original example dialogs mean the plausible sets of a user's inputs and the agent's outputs. Figure 2 shows the dialog structure.

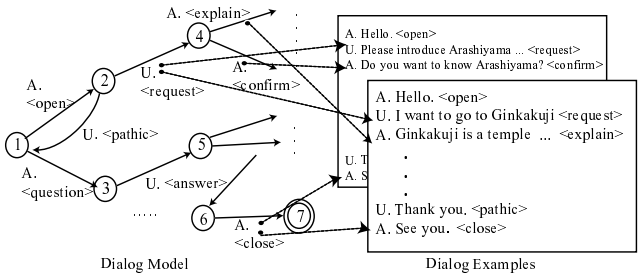


Fig. 2. Dialog model and example dialog

We introduce the keyword score of each words to recognize the user's utterance. When a user inputs an utterance, the nearest example utterance is calculated by the simple dot product of word score vectors.

Applying Wizard of Oz Method to Collect Example Dialogs
The *Wizard of Oz (WOZ)* simulating [4] is a method that a user and a person called *Wizard* who behaves as if he/she were a system talk together. It is because a human-human dialog should not be applied to human-computer dialog interfaces. There are differences in utterances used when a user thinks the partner of the communication is a computer and utterances used when he/she thinks the partner is a person [3]. This method is usually used for prototyping a system.

We apply the WOZ method to develop learning interface agents. This framework has the following two features:

Example-based development Instead of dialogs which developers *guess*, dialogs which users and the Wizard *actually talk* are used for the systems. The range of actions which a person does are limited if the situation is established clearly [6]. Therefore, the example-based approach is suited to the construction of dialog agents.

Human-assisted development The role of the Wizard is to assist the learning process as well as supplementing functions of the system. At the beginning, the system with a few dialog examples has little intelligence. As the system progresses, the role of Wizard is replaced by the system, and the system can be evolved to the real interface agent. Therefore, we consider the WOZ as the human-assisted method for learning interface agents[3].

3 WOZ-based Agent System

We implemented a support system with the mechanism described in Section 2.

The system architecture is shown as Figure 3. Each component of the architecture has the functions and contents as below:

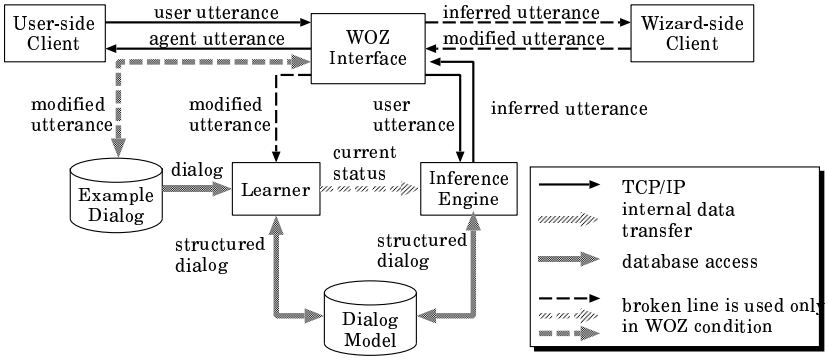
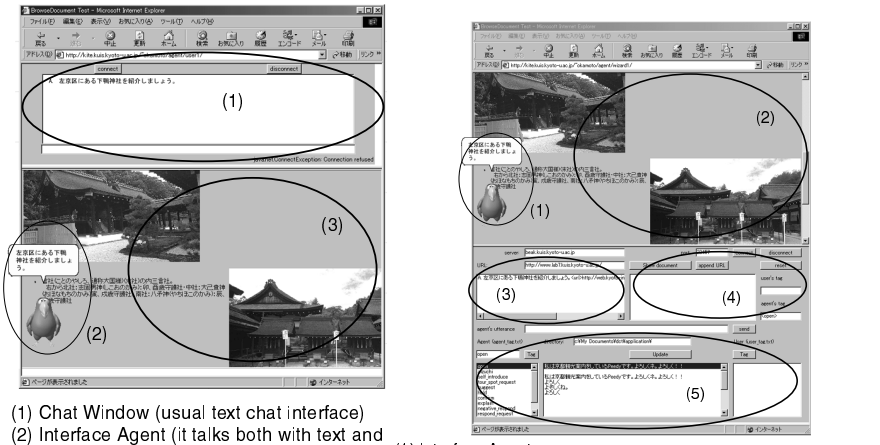


Fig. 3. Architecture of learning interface agent

User-side Client It is used by a user. The current version (Figure 4(a)) is a chat system which is expanded with an interface agent of Microsoft Agent[4]. The agent talks both with text and speech. When the agent talks about a topic, a Web page related to the dialog will be shown in the Web browser.

[3] In the WOZ condition, both the Wizard and the system play the role of the agent.
[4] http://msdn.microsoft.com/msagent/



(1) Chat Window (usual text chat interface)
(2) Interface Agent (it talks both with text and voice)
(3) Web Page Related to Contents (When a Web page is associated to the agent's utterance, the page pops up at the same time the agent speaks)

(a) User-side client

(1) Interface Agent
(2) Web Page Related to Contents (the same page the user sees)
(3) Log Window (chat log)
(4) Inferred Message Window (inferred utterances by the system)
(5) Preset Dialogs and Utterance Tags (the Wizard can input or modify the utterance)

(b) Wizard-side client

Fig. 4. Screenshot of each client

Wizard-side Client The Wizard uses a client (Figure 4(b)) which is the extended version of the User-side Client. It has two additional features. (a) The Wizard can associate a related Web page with a dialog. With this feature, the agent can explain each topic on the corresponding Web page. (b) The Wizard can select an utterance from inferred utterances by the system or preset utterances from the menu.

WOZ Interface It controls the dialog stream among each client, the Learner, and the Inference Engine. When the Wizard-side Client is disconnected, the whole system works as an individual agent system.

Example Dialog It has the collected dialogs. Each utterance is annotated by the Wizard.

Dialog Model It has the current FSM and all utterances of which the FSM consists. The FSM is constructed by the Learner, and referred by the Learner and the Inference Engine.

Learner It constructs an FSM from annotated dialogs in the Example Dialog, and updates each word score of utterances in the WOZ condition according to the utterances from the WOZ Interface.

Inference Engine It infers next utterance from an input from the user. Then, it sends the utterance to the Wizard in the WOZ condition, otherwise it sends the utterance to the user. In the WOZ condition, it also receives the current status (including current FSM state) from the Learner.

The system flow is as below:

1. When a user inputs an utterance, the Inference Engine infers the next utterances and sends them to the Wizard-side Client via the WOZ Interface. If there is not the Wizard, the 'best' candidate is sent to the user.

2. When the Wizard selects a candidate, the inferred utterance and its utterance tag are sent to the Learner and the Example Dialog.
3. If there are not any proper candidates, the Wizard has to select an utterance from the preset utterances or input one, and annotate both the user's utterance and the agent's utterance. The Wizard can add a URL to the utterance if the Wizard wants to show a Web page to the user.
4. In any cases, the utterance is spoken by the interface agent. If a Web page is associated with the utterance, the page is shown on the User-side client.

4 Empirical Evaluation : Kyoto Guide Task

This section shows an agent Peedy which introduces tour spots in Kyoto, Japan.

Setting

In this task, the agent talks with a user along to the following flow:

First, Peedy introduces itself. Then, he asks the user what he/she wants to know (e.g., temples, shrines, and so on). If there are a request and any correspoonding example dialogs, he explains the tour spot; otherwise, he suggests a tour spot. Finally, there are not any requests, he closes the dialog.

We collected 42 dialogs consists of 410 utterances by the agent and 167 utterances by users through the WOZ method. The example includes 103 tour spots. After the ALERGIA algorithm (the parameter $\alpha = 0.975$), we obtained an FSM-based model with 127 states and 380 transitions. Figure 5 shows an example dialog between the agent system with this model and a user. We observed 9 dialogs among the system with the learned dialog model and student users.

A. (means agent) Hello, I'm Kyoto tour guide, Peedy. Nice to meet you. ⟨open⟩
A. I was a guide of ... Where do you want to go sightseeing? ⟨self introduction⟩
U. (means user) Please tell me Nijo Castle. ⟨request TS⟩
A. Nijo Castle is ... ⟨explain TS⟩
(A Web page related to Nijo Castle appears in user's browser)
U. I see. ⟨response⟩
A. Do you want to go to another place? ⟨lead to another TS⟩
U. I want to buy my souvenir. ⟨request TS⟩
A. Can I introduce a famous shrine in Kyoto? ⟨suggest TS⟩
U. Yes. ⟨request TS⟩
A. I'll introduce Kamo-Betsurai shrine in Kita-ku. ... ⟨explain TS⟩
U. I see. ⟨response⟩
.....
A. How about today's guide? Is it good? ⟨lead to closing⟩
U. Yes. ⟨accept closing⟩
A. See you. ⟨close⟩

Fig. 5. Example dialog (the original dialog is in Japanese)

Lessons Learned

We found the following empirical result about our system from the actual use:

- The task and domain
  There were many tour spots which are not in the example learning set. It means the domain is larger than what we assumed.
- The learning algorithm and inference process
  There were two kinds of users. The first kind of persons requested different

tour spots one after another, and they often input only noun words as using a search engine. The other kind of persons replied to the agent's each answer explanatorily. Some dialogs resulted in complete failures.
When the agent spoke a wrong utterance, some users asked the agent again. It means they could not find what the agent did not understand, e.g., the agent did not know the spot, or the spot is known in another name.

5 Conclusion

In this paper, we proposed a framework to construct a learning dialog agent from example dialogs. For the dialog model of the agent, we used the FSM constructed by the ALERGIA algorithm. For collecting the example dialogs, we use WOZ method. We implemented a Web-based system and empirically examined the system which treats with a guide task in Kyoto through the experimental use.

Future work includes extending the mechanism to improve dialog management, and to treat with much knowledge and more difficult situations. Many dialog failures are from users' embarrassment that the user could not suppose the agent's internal state. The next version needs to show their status more than the current version. In our example-based approach, all knowledge is in the example. When we design agents for wider domains, it is necessary to treat with the knowledge and example dialogs independently.

We also consider that the tour-guide agent in Kyoto will be applicable in Digital City Kyoto [5].

Acknowledgement

We would like to thank associate professor Yasuhiko Kitamura at Osaka City University for his devoted advice. The work is partially supported by the New Energy and Industrial Technology Development Organization for Senior Support System Project, by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (A), 11358004, 1999, and by the Japan Science and Technology Corporation for Core Research for Evolutional Science and Technology Digital City Project.

References

1. J. Alexandersson, E. Maier and N. Reithinger, "A Robust and Efficient Three-Layered Dialogue Component for a Speech-to-Speech Translation System," *Proc. EACL-95*, pp. 188–193, 1995.
2. R. C. Carrasco and J. Oncina, "Learning Stochastic Regular Grammars by Means of a State Merging Method," *Proc. ICGI-94*, pp. 139–152, Springer-Verlag, 1994.
3. J. M. Carroll and A. P. Aaronson, "Learning by Doing with Simulated Intelligent Help," *Communications of the ACM*, Vol. 31, No. 9, pp. 1064–1079, 1988.
4. N. M. Fraser and G. N. Gilbert, "Simulating Speech Systems," *Computer Speech and Language*, Vol. 5, No. 1, pp. 81–99, 1991.
5. T. Ishida, J. Akahani, K. Hiramatsu, K. Isbister, S. Lisowski, H. Nakanishi, M. Okamoto, Y. Miyazaki and K. Tsutsuguchi, "Digital City Kyoto: Towards A Social Information Infrastructure," *Proc. CIA-99*, pp. 23–35, Springer-Verlag, 1999.
6. B. Reeves and C. Nass, *The Media Equation*, Cambridge University Press, 1996.