IEEJ TRANSACTIONS ON ELECTRICAL AND ELECTRONIC ENGINEERING
IEEJ Trans 2015; **10**: 404–410
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI:10.1002/tee.22100

Paper

TFSM-based Dialogue Management Model Framework for Affective Dialogue Systems

Fuji Ren[*,**a], Member
Yu Wang[**], Non-member
Changqin Quan[**], Member

A new dialogue management model for affective dialogue system, which aims to provide a service of information inquiry and affective interaction, is proposed in this paper. First, we construct two finite state machines (TFSM) to model the user and the system, respectively, and simulate the dialogue process as an information exchange between the two state machines. All possible state transitions in dialogue and its probabilities of the user are summarized as a user model, which is helpful for the system to inference and predict the user's internal states. Second, we further discuss the implementation methods of information inquiry and emotional response modules. Finally, we employ the return function of partially observable Markov decision processes (POMDP) model to analyze and evaluate the TFSM-based dialogue management model. The experimental results not only show the relationships between the average returns, recognition error rates, and state transition probabilities but also confirm that our TFSM-based dialogue management model outperforms the conventional FSM model. © 2015 Institute of Electrical Engineers of Japan. Published by John Wiley & Sons, Inc.

Keywords: dialogue management model, affective dialogue system, finite state machine, affective computing

Received 23 April 2014; Revised 25 November 2014

1. Introduction

Human–computer interaction has been of wide concern and has become a hot research topic in the last decades. A spoken dialogue system (SDS) is an intelligent interaction system that is designed to provide fast and convenient service for users through natural conversation. Supported by a series of projects in the US and the EU, SDS has been designed and developed since the 1990s [1,2] mainly to provide information inquiry service. Generally speaking, an SDS is usually composed of automatic speech recognition, natural language understanding, dialogue management (DM), natural language generation, and text to speech conversion. The main framework of an SDS is shown in Fig. 1. The DM module plays a key control role in an SDS. It recognizes the user's intention and outputs the system responses. To date, there are many models proposed to design the DM module [3], such as the finite state machine (FSM) [4,5], frame-based or slot-filling [6], example-based [7], partially observable Markov decision processes (POMDP) [8–10], etc. Each of these DM models has its own advantages and disadvantages. In general, the FSM model has a clear structure and is easy to develop to effectively control the dialogue process, but it is not suitable for a complex dialogue task. The advantage of the frame-based model is that it can handle more flexible inputs, but it usually leads to the dialogue unnaturally. The POMDP model is a very popular method in the theoretical studies [11–13] currently, which has been proven, in theory, to be a good model to deal with uncertain problems in speech recognition and language understanding [9,13,14]. An additional advantage of this model is that it can be applied to more fields by factoring its state space [15].

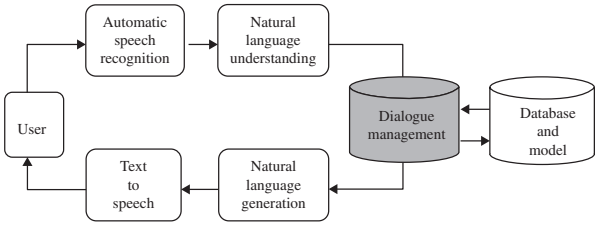

Fig. 1. Architecture of spoken dialogue system

However, it also has many defects: e.g. it cannot handle multitopic tasks, and the tagging and training corpus are expensive and time consuming, etc.

Affective computing was first proposed by Picard [16], which aims to give a computer the ability to perceive, recognize, and understand human emotions and to make an intelligent, sensitive, friendly response to human emotions. Currently, it is generally agreed that emotion is a part of intelligence. So how to endow the machine with the ability of emotional interaction with humans has become an important and challenging subject in intelligent human–computer interaction [17]. To date, most studies in affective computing are focused on detecting and recognizing emotional information with different modalities [18,19], such as speech, facial expression, posture, text, physiological information [20–22], etc. Affective dialogue system can be viewed as an enhanced version of SDS that incorporates emotion recognition, emotional interaction, as well as emotion generation and expression (Fig. 2). It focuses not only on meeting the user's needs but also in responding to the user's emotions. Designing and developing affective dialogue systems has aroused much interest [23,24]. Most research in this field so far has focused on speech emotion recognition [25–27] and robotic emotional expression generation [28]. However, these studies are necessary but not sufficient to build an available system. A new kind of DM mechanism integrates emotion information, called affective

[a] Correspondence to: Fuji Ren. E-mail: ren@is.tokushima-u.ac.jp

[*] Institute of Technology and Science, The University of Tokushima 2-1, Minamijyosanjima-cho, Tokushima 770-8506, Japan

[**] School of Computer and information, Hefei University of Technology, Hefei 230009, China

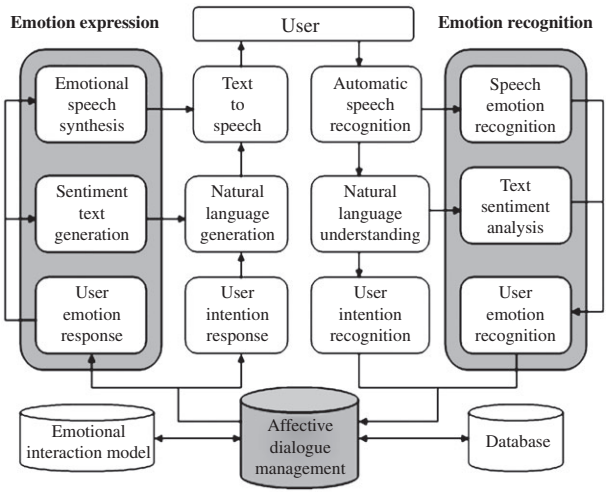

Fig. 2. Architecture of affective dialogue system

dialogue management, is indispensable to connect the modules of emotion recognition and emotion expression. A good affective dialogue management model should not only incorporate an emotional interaction method into various DM models but also be portable and easy to be developed for a given task. Compared to numerous researches in the DM field, the studies on affective dialogue management are very rare. Bui [29,30] proposed an affective dialogue management model based on a factored POMDP method, where the user's state is factored into goal, action, and affective state. Wang [31] improved the factored model by further dividing the system's action into two parts of emotional coping and goal response, where the former is to cope with user's emotion and the latter is to meet the needs. However, many parameters are handcrafted in the above two models, so it is unclear how to estimate the state transition probabilities or user model in practice.

In this paper, we try to design a simple and practical DM model for affective dialogue systems by using the useful method of FSM. There are two main differences between our model framework and the conventional FSM models. First, we construct two finite state machines (TFSM) to model the user and the system, respectively, and simulate the dialogue process as an information exchange between the two FSMs. In fact, a separate FSM model for the user is beneficial for the system to predict the user's intentions. Second, our proposed model aims to not only offer a service of information inquiry but also respond to the user's emotional state after acquiring the query results. An emotional expression state and an emotional response state are added in the two FSM model for realizing affective interaction. We also discuss the implementation methods of the proposed model, including task-based information inquiry and sub-FSM-based affective interaction. Moreover, we adopt a new evaluation method instead of the traditional task completion rate, namely the return function, to evaluate the system performance of our TFSM-based affective DM model.

The rest of this paper is organized as follows: Section 2 describes the framework of the TFSM-based affective dialogue management model, and introduces the interaction flow and user model in detail. Section 3 discusses the implementation methods of information inquiry and affective interaction module. In Section 4, we analyze the influence of different probabilities on the system performance, and compare the TFSM model to conventional FSM model. We conclude our work in Section 5.

2. TFSM-based Model Framework

Formally, a FSM model is defined as a 5-tuple

$$M = \langle S, s_0, \Sigma, T, F \rangle \tag{1}$$

Table I. Possible state transitions of the user

Current state	Next state	State transition	T_i
Initial	*Query*	*Initial*→*Query*	T_0
Query	*Query*	*Query*→*Query*	T_1
	Express	*Query*→*Express*	T_2
	End	*Query*→*End*	T_3
	Confirm	*Query*→*Confirm*	T_4
Confirm	*Query*	*Confirm*→*Query*	T_5
	Express	*Confirm*→*Express*	T_6
	End	*Confirm*→*End*	T_7
	Confirm	*Confirm*→*Confirm*	T_8
Express	*Query*	*Express*→*Query*	T_9
	Express	*Express*→*Express*	T_{10}
	End	*Express*→*End*	T_{11}
End	*End*	*End* →*End*	T_{12}

where S is the finite set of states, $s_0 \in S$ is the initial state, Σ is the finite set of inputs, $T : S \times \Sigma \to S$ is the state transition function, $T(s, a) = s'(s, s' \in S, a \in \Sigma)$ expresses the transfer from state s to s' after an input a, and $F \subseteq S$ is the set of end states.

In general, conventional FSM-based DM model only views the system as a finite automata and does not take into account the context of the user's states. In fact, the transfer of user's intentions into a dialogue can also be modeled as an FSM, and the system's FSM model should be designed according to it.

2.1. Constructed user model and system model

Suppose we want to construct an SDS that can supply both information inquiry and emotional interaction. In this case, the state set of user's intentions should at least include *Initial, Query, and Emotional expression* (hereafter called *Express*), and *End*, where intentions in the state *Query* and *Express* are the querying information and expressing emotions, respectively. Moreover, considering that the system would ask to confirm some incomplete inputs, a *Confirm* state should be added to the state set.

For simplicity, we can assume that a user will always start a conversation in the *Initial* state, then turn into the *Query* state, and finish the dialogue in the *End* state. Besides that, a free user may take several different transitions to suit his or her personal needs in other three states. To be clear, we list and number all possible state transitions in Table I. Furthermore, we establish the user's FSM model (Fig. 3, Left) to simulate the user's state transitions in a dialogue.

Corresponding to the user's FSM model, we build the system's FSM model accordingly (Fig. 3, Right). It also has five states: *Initial*, *Answer*, *Ask*, *Emotional response* (or *Respond*), and *End.* Generally, the system will provide a query result to user in the *Answer* state. The *Ask* state makes the system to have a right to ask questions in dialogue and thus achieve a mixed initiative effect. In the *Respond* state, the system would make a reasonable emotional response to respond to the user's emotion effectively based on the current emotion recognition results, which makes the conversation more intelligent and harmonious.

2.2. Interaction framework and flow

Based on the above two FSM models, we propose a TFSM-based interaction model framework to simulate the dialogue process between the user and the system, as shown in Fig. 3) in which the solid arrows show the state transitions within the two FSM models, while the dashed arrows represent the outputs. This domain-independent framework includes two modules: *Information inquiry* and *Affective interaction,* which not only make the structure and function of the system clearer but also facilitate the design and development of each module. In our TFSM-based interaction model, the output sets of the user and system are also the input sets of each other, respectively, which are given in Table II. It

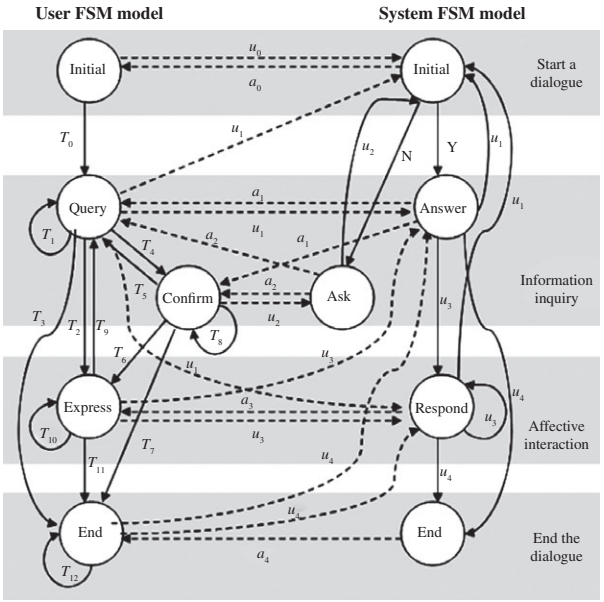

Fig. 3. TFSM-based interaction model framework

Table II. Output sets of the user and system

i	Outputs of the user $O_U = \{u_i\}$	Outputs of the system $A_S = \{a_i\}$
0	Starting dialogue	Greetings
1	Query messages	Query results
2	Confirm messages	Missing slot values
3	Emotional expression messages	Emotional response actions
4	Ending messages	Ending actions

should be noted that the outputs u_i and a_i only represent the type of the current message; its details will be updated constantly in each round of the dialogue.

The dialogue flow is described as follows:

1. In the *Initial* state, the user would send a starting message u_0 to start a dialogue; then a greeting action a_0 from the system would be fed back to the user, which can make the user turn into the *Query* state by T_0.
2. In the *Query* state, the user would send a query message u_1 to the system (*Initial*), which would judge whether the message is sufficient or not. If u_1 is sufficient (Y), the system would turn into the *Answer* state and give a query result a_1 to the user; otherwise (N) it would move into the *Ask* state and send a question a_2 about the missing slot values.
   - The query result a_1 would lead to three possible state transitions of the user: still be in the *Query* state to query another information by T_1, or move into the *Express* state by T_2, or turn into the *End* state to finish the dialogue by T_3.
   - The question $a_2$would lead the user transfer to the *Confirm* state by T_4.
3. In the *Confirm* state, the user would send a confirm message u_2 to the system (*Ask*), and then the system would be back to the *Initial* state for a new round of judgment on u_2. If u_2 is sufficient, the system would give a query result a_1; otherwise it will send another question a_2 again.
   - There are three possible transitions after the user receives a_1 under the *Confirm* state: turn into the *Query* state by T_5, turn into the *Express* state by T_6, or move into the *End* state by T_7.
   - The user will still be in the *Confirm* state by T_8 after receiving another question a_2, and continue to send a confirm message u_2 to the system (*Ask*), and restart a new round of *Ask–Confirm* cycle.
4. In the *Express* state, the user would send an emotional expression message u_3 to the system (*Answer*), which would turn to the *Respond* state and send an emotional response a_3. There are also three possible state transitions after the user receive a_3 under the *Express* state:
   - it may be back to the *Query* state again by T_9, and send a new query message u_1 to the system (*Respond*); then the system would be back to the *Initial* state for a new round of judgment;
   - it may still be in the *Express* state by T_{10}, send another emotional message u_3, and restart a new round of *Express–Respond*;
   - it may be turn into the *End* state directly by T_{11}.
5. In the *End* state, the user would send an ending message u_4 to the system (*Answer* or *Respond*); the system would turn into the *End* state and send an ending action a_4 to finish the dialogue. If the system sends other wrong messages, we can assume that the user would no longer reply and also still be in the *End* state by T_{12}.

According to this dialogue flow, we can get a simple user model that manifests by the state transition function as follows.

$$P(T_i) = P(s'|s,a) = \begin{cases} 1 & s = s_0,\ a \neq a_0,\ s' = s_0, \\ 1 & T_0 : s = s_0,\ a = a_0,\ s' = s_1, \\ P(T_1) & T_1 : s = s_1,\ a = a_1,\ s' = s_1, \\ P(T_2) & T_2 : s = s_1,\ a = a_1,\ s' = s_3, \\ P(T_3) & T_3 : s = s_1,\ a = a_1,\ s' = s_4, \\ 1 & T_4 : s = s_1,\ a = a_2,\ s' = s_2, \\ 1 & s = s_1,\ a \neq a_1,\ a_2,\ s' = s_1, \\ P(T_5) & T_5 : s = s_2,\ a = a_1,\ s' = s_1, \\ P(T_6) & T_6 : s = s_2,\ a = a_1,\ s' = s_3, \\ P(T_7) & T_7 : s = s_2,\ a = a_1,\ s' = s_4, \\ 1 & T_8 : s = s_2,\ a = a_2,\ s' = s_2, \\ 1 & s = s_2,\ a \neq a_1,\ a_2,\ s' = s_1, \\ P(T_9) & T_9 : s = s_3,\ a = a_3,\ s' = s_1, \\ P(T_{10}) & T_{10} : s = s_3,\ a = a_3,\ s' = s_3, \\ P(T_{11}) & T_{11} : s = s_3,\ a = a_3,\ s' = s_4, \\ 1 & s = s_3,\ a \neq a_3,\ s' = s_4, \\ 1 & T_{12} : s = s_4,\ a \in A,\ s' = s_4, \\ 0 & otherwise \end{cases} \tag{2}$$

where $s_0 = Initial$, $s_1 = Query$, $s_2 = Confirm$, $s_3 = Express$, $s_4 = End$. $P(T_i)$ defines the probabilities of state transitions $T_i, i = 0, \cdots, 12$, and $P(T_0) = P(T_4) = P(T_8) = P(T_{12}) = 1$, $P(T_1) + P(T_2) + P(T_3) = 1, P(T_5) + P(T_6) + P(T_7) = 1, P(T_9) + P(T_{10}) + P(T_{11}) = 1$.

The user model serves not only to recognize the user's current state s but also to predict the next state s'. That is, after receiving the user's output u_i, the system can identify its type base both on its linguistic content and the user model. Moreover, different users will always have different transition probabilities. We will further discuss its effect on the return function in Section 4.

3. Implementation Methods of Affective Dialogue Management

With the domain-independent interaction model framework for affective dialogue management in Section 2, we further discuss the implementation methods of information inquiry and affective interaction modules in this section.

3.1. Information inquiry In a single-topic dialogue system, whether the query message is sufficient or not can be judged by the slot-filling DM method; that is, the system will ask

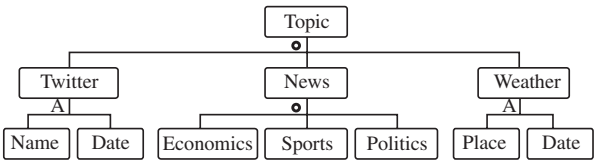

Fig. 4. Task-based DM model

Table III. Emotional responses of the system

i	User emotion states e_i	System emotional responses A_i
1	Happy	Show joy and pleasure
2	Angry	Feel nervous and scared
3	Sad	Give comfort and encouragement
4	Surprise	Be curious and confusion

Table IV. Sub-FSM affective interaction model

$T(A_i, e_i)$	e_1	e_2	e_3	e_4
A_0	A_1	A_2	A_3	A_4
A_1	A_1	A_2	$\{A_3, A_4\}$	A_1
A_2	$\{A_1, A_4\}$	A_4	A_3	A_1
A_3	A_1	$\{A_2, A_4\}$	A_3	A_1
A_4	$\{A_1, A_4\}$	A_2	$\{A_3, A_4\}$	A_1

the user constantly for each slot value and give a feedback until every slot values has been filled. For example, a query message on weather should include two slots: place and date. A multitopic dialogue system will include such topics as weather, news, twitter, and so on. A task-based DM model by an AND/OR (A/O) tree can be employed to realize the module of information inquiry (Fig. 4). Specifically, if the user's intention is to know about some news, he or she only needs to give a category such as economic, political, or sport, while the name and date both must be provided if the user wants to know about someone's twitter. In this case, a method of keyword matching can be utilized to determine the topic and slot from the user's outputs. If the slot values are incomplete, the system would turn into the *Ask* state to ask the missing slots.

3.2. Affective interaction What responses (expressions, words, actions, etc) are appropriate that the system should take to respond to user's emotional state in the *Respond* state? A mapping between the user's emotional states and the sets of system responses can be designed for this module. Specifically, for a given user emotion, we can establish a corresponding set, which is composed of some similar or common emotional responses, and then select any one of them to respond to user's emotion, as shown in Table III below.

For several rounds of human–computer affective interaction, we can model the response set $A_i (i = 1, 2, 3, 4)$ as a state, and take the user's emotion states e_i as the new transfer conditions, and then build a sub-FSM model to describe the affective interactions under our TFSM-based framework, (see Table IV). A_0 is the initial state, $T(A_i, e_i)$ indicates what transition the system will take after receiving an input e_i under the state A_i, and $\{A_i, A_j\}$ denotes the system will select the responses A_i or A_j randomly. For example, if the system has three possible state transitions in the state A_3, turn to the state A_1 if the user's next emotion state is *Happy* or *Surprise*, and turn to A_2 or A_4 if the input is *Angry*, and be still in the state A_3 if the next input is *Sad*.

4. Experimental Results and Discussions

Most SDSs based on conventional FSM model usually evaluate their system's performance by the task completion rate. However, this approach has two weaknesses: First, whether the task has been completed or not usually comes from a subjective judgment of the experimenters. Second, the task completion rate is a synthetic evaluation index of the system; it depends more on the capability of speech and language recognition modules, instead of a good dialogue management model.

Williams [11,15] provided a principled way for handcrafted DM policies that is represented as finite state controller to be compared directly with POMDP solutions. In this section, we also employ the return function to analyze and evaluate our TFSM-based DM model.

4.1. POMDP solutions Formally, a POMDP model is defined as a 8-tuple

$$\langle S, A, T, O, Z, R, \mathbf{b}_0, \gamma \rangle \tag{3}$$

where S is the state set of user, A is the action set of system, T is the transition function $T(s, a, s') = P(s'|s, a)$, O is the observation set, $Z(a, s', o') = P(o'|a, s')$ is the observation function, and $R(s, a)$ is the immediate reward.

In POMDP model, the user's current state is unobserved, and the probability distribution of each state is called a belief state. The initial belief state is always denoted by $\mathbf{b}_0$, and the next belief state $\mathbf{b}'$ will be updated by the following formula [11]:

$$\begin{aligned} b'(s') &= b_a^{o'}(s') = P(s'|\mathbf{b}, a, o') \\ &= \underbrace{k \cdot P(o'|a, s')}_{Z} \sum_{s \in S} \underbrace{P(s'|s, a) b(s)}_{T}, \end{aligned} \tag{4}$$

where k is a normalization factor. The immediate reward for the system to take action a under belief state $\mathbf{b}$ is calculated by

$$R(\mathbf{b}, a) = \sum_{s \in S} R(s, a) b(s) \tag{5}$$

The goal of the system is to choose actions that fulfill its task as well as possible, i.e. to maximize the cumulative, infinite-horizon, discounted reward, which is called the *return* function: that is

$$\sum_{t=0}^{\infty} \gamma^t R(\mathbf{b_t}, a_t) = \sum_{t=0}^{\infty} \gamma^t \sum_{s \in S} R(s, a_t) b_t(s), 0 < \gamma < 1, \tag{6}$$

where γ is a discount factor.

To facilitate the understanding and comparison, we rewrite the parameters of our model in Section 2 according to (3), where the observation function and immediate reward are handcrafted based on the practical background and some assumptions.

- The state set of the user $S = \{s_0, s_1, s_2, s_3, s_4\}$;
- The action set of the system $A = A_S = \{a_0, a_1, a_2, a_3, a_4\}$;
- The state transition function T of the user has been given in (2);
- The observation set of the system $O = O_U = \{u_0, u_1, u_2, u_3, u_4\}$;
- The observation function can be defines as follows:

$$P(o'|a, s') \approx P(o'|s') = P(o|s) = \begin{cases} 1 - p_0 & s = s_0, \ o = u_0, \\ \frac{p_0}{|S|-1} & s = s_0, \ o \neq u_0, \\ 1 - p_1 & s = s_1, \ o = u_1, \\ \frac{p_1}{|S|-1} & s = s_1, \ o \neq u_1, \\ 1 - p_1 & s = s_2, \ o = u_2, \\ \frac{p_1}{|S|-1} & s = s_2, \ o \neq u_2, \\ 1 - p_2 & s = s_3, \ o = u_3, \\ \frac{p_2}{|S|-1} & s = s_3, \ o \neq u_3, \\ 1 - p_3 & s = s_4, \ o = u_4, \\ \frac{p_3}{|S|-1} & s = s_4, \ o \neq u_4, \\ 0 & otherwise \end{cases} \tag{7}$$

Table V. The immediate rewards

$R(s,a)$	a_0	a_1	a_2	a_3	a_4
Initial	+2	−2	−2	−2	−3
Query	−2	+2	+1	−1	−3
Confirm	−2	+2	+1	−1	−3
Express	−2	−1	−1	+3	−3
End	−2	−2	−2	−2	+3

where

$$p_0 = 1 - p(u_0|s_0), \quad p_3 = 1 - p(u_4|s_4) \tag{8}$$

denote the recognition error rates of the system on user's outputs u_0 and u_4 (starting and ending messages), respectively, which can also be interpreted as the recognition error rates of the *Initial* and *End* state, respectively.

$$p_1 = 1 - p(u_1|s_1) = 1 - p(u_2|s_2) \tag{9}$$

is the average recognition error rate of user's outputs u_1 and u_2 (query and confirm messages); it can also be interpreted as the average recognition error rates of the *Information Inquiry* module.

$$p_2 = 1 - p(u_3|s_3) \tag{10}$$

is the recognition error rate of user's outputs u_3 (emotional expression messages); it can also be interpreted as the recognition error rates of the *Affective Interaction* module.

- The immediate reward $R(s,a)$ is defined in Table V. It reflects the user satisfaction for system actions. One of our main principles for determining these rewards is that an appropriate action will get a positive reward while the rest are negative.
- The initial belief state $\mathbf{b}_0$ of the system is $(1,0,0,0,0)$, i.e. the user is in the *Initial* state. A discount factor $\gamma = 0.99$ is used for all experiments.

4.2. Experimental results Our experiment is based on the randomized point-based value iteration algorithm *Perseus* [32], which is the most widely used approximate POMDP algorithm.

The first experiment demonstrates the relationship between the system performance and the observation function (recognition error rates). In this experiment, we assume that all the four error rates p_i, $0 \le i \le 3$ are equal, which can be viewed as the average error rate of the system to identify the various messages. The user model $P(T_i)$, $0 \le i \le 12$ is set in turn as 1, 0.25, 0.5, 0.25, 1, 0.25, 0.5, 0.25, 1, 0.25, 0.5, 0.25, and 1. The result of this experiment is shown in Fig. 5, from which we can find that the average returns decreases with increasing average error rates p_i. It also confirms

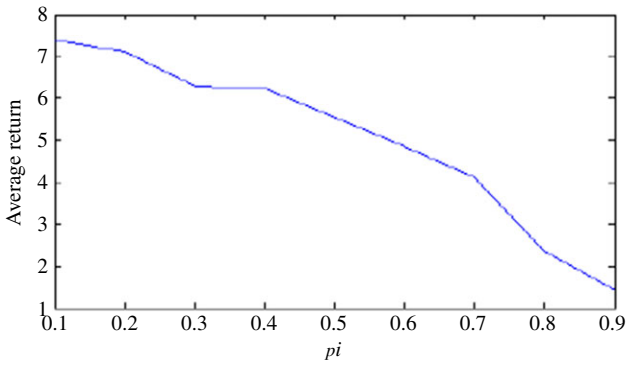

Fig. 5. Average returns vs average recognition error rates p_i

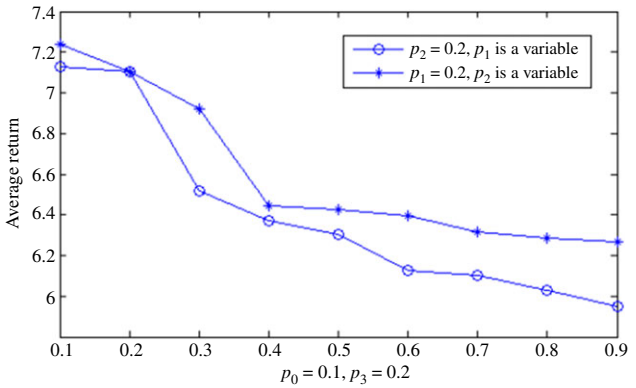

Fig. 6. Average returns vs recognition error rate p_1 and p_2

the fact that a system with a lower average recognition error rate will always perform better.

In fact, it is easier to recognize the user's *Initial* and *End* states, i.e. the error rates p_0 and p_3 are usually smaller than the other two and less affected by system. Based on this consideration, we design the second experiment to further analyze and compare the effect of error rates p_1 and p_2 on the system performance. In this experiment, the parameters p_0, p_3 are set to 0.1and 0.2, respectively; the user model remains the same. From Fig. 6, we can see that the average returns decrease with the increasing rates p_1 and p_2. By comparing the two graphs, we can also see that the returns in the case $p_1 = 0.2$ are always bigger than the others. This provides evidence that a good performance of the system should depend more on the recognition error rate p_1 than p_2.

Different users will always have different user models or state transition probabilities. The purpose of the third experiment is to study the influence of different user models on the system performance. Based on the intention, we change the transition probabilities 0.25, 0.5, 0.25 (user 1) in the previous experiments to 0.05, 0.9, 0.05 (user 2), and $1/3, 1/3, 1/3$(user 3), respectively. Figure 7 shows the returns all decrease with the increasing $p_1 = p_2$ in these three cases. For a given $p_1(p_2)$, we can find that the closer the user's state transitions, the larger the average returns. In other words, if the user tends to select a relatively fixed state transition (user 2), then the system that is expected to get a good performance needs to have a lower error rate.

The fourth experiment is designed to compare our TFSM model with conventional FSM-based DM model. In fact, there is no user model in a conventional FSM model, but only a system's FSM model. So the user's current state and state transitions in a dialogue are unable to be estimateed. We can assume that the user's state transition functions are equally possible for all states; that is

$$T(s,a,s') = P(s'|s,a) = P(s') = P(s) = \frac{1}{|S|} \tag{11}$$

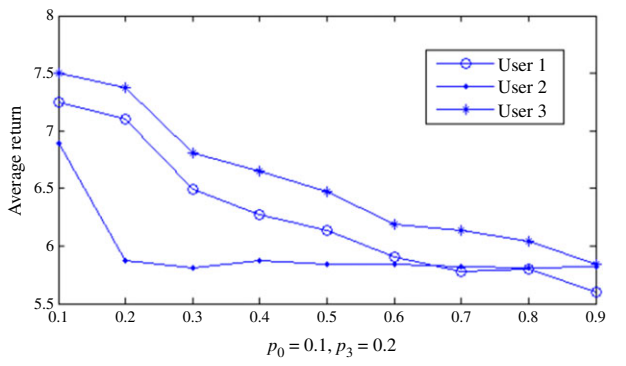

Fig. 7. Average returns vs $(p_1 = p_2)$ for three user models

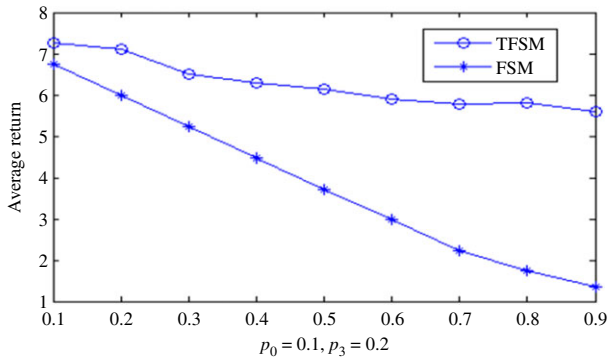

Fig. 8. Average returns vs ($p_1 = p_2$) for TFSM and FSM models

Since there are five states, the transitions of conventional FSM are equal to 0.2. We also set the user model of our TFSM model same as in the first two experiments. Figure 8 shows the average returns of the two models, from which we can find that the TFSM-based DM model is always outperforms the FSM model for all values of recognition error rates $p_1 = p_2$. It indicates that a user model can effectively improve the system performance by increasing the identification capability of the user states.

4.3. Discussions In practice, the user model or state transition probabilities can be acquired by using the method of user simulation technology WOZ (Wizard of OZ). Specifically, we can simulate the dialogue process and collect the dialogue corpuses by selecting two groups of speakers to play the role of user and system, respectively. The speakers playing the role of user are unconstrained and freely transfer among the user states, and the speakers of system are limited and must transfer by the rules. All the simulated dialogue corpuses will be classified and tagged for training the user model.

With the help of the user model and the return function, POMDP model can automatically generate an optimal dialogue policy which has been demonstrated to be suited for robust dialogue management with a small state space. However, it is still unclear how to obtain effective dialogue corpuses and train a reasonable user state transition model for a given task. Our TFSM-based model interaction framework also provides a useful tool to solve this problem.

5. Conclusions

In this paper, we proposed a new TFSM-based interaction model framework for affective dialogue management. A separate FSM model for the user is beneficial for the system to predict the user's internal states. The proposed model concentrates on creating a practical affective dialogue system that can not only meet the user's needs but also respond to the user's emotion. We further discussed the implementation methods of information inquiry and affective interaction module, including embedded a sub-FSM model to design the emotional response transitions. Moreover, we utilized the return function of the POMDP model to evaluate the performance of our TFSM-based DM model. The experiment results demonstrated the relationship between the performance of system and recognition error rates, as well as the state transition functions, and also confirmed that our TFSM-based dialogue management model outperforms the conventional FSM model.

Our future work mainly includes two parts. The first part is to develop a simple affective dialogue system for our humanoid robot based on the TFSM-based interaction model. The second is to continue to improve and evaluate our TFSM-based affective dialogue management model, with emphasis on the emotional interaction module.

Acknowledgments

This work was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (A), 22240021, the National High-Tech Research & Development Program of China (863 Program, Grant No. 2012AA011103), and the National Natural Science Foundation of China (Grant No. 61432004). ANHUI Province Key Laboratory of Affective Computing & Advanced Intelligent Machine (Grant No. ACAIM150107).

References

(1) Walker MA, Rudnicky AI, Aberdeen JS, Bratt EO, Garofolo JS, Hastie HW, Le AN, Pellom BL, Potamianos A, Passonneau RJ, Prasad R, Roukos S, Sanders GA, Seneff S, Stallard D, Whittaker S. DARPA communicator evaluation: progress from 2000 to 2001. *INTERSPEECH*, 2002; 273–276.
(2) Mariani J, Lamel L. An overview of EU programs related to conversational/interactive systems. *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, 1998; 247–253.
(3) Lee C, Jung S, Kim K, Lee D, Lee GG. Recent approaches to dialog management for spoken dialog systems. *Journal of Computing Science and Engineering* 2010; **4**(**1**):1–22.
(4) Abe K, Kurokawa K, Taketa K, Ohno S, Fujisaki H. A new method for dialogue management in an intelligent system for information retrieval. *Proceedings of the 6th International Conference on Spoken Language Processing*, 2000; 118–121.
(5) Lee C, Cha YS, Kuc TY. Implementation of dialogue system for intelligent service robots. *International Conference on Control, Automation and Systems*, IEEE, 2008; 2038–2042.
(6) Oh HJ, Lee CH, Jang MG, Lee KY. An intelligent TV interface based on statistical dialogue management. *IEEE Transactions on Consumer Electronics* 2007; **53**(**4**):1602–1607.
(7) Lee C, Jung S, Kim S, Lee GG. Example-based dialog modeling for practical multi-domain dialog system. *Speech Communication* 2009; **51**(**5**):466–484.
(8) Levin E, Pieraccini R, Eckert W. A stochastic model of human–machine interaction for learning dialog strategies. *IEEE Transactions on Speech and Audio Processing* 2000; **8**:11–23.
(9) Roy N, Pineau J, Thrun S. Spoken dialogue management using probabilistic reasoning. *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2000; 93–100.
(10) Young S, Gasic M, Thomson B, Williams JD. POMDP-based statistical spoken dialog systems: a review. *Proceedings of the IEEE* 2013; **101**(**5**):1160–1179.
(11) Williams JD, Young S. Partially observable Markov decision processes for spoken dialogue systems. *Computer Speech and Language* 2007; **21**(**2**):393–422.
(12) Williams JD, Young S. Scaling POMDPs for spoken dialog management. *IEEE Transactions on Audio, Speech, and Language Processing* 2007; **15**(**7**):2116–2129.
(13) Young S, Gasić M, Keizer S, Mairesse F, Schatzmann J, Thomson B, Yu K. The hidden information state model: a practical framework for pomdp-based spoken dialogue management. *Computer Speech and Language* 2010; **24**(**2**):150–174.
(14) Kim D, Kim JH, Kim KE. Robust performance evaluation of POMDP-based dialogue systems. *IEEE Transactions on Audio, Speech, and Language Processing* 2011; **19**(**4**):1029–1040.
(15) Williams JD, Poupart P, Young S. Factored partially observable Markov decision processes for dialogue management. *Proceedings of the 4th Workshop on Knowledge and Reasoning in Practical Dialog Systems*, 2005; 76–82.
(16) Picard RW. *Affective Computing*. The MIT Press: Cambridge; 1997.
(17) Lopatovska I, Arapakis I. Theories, methods and current research on emotions in library and information science, information retrieval and human–computer interaction. *Information Processing and Management* 2011; **47**:575–592.

(18) Ren FJ. Affective information processing and recognizing human emotion. *Electronic Notes in Theoretical Computer Science* 2009; **25**:39–50.
(19) Calvo RA, D'Mello S. Affect detection: an interdisciplinary review of models, methods, and their applications. *IEEE Transactions on Affective Computing* 2010; **1**(**1**):18–37.
(20) Ren FJ, Kang X. Employing hierarchical Bayesian networks in simple and complex emotion topic analysis. *Computer Speech & Language* 2013; **27**(**4**):943–968.
(21) Ren FJ, Wu Y. Predicting user-topic opinions in twitter with social and topical context. *IEEE Transactions on Affective Computing* 2013; **4**(**4**):412–424.
(22) Li J, Ren FJ. Hybrid approach for word emotion recognition. *IEEJ Transactions on Electrical and Electronic Engineering* 2013; **8**(**6**):616–626. DOI:10.1002/tee.21905
(23) André E, Dybkjær L, Minker W, Heisterkamp P. Affective dialogue systems: tutorial and research workshop. *Lecture Notes in Artificial Intelligence*. Springer-Verlag: Berlin 2004.
(24) Skowron M, Theunis M, Rank S, Kappas A. Affect and social processes in online communication-experiments with an affective dialog system. *IEEE Transactions on Affective Computing* 2013; **4**(**3**):267–279.
(25) Pittermann A, Pittermann, J, Minker, W. Emotion recognition and adaptation in spoken dialogue systems. *International Journal of Speech Technology* 2010; **13**:49–60.
(26) López-Cózar R, Silovsky J, Kroul M. Enhancement of emotion detection in spoken dialogue systems by combining several information sources. *Speech Communication* 2011; **53**:1210–1228.
(27) Callejas Z, Griol D, López-Cózar R. Predicting user mental states in spoken dialogue systems. *EURASIP Journal on Advances in Signal Processing* 2011; **6**:pp1–21.
(28) Han MJ, Lin CH, Song KT. Robotic emotional expression generation based on mood transition and personality model. *IEEE Transactions on Cybernetics* 2013; **43**(**4**):1290–1303.
(29) Bui TH, Poel M, Nijholt A, Zwiers J. A tractable DDN-POMDP approach to affective dialogue modeling for general probabilistic frame-based dialogue systems. *Natural Language Engineering* 2007; **15**(**2**):273–307.
(30) Bui TH, Zwiers J, Poel M, Nijholt A. Affective dialogue management using factored POMDPS. *Interactive Collaborative Information Systems* 2010; **281**:207–236.
(31) Wang Y, Li L, Ren FJ, Quan C. Q. POMDP based affective model and its application on an intelligent music player. *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering*, 2012; 407–417.
(32) Spaan MTJ, Vlassis NA. Perseus: randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research* 2005; **24**:195–220.

Fuji Ren (Member) was born in 1959 in China. He received the B.E. and M.E. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 1982 and 1985, respectively, and the Ph.D. degree from the Faculty of Engineering, Hokkaido University, Japan, in 1991. He worked at CSK, Japan, where he was a Chief Researcher of NLP from 1991. From 1994 to 2000, he was an associate Professor with the Faculty of Information Sciences, Hiroshima City University. He became a Professor in the Faculty of Engineering, the University of Tokushima, in 2001. His research interests include natural language processing, artificial intelligence, language understanding and communication, and affective computing. Prof. Ren is a member of the IEICE, CAAI, IEEJ, IPSJ, JSAI, and AAMT, and a senior member of the IEEE. He is a fellow of the Japan Federation of Engineering Societies, and is the president of International Advanced Information Institute.

Yu Wang (Non-member) was born in 1983 in China. He received the B.S. degree from Anqing Normal University in 2004 and the M.S. degree from Hefei University of Technology (HUT) in 2009. Currently, he is pursuing the Ph.D. degree with the School of Computer and Information, HUT. He is also a Lecturer with the Department of Mathematics and Physics, Hefei University. His research interests include spoken dialogue systems and affective computing.

Changqin Quan (Member) was born in Wuhan, China, in 1978. She received the B.E. and M.E. degrees from Huazhong Normal University, Wuhan, China, in 2000 and 2005, respectively, and the Ph.D. degree from the University of Tokushima, Tokushima, Japan, in 2011, and. She is currently a Professor with Hefei University of Technology. Her research interests include natural language processing, machine learning, and affective computing.