On-Line Learning of a Persian Spoken Dialogue System Using Real Training Data

Maryam Habibi
Department of Computer Engineering,
Sharif University of Technology, Tehran, Iran
E-mail: ma_habibi@ce.sharif.edu

Hossein Sameti
Department of Computer Engineering,
Sharif University of Technology, Tehran, Iran
E-mail: sameti@sharif.edu

Hesam Setareh
Department of Computer Engineering,
Sharif University of Technology, Tehran, Iran
E-mail: setareh@ce.sharif.edu

Abstract

The first spoken dialogue system developed for the Persian language is introduced. This is a ticket reservation system with Persian ASR and NLU modules. The focus of the paper is on learning the dialogue management module. In this work, real on-line training data are used during the learning process. For on-line learning, the effect of the variations of discount factor (g) on the learning speed is investigated as the second contribution of the research. The optimal values for g were found and the variation pattern of the action-value function (Q) in the learning process was obtained. A probabilistic policy for selecting actions is used in this work for the first time instead of greedy policies employed in previous works.

Keywords: *Learning spoken dialogue, action value function, discount factor.*

1. Introduction

Applying spoken dialogue systems (SDS's) are being grown in the real life more rapidly because of the advances done in the design and management of these systems. The traditional touch tone computer telephony systems are being substituted by the SDS's. In a typical SDS, the user speaks naturally to the system through a phone line and the system provides the required information or performs the required action. Banking and ticket reservation are typical examples of the prevalent SDS's. A spoken dialogue system has five units: automatic speech recognition (ASR), natural language understanding (NLU), dialogue management (DM), spoken language generation (SLG), and text to speech (TTS).

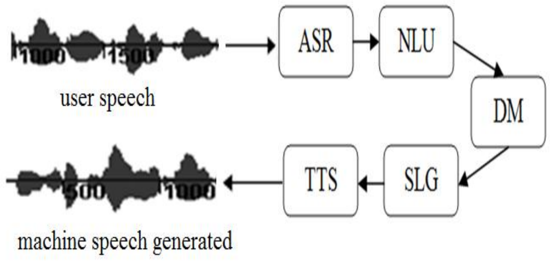


Figure 1. Spoken dialogue system

The block diagram of an SDS is shown in Fig. (1) The ASR goal is to transcribe the speech. Its input is an audio stream and its output is the recognized utterance. The NLU unit produces meaning representation from the ASR output. Due to the errors in the recognition and understanding processes the SDS may respond incorrectly. The DM unit is designed to reduce the effect of these errors. Its input is the output of NLU unit, and its outputs are the communicative goals. The task of the SLG unit is to produce an output string to be read to the user. The SLG unit receives the communicative goals as the input and generates strings spoken to the user in an audio stream form as the output.

Implementation and construction of an SDS is difficult in large tasks. The output of the ASR and NLU units always contain errors; the ASR output containing errors are used to generate simulated training data [1] employed by researchers such as in [2] for implementation of SDS. To handle this complexity and to reduce the effect of errors on the performance of the system, mathematical methods are applied for modeling and learning the DM unit.

The DM unit has an essential role in SDS's. The DM unit interacts in a natural way to let users get the information that the system supports and do what the users desire. Selecting the best action in the conversation is the goal of the DM unit. The first generations of the DM units used manually-designed policies. Today, statistical methods empower a dialogue management system to find optimal policies. There are two main statistical approaches for learning the DM unit; the first approach is tracking multiple dialogue states. This approach can be classified into three categories: exact updating [3-6], network based approximations [7-10], and M-Best methods [11, 12]. The second approach is to select the actions of the dialogue system automatically, and can be classified into 5 categories: Markov decision process based methods [13, 14], partially observable Markov decision process based methods [15, 16], feature-based reinforcement learning [17], utility maximization [18], and supervised learning based approaches [19].

Previous works are mostly based on Markov decision processes [13]. While the Markov decision process (MDP) provides a natural model for conversation in theory, it has achieved limited success in real practical applications. Then partially observable Markov decision process (POMDP) provides a statistical framework for handling uncertainty in the states for the English language [2, 20, 21, 22], but this method requires a large training dataset and is computationally very complex [1]. To overcome this complexity, some researchers use free-model approach, and recently researchers compared model-based and model-free approaches for SDS's to show that the model-based approach didn’t perform as well as the model-free approach [23]. Researchers used simulated training data for the English language [2, 20, 21, 22]. The model-free approach of reinforcement learning has near-optimal solution and needs a lot of data for learning. The problem occurring in the learning procedure of the SDS's is that the DM unit learns from the first samples of the training data and usually is not affected by the rest of it. In order to influence the training procedure in a way that all the training data affects the DM learning procedure, in this paper we focus on finding the optimum value for the discounting factor in the learning process.

Previous learning methods for training DM unit [20, 24] have used a greedy policy for assigning actions to the states, i.e., they assign the most probable action to the state. In this work, the policy used for selecting actions during learning is based on a probabilistic selection method; furthermore, real users are employed for generating the training data.

This paper is organized as follows. Section 2 discusses the reinforcement learning (RL) methods and the RL methods used for learning DM so far. Section 3 contains our approach for modeling and learning the DM unit. Section 4 contains our implementation issues such as conditions of our experiments and the data collection procedure. Section 5 shows our experimental results. Section 6 contains summary and conclusions.

2. Learning Methods Used for Dialogue Management

The process of learning through reinforcement learning methods can be defined as a process of modification of each component of the agent to bring the component into closer agreement with the feedback obtained by observing the environment. As a result, reinforcement learning methods are used by SDS's for corresponding each state to the best action because these learning methods can reduce the effect of the errors that may occur in ASR and NLU units on the performance of the system. Two important characteristics of the reinforcement learning methods are their capability to use trial-and-error and delay reward, which empower an agent to learn how to perform in an autonomous manner and improve the quality of its behavior [8, 26].

2.1. Reinforcement Learning Methods

Reinforcement learning explains how to map situations to actions for getting the maximum reward signal [8, 26]. RL methods are categorized in three types which are 1) dynamic programming (DP), 2) Monte Carlo methods (MC), and 3) temporal difference (TD). The DP methods are model-based methods which provide well-developed and simple mathematical calculations but they need a precise and complete model of the environment. The MC methods are model-free methods, i.e., they don't need a model, and the problems are solved based on average sample return. Their drawback, however, is that they cannot be used for batch training since they are incremental learning methods. Moreover, since their value functions are updated after each episode, these learning methods are very time consuming. The TD methods are free-model methods similar to the MC methods, but their convergence speed is higher than the MC because of their state value functions updated after each step instead of each episode.

Model-based approaches are used to compute optimal assignment of actions to states. It is assumed that S is a set of the states comprising different states of the environment and $A(s)$ is the set of actions. It consists of all the actions an agent can do when it interacts with a user. Both of these two sets are finite and its dynamics are based on a set of transition functions $\rho_{s\acute{s}}^{a}$ that represents the transition probability distribution between two states of S given a specific action a and $R_{s\acute{s}}^{a}$ is the reward function representing the value of each action in each state for all $s \in S, a \in A(s)$ and $\acute{s} \in S$. Then DP can be used to compute the optimal action-value functions $(Q^*(s,a))$ according to Bellman optimality equations (Eq. 1).

$$Q^*(s,a) = \sum_{\acute{s}} \rho_{s\acute{s}}^{a} [\, R_{s\acute{s}}^{a} + \gamma \max_{a'} Q^*(s',a')] \tag{1}$$

γ being the discount factor $(0 \le \gamma \le 1)$. The effect of γ is that the states visited in the beginning of each episode are given less reward due to the effect of the discount factor.

MC methods are model-free approaches. These are incremental methods usually used for episodic tasks. In the MC methods, at the end of each episode $Q^{*}(s,a)$ will be updated based on the average of the rewards obtained based on the current policy, the algorithm of which is shown in Fig. 2 [25].

TD methods which combine the advantages of the DP method and MC method can learn experiences without a model of the environment, and update $Q^{*}(s,a)$ step by step, not episode by episode [25]. However, their analyzing is difficult.

2.2. Reinforcement Learning for the DM

The speed of learning of the MC methods is lower than the DP methods, but the DP methods need the complete model including transition probabilities and expected immediate rewards. Conventional dialogue systems use simulated training data for modeling the environment [1]. DP methods use the simulated data for learning; this is the reason researchers have used the DP method [20, 23, 26, 27]. The MC methods are also used by certain researchers for the DM unit learning [2, 24]. The simulated training data have been used for training the DM unit [2, 20, 23, 24, 26, 27]. The work presented in this paper is done using on-line MC methods for learning of the DM unit by real user training data, in which real users provide real errors in the outputs of the ASR and NLU units.

```
Initialize, for all $s$ Î $S$, $a$ Î $A(s)$:
    $Q(s,a) \leftarrow$ arbitrary
    Returns$(s,a) \leftarrow$ empty list
    $\pi \leftarrow$ an arbitrary ε-soft policy

Repeat forever:
    (a) Generate and episode using $\pi$
    (b) For each pair s,a appearing in the episode:
            $R \leftarrow$ return following the first occurrence of $s,a$
            Append $R$ to $Returns\ (s,a)$
            $Q(s,a) \leftarrow$ average $(Returns(s,a))$
    (c) For each $s$ in the episode:
            $a^{*} \leftarrow \mathrm{argmax}_{a} Q(s,a)$
            For all $\alpha$ Î $A(s)$:
            $p(s,a) \leftarrow \begin{cases} 1-e+e/|A(s)| & if\ a=a^{*} \\ e/|A(s)| & if\ a \neq a^{*} \end{cases}$
```

Figure2: Monte-Carlo algorithm [25]

3. Proposed Approach for DM Modeling and Learning

In this paper, a ticket reservation system is designed. Getting the origin and the destination of the route are its initial goals and reserving tickets for users is the final outcome of this system. The designed system has 11 states shown in Fig. 3. Each state contains two slots, the first slot is the origin and the second slot is the destination. Each slot can have three values. “0” means that the slot-value is unknown, “1” indicates that the slot-value is known and “2” means that the slot-value is confirmed. In each state the “0” causes the system to do the request action and questions about the slot-value. For the “1” indicator, the system can do two types of actions; the system can go to the goal state with doing the Go-to-goal action or can do the confirmation action. If the system does the confirmation action and the user's judgment is YES, the system goes to a state that the confirmed slot is “2”. If the user's judgment is NO the system goes to a state that the confirmed slot is “0”. If the output of NLU is “NULL”, i.e., the NLU cannot match the output of the ASR with any grammar, the system remains in its previous state and repeats its previous action. The start state is defined for restarting the system. The goal state is the final state where the target is identified and the user is asked about the validity of the selected target.

Reinforcement learning based on MC method is used for learning. The goal of learning in this system is mapping each state to a specific action so that shorter dialogues and fewer errors are resulted. The action-value function, Q, represents the sum of rewards obtained for each action during the learning process. In the initial learning dialogue Q is set to be equal for all actions; then at the end of any episode, Q changes based on the reward acquired by any of these actions in the learning steps. For learning the DM unit, at first we use initial random policy, and then the system is trained with incremental training data. Our policy for selecting an action is not greedy or ϵ-greedy. It is based on a probabilistic model, i.e., the probability of selecting each action depends directly on its Q value. It is computed by dividing the Q value of that action by the sum of the Q values of all actions within the same state. Since users are diverse and their judgments are different, when there are not enough training dialogues, finding out a suitable γ for fast and correct convergence is a problem. After convergence, we change the discount factor γ, then we retrain the DM unit with another initial random policy. This procedure is repeated with four different values of γ.

The reward function defines which policy of the agent maximizes the average return reward. As a result, the reward function is essential in the process of finding the optimum policy. Our reward function is defined in Table 1 based on two intentions, the first is minimization of the length of the dialogue session, the penalty of each action is the time of performing that action and the second intention is generating correct responses by the agent no matter how long the dialogue takes. As a result, two types of reward functions are defined, one of which, named implicit reward, is produced in a way that it can be used to learn the system whether it should ask questions for taking confirmation or not. Two actions, confirmation and Go-to-goal, are defined for certain states. If a confirmation action is performed by the agent and confirmed with the YES judgment of the user, this action within state gets a negative reward and other actions within the state get positive rewards, and conversely for a confirmation action performed by the agent and replied by the NO judgment of the user, this action within the state gets a positive reward, and other actions within the state get negative rewards. The other reward is named explicit reward and is based on asking the user about the validity of the selected target at the end of each episode. If the user's judgment is YES, positive rewards are given to all actions that are selected throughout the episode and negative rewards are given to the other actions; conversely when the user judgment is NO, negative rewards are given to all actions that are selected during the episode, and positive rewards are given to the other actions. Then by changing the discount factor, the percent of reward given actions in each state through explicit reward is changed so that by reducing the discount factor, the influence of its explicit reward is ignored compared to its implicit rewarding.

The advantage of the MC method with real users is that the system learns gradually with real noise. Consequently, it is more accurate than the DP methods with simulated users.

4. Implementation Issues

In the implemented system, users call the ticket reservation system and state their requests, at the end of each episode of dialogues, users are asked to give their judgment about the conversation. This judgment is used for explicit rewarding in addition to the automatic implicit rewards incorporated in this system. University students are asked to call the system.

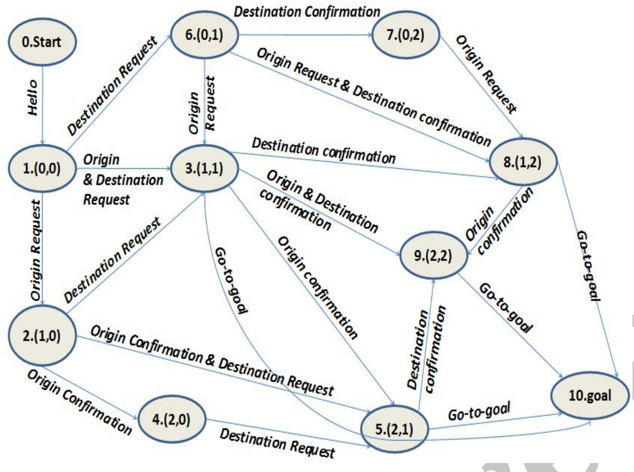


Figure 3. State-action block diagram

Table 1. Reward functions of the system

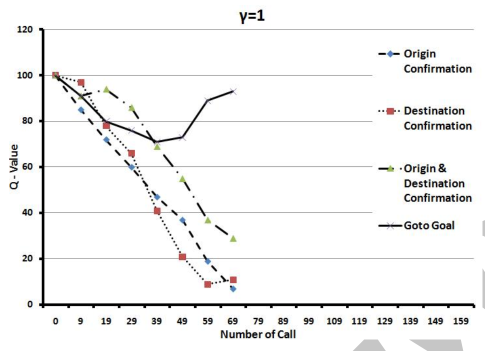
A non selected action with the "YES" judgment at the end of episode	-3
A selected action with the "YES" judgment at the end of episode	+5
A non selected action with the "NO" judgment at the end of episode	+2
A selected action with the "NO" judgment at the end of episode	-2
A confirmation action selected with the "YES" user judgment	-4
A confirmation action selected with the "NO" user judgment	+4

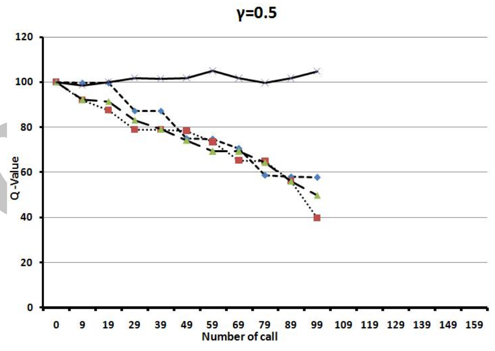
The ASR used is a large vocabulary (2000 words) speaker-independent continuous Persian speech recognition engine. Context independent hidden Markov models are trained for each phoneme. The language model used is a trigram model. The NLU uses a context-free grammar that is represented with partial robust parsing. This grammar has 400 rules and 64 non-terminals and is designed for analyzing probable input sentences. Recorded sentences are used by the SLG.

5. Experimental Results

In our experiments, the implicit reward criterion is unchanged and the effect of explicit reward is controlled by changing the γ factor. We intend to evaluate the effect of γ on the number of required training dialogues. Increasing the γ factor results in faster learning while it may cause some parts of the on-line training data to be ignored. On the other hand, reducing the γ value slows down the learning process while it allows the system to use more comprehensive training data. At first, γ is sequentially set to 1.0, 0.8 and 0.5. The convergence to the optimal policy happens when the Q-value of a single action within a state surpasses significantly the Q-value

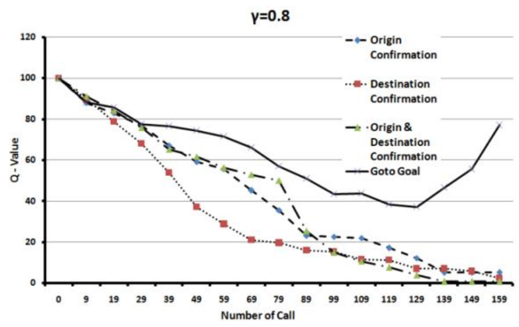
of all other actions of the same state. The important point is that if the Q-value of the winning action within a state remains constant during the learning process, it means that the learning process is not functioning correctly. In other words, in order to make sure the learning procedure is working is that a minimum is obtained in the Q-value curve of the winning action before the final convergence (Figs. 4a, 5a, 5b). For γ=1.0 and γ=0.8 the size of the provided training data was enough for convergence but for γ=0.5 it wasn't. Therefore, the value of 0.85 for γ was also used. According to the learning results for all states, the speed of converging to the optimal policy decreases as the γ value reduces and the accuracy of convergence with few training dialogues is reduced as the γ factor increases. Figs. 4a, 4b, 5a, and 5b show the graphs of variations of Q-value versus the number of calls for state "3(1,1)" that uses both implicit and explicit rewards. At the end of learning, the action mapped to the state is the Go-to-goal for all values of γ. For γ=0.5 the effect of explicit reward cannot be seen after 100 dialogues as shown in Fig. 4b. For γ=1.0, the best action changes during learning and the best action is chosen incorrectly when few training dialogues are used (Fig. 4a). For γ=0.8, however, the number of training dialogues increases and the best action keeps the highest Q-value during the whole learning process as shown in Fig. 5a. Setting γ=0.85 results in similar correct results of γ=0.8 with about 40 less dialogue sessions.

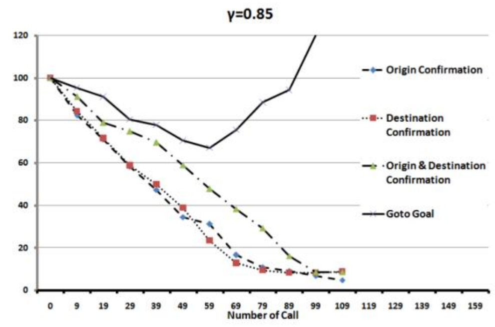




(a) (b)

Figure 4. Variation of Q-value for state 3(1,1) using a: γ =1.0, and b: γ=0.5.





(a) (b)

Figure 5. Variation of Q-value for state 3(1,1) using a: γ=0.8, and b: γ=0.85.

6. Summary and Conclusion

The work presented in this paper is the first research on learning spoken dialogue systems for the Persian language. Model-free reinforcement learning was used for training the dialogue management unit in SDS. Real users for providing on-line training data were also employed during the learning process. The effect of discount factor (γ)variations on convergence speed and accuracy of the learning process was examined. Learning with γ=0.5 or lower resulted in ignoring the user judgment and increasing the effect of implicit reward on learning.

Learning undiscounted, γ=1.0, with few training data resulted in getting to improper situations during the first steps of learning because users would judge based on final states residing in their memory. Finally, the optimal γ value obtained on the Persian language with on-line learning and real training data was between 0.8 and 0.85. For these γ values, the final action selected at the end was the best action throughout the learning process with the difference that γ=0.85 resulted in faster convergence compared to γ=0.8.

References

[1] J Schatzmann, B Thomson, and SJ Young, "Error Simulation for Training Statistical Dialogue Systems", *In ASRU 07, Kyoto*, Japan, 2007.

[2] F. Lef`evre_, M. Gaˇsi´c, F. Jurˇc´ıˇcek, S. Keizer, F. Mairesse, B. Thomson, K. Yu and S. Young, "k-Nearest Neighbor Monte-Carlo Control Algorithm for POMDP-based Dialogue Systems", SIGDIAL: *The 10th annual meeting of the special interest group in discourse and dialogue*, PP. 272-275, 2009.

[3] JD Williams and SJ Young, "Scaling POMDPs for Spoken Dialog Management", *IEEE Trans. on Audio, Speech and Language Processing*, 15(7): PP. 2116-2129, 2007.

[4] JD Williams, P Poupart, and SJ Young, "Partially Observable Markov Decision Processes with Continuous Observations for Dialogue Management", *Proc. Sig Dial Workshop on Discourse and Dialogue*, Lisbon, Portugal, PP. 192-217, 2005.

[5] Cheongjae Lee and Gary Geunbae Lee, "Robust Dialog Management with n-best Hypotheses Using Dialog Example and Agenda", *Proc. Association for Computational Linguistics Human Language Technologies (ACL-HLT)*, Columbus, Ohio, PP. 630-637 2008.

[6] E Horvitz and T Paek, "A Computational Architecture for Conversation", *Proc 7th International Conference on User Modeling (UM)*, Banff, Canada, pages 201–210, 1999.

[7] B Zhang, Q Cai, J Mao, and B Guo, "Planning and Acting under Uncertainty: A New Model for Spoken Dialogue System", *Proc Conf on Uncertainty in Artificial Intelligence (UAI)*, Seattle, Washington, pages 572–579, 2001.

[8] B Thomson, J Schatzmann, and SJ Young, "Bayesian Update of Dialogue State for Robust Dialogue Systems", *Proc Int Conf Acoustics Speech and Signal Processing ICASSP*, Las Vegas, USA, 2008.

[9] JD Williams, "Applying POMDPs to Dialog Systems in the Troubleshooting Domain", *NAACL HLT Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, Rochester, New York, USA, PP. 1–8, 2007.

[10] B Zhang, Q Cai, J Mao, E Chang, and B Guo, "Spoken Dialogue Management as Planning and Acting under Uncertainty", *Proc Eurospeech, Aalborg*, Denmark, pp.: 2169–2172, 2001.

[11] SJ Young, JD Williams, J Schatzmann, MN Stuttle, and K Weilhammer, "The Hidden Information State Approach to Dialogue Management", *Technical Report CUED/F-INFENG/TR. 544*, Cambridge University Engineering Department, 2006.

[12] H Higashinaka, M Nakano, and K Aikawa, "Corpus-based Discourse Understanding in Spoken Dialogue Systems", *Proc Association for Computational Linguistics (ACL)*, Sapporo, Japan, 2003.

[13] Heriberto Cuay´ahuitl, Steve Renals, Oliver Lemon, and Hiroshi Shimodaira, "Reinforcement Learning of Dialogue Strategies with Hierarchical Abstract Machines", *Proc Workshop on Spoken Language Technologies (SLT)*, Aruba, PP. 182 185, 2006.
[14] M Denecke, K Dohsaka, and M Nakano, "Learning Dialogue Policies Using State Aggregation in Reinforcement Learning", *Pro Intl Conf on Spoken Language Processing (ICSLP)*, Jeju, Korea, PP. 325–328, 2004.
[15] TH Bui, "Toward Affective Dialogue Management Using Partially Observable Markov Decision Processes", *PhD thesis*, University of Twente, Enschede, October 2008.
[16] Jason D Williams, "Demonstration of a POMDP Voice Dialer", *Proc Demonstration Session of Association for Computational Linguistics Human Language Technologies (ACL-08: HLT)*, Columbus, Ohio, PP. 1-4, June, 2008.
[17] B Thomson, J Schatzmann, K Welhammer, H Ye, and SJ Young, "Training a Real-world POMDP Based Dialog System", *Proc Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT) Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies, Rochester*, New York, USA, PP. 9–17, 2007.
[18] E Levin and R Pieraccini, "Value-based Optimal Decision for Dialog Systems", *Proc Workshop on Spoken Language Technologies (SLT)*, Aruba, PP. 198–201, 2006.
[19] Cheongjae Lee, Sangkeun Jung, Jihyun Eun, Minwoo Jeong, and Gary Geunbae Lee, "A Situation Based Dialogue Management Using Dialogue Examples", *Proc Intl Conf on Acoustics Speech and Signal Processing (ICASSP)*, Toulouse, France, 2006.
[20] M. Gaˇsi´c, S. Keizer, B. Thomson, F. Mairesse, J. Schatzmann, K. Yu, and S. Young, "Training and Evaluation of the HIS-POMDP Dialogue System in Noise",*Proc. 9th SigDial*, Columbus, Ohio, 2008.
[21] JD Williams and SJ Young, "Partially Observable Markov Decision Processes for Spoken Dialog Systems", *Computer Speech and Language*, vol. 21, no. 2, PP. 393-422, 2007.
[22] Trung H. BUI, Mannes POEL, Anton NIJHOLT, and Job ZWIERS, "A POMDP Approach to Affective Dialogue Modeling", Fundamentals of Verbal and Nonverbal Communication, 2007.
[23] Tim Paek and David Maxwell Chickering, "The Markov Assumption in spoken dialogue management", *Proc of the 6th SigDial Workshop on Discourse and Dialogue*, Lisbon, Portugal September 2-3, 2005.
[24] S Singh, D Litman, M Kearns, M Walker, "Optimizing Dialogue Management with Reinforcement Learning NJF", *Journal of Artificial Intelligence Research*, 2002.
[25] Richard S. Sutton, Andrew G. Barto, "Reinforcement Learning: An Introduction", Cambridge, MA: MIT Press, 1998.
[26] Lemon and O. Pietquin, "Machine Learning for Spoken Dialogue Systems", *Proc. of Interspeech*, 2007.
[27] JD Williams, "Partially Observable Markov Decision Processes for Spoken Dialogue Management", *The doctoral thesis*, University of Cambridge, 2006.