A Quality-Focused Spoken Dialog System With Reinforcement Learning And Simulated User

Minh-Quang Nguyen
Dept. of computer
UQAM
Montreal, Canada
nguyen.minh-quang@uqam.ca

Philip H.P. Nguyen
Dept. of justice
Government of South Australia
Adelaide, Australia
nguyen.philip@saugov.sa.gov.au

Douglas O'Shaughnessy
INRS-EMT
Montreal, Canada
dougo@emt.inrs.ca

Jean-Guy Meunier
Dept. of philosophy
UQAM
Montreal, Canada
meunier.jean-guy@uqam.ca

***Abstract*—In this paper, we propose a solution to the problem of formulating strategies for a spoken dialog system. Our approach is based on reinforcement learning (RL) with the help of a simulated user (SU), involving unsupervised learning and trials-and-errors with a return value (negative or positive) for each decision, in order to identify an optimal dialog strategy. Our method considers the Markov decision process (MPD) to be a framework for representation of speech dialog in which the states represent history and discourse context, the actions are dialog acts and the transition strategies are decisions on actions to take between states. We present our reinforcement learning approach with a novel objective function that is based on dialog quality as well as other quantitative factors.**

Keywords- Learning control systems; Unsupervised learning; Markov processes; Artificial intelligence; Intelligent systems

I. INTRODUCTION

Speech recognition and speech synthesis techniques have become increasingly efficient and robust, facilitating implementation of human-machine spoken dialog systems. In these applications, a machine speaks to a human by imitating human communication acts. However, human-machine dialogs still lack naturalness and flexibility. One of the most important issues in this domain is the management of conversational interactions between human and machine. These interactions do not occur randomly, but rather follow precise rules of the communication acts.

While some research is focused on the acoustic and semantic aspects of speech signals (what to say), other is directed towards dialog strategies (how to say) in order to control those interactions. A number of machine learning approaches for the design of such strategies have been proposed in literature [2][3][4][5][7][8][9]. One recent promising technique is reinforcement learning (RL) involving a simulated user (SU). With RL, a machine could develop an optimal strategy from observation examples, provided that they are comprehensive. However, in the current state of the art, it is not possible to produce such a strategy by directly learning from corpora of dialog data (Schatzman et al., 2006) [8], mainly due to their small sizes, which are insufficient to permit exploration of all possible states and actions pertinent to a dialog. In addition, it is not certain that an optimal strategy is present in those corpora even when they are of reasonable sizes. Hence the idea of creating a simulated user to assist learning [4][7][8][9]. In our implementation, we model dialog acts on Markov properties (actions, states, and transitions) [4][10]. When these properties are satisfied, the resulting dialog strategy is called a Markov Decision Process (MDP) (Sutton and Barto, 1998) [10].

The main feature of our architecture resides in a novel objective function that achieves optimal dialog strategy based on quality of conversation [2], rather than its "quantity" (or duration), similar to what is proposed in [4]. This quality could be measured via the variation of illocution questions such as direct, implicit, explicit and repetitive questions.

Our paper is organized as follows: Section 2 describes the Markov Decision Process (MDP). Section 3 summarizes the RL technique. Section 4 details our proposed RL architecture with a simulated user, including the parameterization of our objective function and the initialization of the reward variables, all necessary for a satisfactory learning. Implementation results follow in Section 5. And finally, Section 6 concludes our proposal and suggests new directions for research.

II. DIALOG AS MARKOV DECISION PROGRESS

Recent research suggests that the formalism of the Markov Decision Process (MDP) could be used in the representation of dialog acts and in the modeling of problems relating to dialog strategy optimization [4][5][8]. As per [6][10], a MDP is a 4-tuple: (S, A, P(.,.), R(.)) in which:

- $S = \{s_1, s_2, ..., s_n\}$ is the set of states, representing the complete dialog, i.e., the knowledge of the concerned domain. A state at time t is denoted s_t or s, and at time t+1, s_{t+1} or s'.
- $A = \{a_1, a_2, ..., a_m\}$ is the set of actions, which are dialog acts. An action carried out at time t is denoted a_t or a, and at time t+1, a_{t+1} or a'.

- P: S×A→S is the transition function, which associates a state and an action, with another state (which is the outcome of the action). An important property of an MDP is that the probability $P(s_{t+1}, r_{t+1} \mid s_t, a_t)$ of transitioning to state s_{t+1} and collecting the reward r_{t+1} at that state depends solely on the current action a_t and the current state s_t.
- $R(s_t)$ is the reward function, representing the reward received in reaching state s. The goal of an optimal strategy is to maximize the sum of all rewards collected, discounted by a rate γ (between 0 and 1), which could be expressed by the following mathematical formula:

$$\sum_{t=0}^{\infty} \gamma^t R(s_t) \quad (1)$$

MDP permits visualization of a dialog strategy π as a path connecting different states reached through different actions. An optimal strategy π* is a strategy that maximizes the discounted cumulative sum of all rewards collected on that path. The Markov decision problem is to identify that optimal strategy after some learning. RL algorithms help us solve that problem.

III. REINFORCEMENT LEARNING

There exist several learning approaches for dialog strategy optimization such as the non-supervised method from Pietquin (2004) [5] and the hybrid (reinforcement and supervised) method from Henderson et al. (2005) [3]. RL is the best choice for machine learning when the environment is uncertain, unknown or complex. In the case of a spoken dialog system (SDS), the machine usually cannot understand everything that is said by a human. This is due to a variety of limitations, such as degraded speech recognition (e.g., signals distorted by the environment), deficient semantic interpretation, etc. Sometimes, the machine must interact with the environment without being certain about the coherence and/or correctness of its choice of dialog acts. It must then learn by trials and errors, by analyzing all the responses from the user and the outcomes of its actions.

In this perspective, the reward function defined in an MDP permits the machine to progress in its learning despite an uncertain environment. Dialog acts are translated into a sequence of states and actions, with each action leading to a state where a reward is collected. The cumulative reward can be expressed by the following generalized formula that extends (1):

$$\sum_{t=1}^{T} \gamma^t R(s_{t+1}, a_t, s_t) \quad (2)$$

Here the learning task consists of optimizing the interactions between human and machine, and the goal is to find a strategy that maximizes the value of R. That value could be recursively calculated from the state-value function $V^{\pi}(s)$, and the state-action or Q-learning function $Q^{\pi}(s,a)$ of the strategy π [6][10]. The associated optimization functions are $V^*(s)$ and $Q^*(s,a)$, defined as:

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} P(s,a,s')[R(s,a) + \gamma V^*(s')] \quad (3)$$

$$Q^*(s,a) = \sum_{s' \in S} P(s,a,s')[R(s,a) + \gamma \max Q^*(s',a')] \quad (4)$$

Those optimal value functions are related by equation:

$$V^*(s) = \max_{a} Q^*(s,a) \quad (5)$$

Note that $P(s,a,s')$ is the probability of an action selected with a policy π at state s moving to state s'. The $R(s)$ value can be *stochastic* or *determinist*. It is generally stochastic for a SDS because the reward value is not known in advance and depends on the user response, which could be different even when the same state is visited more than once.

A number of algorithms exist for the determination of these optimal values [9]. However, the simplest and most efficient algorithm for RL is Q-learning, which consists of maintaining the Q-value, i.e. the set of all Q(s,a) values for all pairs of state s and action a. A function objective is called at the end of training session to evaluate the maximum value of R. This evaluation permits the identification of the optimal strategy that the system should adopt in order to achieve a quality dialog.

IV. OUR APPROACH

One important aspect in SDS is the strategy of confirmation. The system has to rely on different actions (questions) to interact with the user, similarly to what happens in a human-human dialog. The system should not always repeat the same question when speech recognition becomes deficient, nor should it always use explicit or implicit questions. We first define what a successful dialog should be in an SDS, based on quantitative and qualitative measurement of the dialog. Then, we show how to design an MPD and how to formalize the criteria for a successful dialog into the objective function of reinforcement learning.

A. Quantitative measurement

Quantitative measurement is based on the number of dialog turns (a dialog turn consists of one question and one response). The best dialog under this criterion is the one that permits the system to get confirmation of all responses from the user with a minimum number of dialog turns (which should also not exceed a predefined upper limit). The number of dialog turns in any SDS should also be above a certain lower limit that could be worked out from the number of information slots that the system wishes to fill (e.g., number of nights, number of persons, date, etc. in a hotel reservation system). In our implementation, these limits are as follows:

$$1.6 * N_s <= N_{dt} <= 3.0 * N_s \quad (6)$$

Where N_{dt} denotes a total number of dialog turns in one session of dialog and N_s, the number of information slots. For example, an SDS with 4 information slots should provide a number of dialog turns between 6 (i.e., 1.6*4 = 6.4) and 12 (i.e., 3.0*4). Less than 6 turns is not possible because the system needs to ask at least 4 questions for the 4 information slots, plus 2 questions for confirmation of the responses. Over 12 turns is not acceptable as this might indicate a high error rate of the speech recognition and/or the understanding components. These ratios (1.6 and 3.0) are based on our analysis of a large data corpus of some 10,000 dialogs in the hotel and telephone service domains, generated in laboratory by Nuance[1].

B. Qualitative measurement

The qualitative measure of a dialog has two criteria. The first is to detect redundancy in the question type for the same slot (e.g., M: when are you leaving? U: Monday. M: when are you leaving? U: I said I'm leaving on Monday). The less redundancies the better for the dialog. The second is to examine for each type of illocution question, the ratio between the number of questions of this type and the total number of dialog turns. These ratios must be within certain limits to be acceptable.

Figure 1 shows the ratios of a successful dialog according to Condition (6). A successful dialog must have 20% to 60% of questions to be of type implicit and 20% to 40% of questions to be of type direct. In addition, because the system can deal with different user behaviors and different error rates of the speech recognition and understanding components, the total number of explicit and repetitive questions should not exceed 20% (e.g., 10% for explicit, 5% for Automatic Speech Recognition or ASR errors, and 5% for Natural Language Understanding or NLU error).

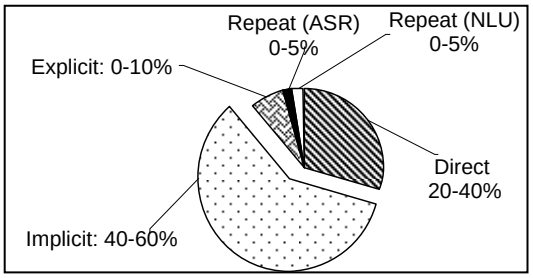

Figure 1. A quality dialog

A high ratio of repetitive questions may indicate a deficiency of the speech recognition module. The two types of repetitive question (ASR and NLU) are to distinguish between what the system can *'hear'* from the user input and what the system can *'understand'* from it. Table 1 shows a successful dialog realized in 6 turns between a user (U) and the system (M), with 4 information slots. 50% of questions are implicit questions (3 out of 6), 33% are direct (2 out of 6), 17% are repetitive ASR questions (1 out of 6), and no explicit questions (0 out of 6). This dialog satisfies our qualitative measure.

M: [direct question] *When would you like to reserve?* ***U****: this Friday* ***M: [direct question]*** *How many nights would you like to stay?* ***U****:3 nights* ***M: [implicit question]*** *and how many people would you like to reserve for 3 nights?* ***U****: We are 2.* ***M: [implicit question]*** *What kind of room would you like for 2?* ***U:*** *uhm…we want a royal suite* ***M: [repetitive question and direct]*** *Sorry, I did not understand what you said, what kind of room would you like?* ***U :*** *ok, we want a suite* ***M: [implicit question]*** *You have reserved for 2 people for 3 nights in a suite starting this Friday. Can you confirm by yes or no?* ***U :*** *yes* ***M: [closing dialog]*** *Thank you for using the hotel reservation dialog system. Good bye.*

Table 1. Example of a successful dialog in the hotel reservation domain.

C. MPD Formalization

An action in an MDP modeling the above system is a question of one of the above five types (i.e., direct, explicit, implicit, and 2 types of repetitive) on an unknown or unconfirmed variable (i.e., a slot). The action at each state depends on the user response. So the process is *stochastic,* and not *stationary*, i.e., the same action is not always performed when the same state is reached. An MDP could be designed to model the above system as follows:

S_n	V_n	E	a_t	a_{t-1}	R	Q
[1,4]	[0,2]	[0,4]	[0,4]	[0,4]	[+1,-1]	-1

Table 2. An MDP formalization with 4 information slots

In Table 2, S_n represents one of the 4 slots (check-in date, number of nights, number of persons, and type of room) that the system wishes to fill..Each slot has three possible values (V_n): unknown, known, and confirmed, which gives a total of $3^4 = 81$ possible states in the MDP. For each state, we have five error rates (E), output by the speech recognition and understanding components. These error rates are generated randomly to simulate different types of user response. We also associate to each state *s* five possible actions (a_t): implicit, explicit, direct and 2 types of repetitive questions (one for ASR error and one for NLU error). The number of combinations of the pair (state, action) thus becomes 81^5 or over $2{,}8x10^{11}$ possible combinations. It is impossible to handcraft all these combinations, so the use of a simulated user to interact with the system is the only sensible option. The recording of the past action a_{t-1} helps detect if the system executes the same action again. The reward value *R* is +1 or -1 depending on whether the action at that state yields a satisfactory response or not. And finally, the Q value is initiated to a negative value for all states before the start of each training session.

[1] Nuance Communication Inc. - www.nuance.com.

D. Reinforcement learning

Our proposed approach is based on an architecture described in [7][8][9]. It consists of two steps: First, a simulated user is created (according to an algorithm given in [1]). Second, a learning agent is built based on a Q-learning algorithm [6][10]. In direct interaction with the simulated user, the learning agent learns its strategy by examining the answers of the simulated user (represented by the reward values *R*). The Q-value is calculated in each training session. After a number of sessions (in general, in the order of a million [7]), training stops and the objective function is evaluated for each Q value. The strategy which corresponds to the maximum value of the objective function is the optimal strategy.

1) Objective function: The objective function encapsulates the quantitative and qualitative measurements of a dialog, described earlier. When the system has a recognition and understanding error rate between 0 and 20%, the system is considered of high performance and we suggest direct questions (such as: when would you like to reserve?). Implicit questions (such as: how many people for this weekend?) are recommended when the rate is between 20 and 40% (i.e., an average performance system). Explicit questions (such as: did you say weekend?) are the best choice if the error rate is in the 40-60% range (i.e., a medium performance system). With a 60-80% error rate (i.e., a low performance system), repetitive NLU questions (such as: sorry, I don't understand, can you repeat please?) should be considered. And when the error rate is over 80% (i.e., a very low performance system), repetitive ASR questions (such as: Sorry, I don't catch it, can you repeat please?) should be the best option. Note that the objective function also detects question redundancy by comparing the current action with the previous one in the MDP. The goal of using different types of question is to give the dialog a style that is as natural and as "human-like" as possible.

2) Simulated user: The simulated user is built separately from, and prior to, the dialog system. The intention is not to optimize the simulated user, in terms of attempting to complete the dialog as soon as possible, or to produce a high quality dialog. These goals are for the dialog system itself. In fact, the simulated user should be at times as "good" (cooperative) as possible and at other times as "bad" (un-cooperative) as possible, while still being realistic (i.e., without being unreasonably illogical), so that it can produce a large number of varied dialogs to train the system. We use a random function to generate cooperative and un-cooperative user behaviors. In real life, a user may not co-operate if the system response is not what the user expects. In our system, a simulated cooperative user would correctly answers the question posed by the system without attempting to disturb it, while a simulated un-cooperative user constantly changes its goal and gives incorrect answers in an attempt to detract the system. These different behaviors of the simulated user permit the generation of a large and varied data corpus for training and testing the SDS.

V. EXPERIMENTSAND RESULTS

We implement our above system with Perl and C++ languages on a Pentium III PC. 9 episodes of training (i.e., 9000 runs) and 1 episode of testing (i.e., 1000 runs) were conducted. We trained first with a simulated un-cooperative user followed by a simulated cooperative user (so that the system could learn from a variety of different scenarios first and then could be more efficient in later dialogs). In testing, we set up random behaviors for the simulated user, which could be both cooperative and un-cooperative in the same dialog (this is to provide worst case scenarios to test the system).

A. Results

Table 3 shows different results in training. When the error rate is less than 20% (i.e., system considered as high performance), cooperative user simulation gives 345 successful dialogs versus only 61 for un-cooperative user simulation. When the error rate is between 20 and 40%, the number of successful dialogs decreases to 187 with a cooperative user and 10 without it. But when the error rate is over 60% (i.e., a low or very low performance system), it is hardly possible for the system to produce any quality dialog at all. In most cases, the objective function failed in the quantitative measurement, mainly because the system could not obtain all the confirmations from the user within the maximum number of dialog turns permitted. In summary, most successful dialogs are achievable when the error rate is below 20%.

Error rate ASR/NLU	**No of Successful dialogs**	
	Un-cooperative user	**Cooperative user**
0-20%	61	345
20-40%	10	187
40-60%	4	60
60-80%	0	3
80-100%	0	2

Table 3: Numbers of successful dialogs after 9 episodes of training.

Figure 2 shows that with a cooperative user, learning increases rapidly in the first episode of training (i.e., 1000 runs) but becomes stationary after 9 episodes (i.e., 9000 runs). We did continue training after 9000 runs but no more quality dialog was achieved. Figure 3 illustrates the results of training with an un-cooperative user. Note that in the second episode the number of successful dialogs only increased when the error rate was low (under 20%).

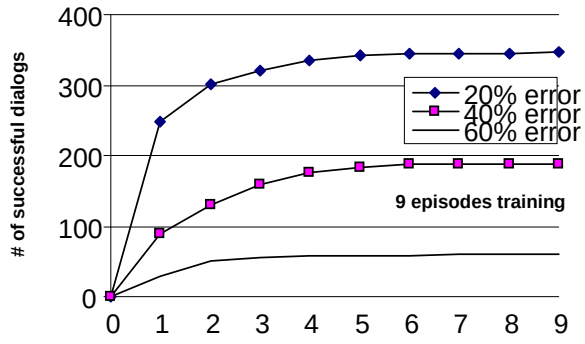

Figure 2. Results of training with a cooperative user

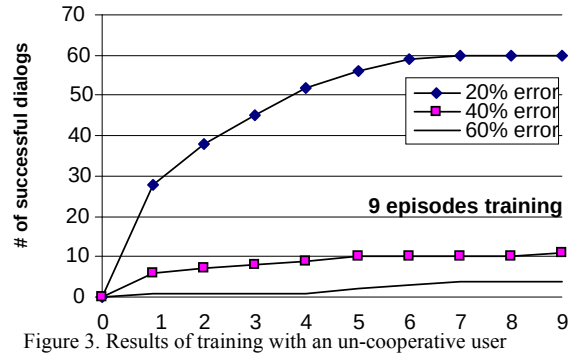

Figure 3. Results of training with an un-cooperative user

Figure 4 shows the results of testing with a simulated user with random behavior. In total, when the error rate is below 20%, 193 successful dialogs were achieved versus only 102 dialogs when the rate is 40%. The results are insignificant when the error rate is over 60%. Only 26 successful dialogs were obtained. Note that these results also depend on how much variation in the "randomness" of the behavior of the simulated user when generated by the random function.

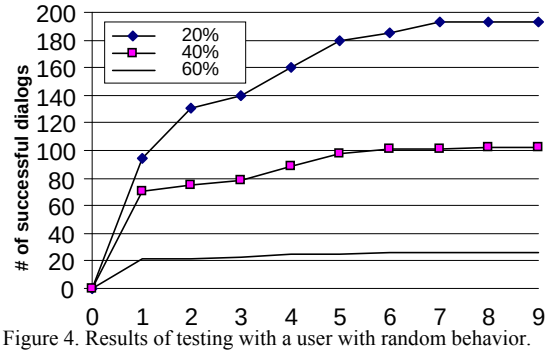

Figure 4. Results of testing with a user with random behavior.

M: [direct question] *When would you like to reserve?*

U *: this Friday*

M: [implicit question] *how many nights would you like to say starting this Friday?*

U*: two nights.*

M: [explicit question] *Did you say 2 nights?*

U: *of course, I said tha.t.*

M: [implicit question] *what kind of room do you want for 2 nights?*

U : *we would like a royal suite.*

M: [direct question] *How many people do you want to reserve for?*

U : *two people*

M: [implicit question] *You have reserved for 2 people in a royal suite for 2 nights starting this Friday. Can you confirm by yes or no?*

U : *yes*

M: [close dialog] *Thank you for using the hotel reservation dialog system. Good bye*

Table 4: Example of a successful dialog with a cooperative user in 6 turns.

Table 4 shows an example of a dialog during testing with a cooperative user. The dialog here respects the qualitative measurement. There are 50% implicit, 33 % direct and 17% explicit questions and no repetitive questions.

VI. CONCLUSION AND FUTURE WORK

Our design of a machine learning model is based on recent research in dialog learning strategy. A successful dialog can be achieved by learning with the help of a simulated user, implemented through MDP and RL. This type of learning is guided by an objective function that implements our measurement criteria which are more focused on the quality of the dialog. Our approach could provide an efficient and reliable solution for SDS applications of the future, in which human-machine dialogs should be as natural and as human-like as possible. However, our experiments also show that with an un-cooperative user, the system cannot provide any quality dialog when the speech recognition and understanding error rate is higher than 60%. Furthermore, in this case, the processing cost increases substantially, in terms of CPU usage (and hence response time). We are working on implementation of different strategies for these scenarios, especially when the error rates are medium and high (i.e., 20%-40% and 40-60%).

ACKNOWLEDGMENT

The authors would like to thank Nuance Communications Inc. for allowing us to conduct this work in its laboratory.

REFERENCES

[1] Cuayahuitl, H., Renals, S., Lemon, O., Shimodaira, H., Human-Computer Dialog Simulation Using Hidden Markov Models. in *Proc. of IEEE ASRU*, Cancun, Mexico, 2005.

[2] English, M., Heeman, P., Learning Mixed Initiative Dialog Strategies By Using Reinforcement Learning On Both Conversants. In *Proc. Of HLT/EMNLP*, pp. 1011-1018, Vancouver, Canada, 2005.

[3] Henderson, J., Lemon, O., Georgila, K., Hybrid reinforcement/supervised learning for dialog policies from communicator data. In *Proc. of IJCAI on KRPDS*, Edinburgh, Scotland, 2005.

[4] Levin, E., Pieraccini, R., Eckert.,W., A Stochastic Model of Human Machine Interaction for Learning Dialog Strategies. In *Proc. of the IEEE ICASSP*, Istanbul, Turkey, pp. 1883-1886, 2000.

[5] Pietquin., O., *A Framework for Unsupervised Learning of Dialog Strategies*. Presses Universitaires de Louvain, *SIMILAR Collection*, ISBN 2-930344-63-6, 2004.
[6] Puterman, M. L., Markov Decision Processes, Wiley, 1994.
[7] Schatzmann, J., Stuttle, M., Weilhammer, K., Young, S., Effects of the user model on simulation-based learning of dialog strategies. *Proc. of ASRU*, San Juan, Puerto Rico, 2005.
[8] Schatzmann, J., Weilhammer, K., Stuttle, M., Young, S., A Survey of Statistical User Simulation Techniques for Reinforcement-Learning of Dialog Management Strategies. *Knowledge Engineering Review*, 2006.
[9] Scheffler, K., Young, S., *Simulation of Human-Machine Dialogs,* Cambridge, U.K.: Engineering Dept., Cambridge University, Tech. Rep. CUED/F-INFENG/TR 355, 1999.
[10] Sutton, R., Barto, A., *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.