ON-LINE POLICY OPTIMISATION OF BAYESIAN SPOKEN DIALOGUE SYSTEMS VIA HUMAN INTERACTION

M. Gašić, C. Breslin, M. Henderson, D. Kim, M. Szummer, B. Thomson, P. Tsiakoulis and S. Young

Cambridge University Engineering Department

ABSTRACT

A partially observable Markov decision process has been proposed as a dialogue model that enables robustness to speech recognition errors and automatic policy optimisation using reinforcement learning (RL). However, conventional RL algorithms require a very large number of dialogues, necessitating a user simulator. Recently, Gaussian processes have been shown to substantially speed up the optimisation, making it possible to learn directly from interaction with human users. However, early studies have been limited to very low dimensional spaces and the learning has exhibited convergence problems. Here we investigate learning from human interaction using the Bayesian Update of Dialogue State system. This dynamic Bayesian network based system has an optimisation space covering more than one hundred features, allowing a wide range of behaviours to be learned. Using an improved policy model and a more robust reward function, we show that stable learning can be achieved that significantly outperforms a simulator trained policy.

Index Terms— dialogue systems, POMDP, Gaussian process

1. INTRODUCTION

The statistical approach to dialogue modelling has been proposed as a means of building domain independent dialogue systems, trainable from data and robust to speech understanding errors [1, 2]. If the dialogue state satisfies the Markov property, the dialogue can be modelled as a Markov decision process (MDP) [1] and reinforcement learning (RL) algorithms can be used for policy optimisation [3]. Since RL is typically slow, policy training in the past has normally required the use of a simulated user [4], and where learning from direct human-computer interaction has been attempted, as in the NJFun system [5], the dialogue systems have been constrained and reliant on a significant amount of built-in expert knowledge.

A recent trend has been to move to the partially observable Markov decision process (POMDP) in order to provide increased robustness to errors in speech understanding [6, 7]. The POMDP-based approach to dialogue management maintains a distribution over every possible dialogue state, the *belief state*, and based on that distribution the system chooses the action that gives the highest expected reward. Various approximations allow this method to be used for building real world dialogue systems [8, 9]. However, POMDP systems are more complex than MDP systems and they typically require $\mathcal{O}(10^5)$ to $\mathcal{O}(10^6)$ dialogues [10, 11] to train using conventional RL algorithms. This makes it prohibitive to train in direct interaction with human users and the use of a simulated user appears essential despite the disadvantages of significant additional development costs and potential discrepancies between real and simulated user behaviour.

Gaussian process (GP) based reinforcement learning [12] has been recently applied to POMDP dialogue policy optimisation in order to exploit the correlations between different belief states and thus speed up the learning process [13]. A Gaussian process also provides an estimate of the uncertainty in the approximation, which can be used to obtain more efficient learning strategies [14]. Furthermore, recent innovations in crowd-sourcing and global telephone call routing via VoIP now allow large numbers of users to be recruited at low cost for large-scale training and testing of dialogue systems [15].

An initial exploration of the use of GP based RL to allow direct policy optimisation with real users was reported in [16]. However, this study used the Hidden Information State system [8], which had only a four dimensional summary space and limited ability to model real user behaviour. Furthermore, the learning process exhibited convergence problems which were thought to be due to inconsistent user feedback making the rewards unreliable.

Here, we investigate learning from human interaction using the Bayesian Update of Dialogue State (BUDS) system. This dynamic Bayesian network based system has an optimisation space covering more than one hundred mixed continuous and discrete features, allowing a wide range of behaviours to be learned. In addition, an improved stochastic policy and a more robust reward function have been introduced. These enable stable learning to be achieved without the substantial cost of building a user simulator.

The principal contributions of this paper are therefore to show that using a robust reward function, on-line learning with real users is practical even for systems with large parameter spaces and to demonstrate that training on real users as compared to a user simulator can yield significantly improved performance. A second contribution is the presentation of our improved Gaussian process-based policy model and an evaluation of its performance trained with a user simulator and testing both with the simulator and with real users.

The remainder of the paper is organised as follows. In the next section we give an overview of Gaussian processes in POMDP based dialogue optimisation. Then in Section 3 we present experimental results using the BUDS dialogue manager. We examine GP policy performance on a simulated user, both during training and testing. We then evaluate two simulator trained policies with real users. Finally, we show that a policy can be trained from direct interaction with real users which converges smoothly and which significantly improves on the performance of the simulator trained policies. Section 4 gives conclusions.

2. GAUSSIAN PROCESSES IN POMDP OPTIMISATION

The role of a dialogue policy π is to map each belief state $\mathbf{b} \in \mathcal{B}$ into an action a so as to maximise the expected cumulative reward, a measure of how good the dialogue is. While it is possible to apply

This work was partly supported by PARLANCE (www.parlance-project.eu), an EU Seventh Framework Programme project (grant number 287615).

policy optimisation directly on the belief space $\mathcal{B}$ [17], it is typically mapped into a smaller scale summary space $\mathcal{C}$. The role of the policy π is then to map each summary state $\mathbf{c}$ into a summary action a from the summary action space $\mathcal{A}$. Once a summary action is found it is heuristically mapped into the master action that the system finally takes.

The expected cumulative reward is defined by the Q-function as:

$$Q(\mathbf{c},a) = E_\pi\left(\sum_{\tau=t+1}^{T}\gamma^{\tau-t-1}r_\tau|c_t=\mathbf{c},a_t=a\right), \quad (1)$$

where r_τ is the reward obtained at time τ, T is the dialogue length and γ is the discount factor, $0<\gamma\leq 1$. Optimising the Q-function is then equivalent to optimising the policy π.

A Gaussian process (GP) is a non-parametric Bayesian probabilistic model that can be used for function regression [18]. It is fully defined by a mean and a kernel function which defines prior function correlations. The kernel function is crucial for obtaining good posterior estimates with just a few observations.

GP-Sarsa is an on-line RL algorithm that models the Q-function as a Gaussian process [19], $Q(\mathbf{c},a)\sim\mathcal{GP}\left(0,k((\mathbf{c},a),(\mathbf{c},a))\right)$ where the kernel $k(\cdot,\cdot)$ is factored into separate kernels over the summary state and action spaces $k_\mathcal{C}(\mathbf{c},\mathbf{c}')k_\mathcal{A}(a,a')$. For a sequence of summary state-action pairs $\mathbf{C}_t=[(\mathbf{c}^0,a^0),\ldots,(\mathbf{c}^t,a^t)]^\mathsf{T}$ visited in a dialogue and the corresponding observed immediate rewards $\mathbf{r}_t=[r^1,\ldots,r^t]^\mathsf{T}$, the posterior of the Q-function for any summary state-action pair $(\mathbf{c},a)$ is defined by the following:

$$\begin{aligned}
&Q(\mathbf{c},a)|\mathbf{r}_t,\mathbf{C}_t\sim\mathcal{N}(\overline{Q}(\mathbf{c},a),\mathrm{cov}((\mathbf{c},a),(\mathbf{c},a))),\\
&\overline{Q}(\mathbf{c},a)=\mathbf{k}_t(\mathbf{c},a)^\mathsf{T}\mathbf{H}_t^\mathsf{T}(\mathbf{H}_t\mathbf{K}_t\mathbf{H}_t^\mathsf{T}+\sigma^2\mathbf{H}_t\mathbf{H}_t^\mathsf{T})^{-1}\mathbf{r}_t,\\
&\mathrm{cov}((\mathbf{c},a),(\mathbf{c},a))=k((\mathbf{c},a),(\mathbf{c},a))\\
&\quad-\mathbf{k}_t(\mathbf{c},a)^\mathsf{T}\mathbf{H}_t^\mathsf{T}(\mathbf{H}_t\mathbf{K}_t\mathbf{H}_t^\mathsf{T}+\sigma^2\mathbf{H}_t\mathbf{H}_t^\mathsf{T})^{-1}\mathbf{H}_t\mathbf{k}_t(\mathbf{c},a)\\
&\mathbf{H}_t=\begin{bmatrix}1&-\gamma&\cdots&0&0\\0&1&\cdots&0&0\\\vdots&\ddots&\ddots&\vdots&\vdots\\0&\cdots&0&1&-\gamma\end{bmatrix},\\
&\mathbf{k}_t(\mathbf{c},a)=[k((\mathbf{c}^0,a^0),(\mathbf{c},a)),\ldots,k((\mathbf{c}^t,a^t),(\mathbf{c},a))]^\mathsf{T},\\
&\mathbf{K}_t=[\mathbf{k}_t((\mathbf{c}^0,a^0)),\ldots,\mathbf{k}_t((\mathbf{c}^t,a^t))]
\end{aligned} \quad (2)$$

where σ^2 is the additive noise. The noise parameter controls how much variability in the Q-function estimate do we expect during the process of learning.

A basic policy model can be defined ϵ-greedily based on the Q-function posterior from Eq. 2 in the following way:

$$\pi(\mathbf{c})=\begin{cases}\arg\max_a\overline{Q}(\mathbf{c},a) & \text{with prob } 1-\epsilon,\\ \text{random} & \text{with prob } \epsilon.\end{cases} \quad (3)$$

However, the Q-function posterior in Eq. 2 defines a Gaussian distribution for every summary state-action pair. Thus, when a new summary state $\mathbf{c}$ is encountered, for each action $a\in\mathcal{A}$, there is a Gaussian distribution $Q(\mathbf{c},a)\sim\mathcal{N}(\overline{Q}(\mathbf{c},a),\mathrm{cov}((\mathbf{c},a),(\mathbf{c},a))))$. Sampling from these Gaussian distributions gives a set of Q-values for each action $\{Q(\mathbf{c},a):a\in\mathcal{A}\}$ from which the action with the highest sampled Q-value can be selected:

$$\pi(\mathbf{c})=\arg\max_a\{Q(\mathbf{c},a):a\in\mathcal{A}\}. \quad (4)$$

In this way, the stochastic model of the Q-function is effectively transformed into a stochastic policy model, which can be optimised to maximise the reward [20, 16, 17]. Gaussian process estimation however has an inherent problem that the estimate of the variance depends only on the number of points visited during learning. During reinforcement learning we expect to visit the same points many times, especially for large dimensional spaces. Therefore, it is necessary to scale the variances during training $\mathrm{cov}((\mathbf{c},a),(\mathbf{c},a))\approx\eta^2\mathrm{cov}((\mathbf{c},a),(\mathbf{c},a))$ where η is the scaling factor.

Fig. 1. Episodic GP-Sarsa

Initialise $\mathbf{c}$, Choose a arbitrary
$\mathbf{C}_0\leftarrow(\mathbf{c},a)$, $\mathbf{r}_0=[]$
$\mathbf{K}_0\leftarrow[k((\mathbf{b},a),(\mathbf{b},a))]$, $\mathbf{H}_0\leftarrow\begin{bmatrix}1&-\gamma\end{bmatrix}$
for each episode **do**
  for each step in the episode **do**
    Take action a, observe reward r', update summary state $\mathbf{c}'$
    if non-terminal step **then**
      Choose new action $a'\leftarrow\pi(\mathbf{c}')$ (Eq. 3 or 4)
      $\mathbf{C}_{t+1}\leftarrow\begin{bmatrix}\mathbf{C}_t&(\mathbf{c}',a')\end{bmatrix}$
      $\mathbf{K}_{t+1}\leftarrow\begin{bmatrix}\mathbf{K}_t&\mathbf{k}_t(\mathbf{b}',a')\\\mathbf{k}_t(\mathbf{b}',a')&k(\mathbf{b}',a',\mathbf{b}',a')\end{bmatrix}$
      $\mathbf{H}_{t+1}\leftarrow\begin{bmatrix}\mathbf{H}_t&\mathbf{0}\\\mathbf{u}^\mathsf{T}&-\gamma\end{bmatrix}$, where $\mathbf{u}=\begin{bmatrix}\mathbf{0}&1\end{bmatrix}^\mathsf{T}$
    else
      $\mathbf{C}_{t+1}\leftarrow\mathbf{C}_t$, $\mathbf{K}_{t+1}\leftarrow\mathbf{K}_t$
      $\mathbf{H}_{t+1}\leftarrow\begin{bmatrix}\mathbf{H}_t\\\mathbf{u}^\mathsf{T}\end{bmatrix}$, where $\mathbf{u}=\begin{bmatrix}\mathbf{0}&1\end{bmatrix}^\mathsf{T}$
    end if
    $\mathbf{r}_{t+1}\leftarrow\begin{bmatrix}\mathbf{r}_t&r'\end{bmatrix}$
    Update $Q^\pi|\mathbf{r}_{t+1},\mathbf{C}_{t+1}$ (Eq. 2)
    if non-terminal step **then**
      $\mathbf{c}\leftarrow\mathbf{c}'$, $a\leftarrow a'$
    end if
  end for
end for

Due to the matrix inversion in Eq. 2, the computational complexity of calculating the posterior is $O(t^3)$, where t is the number of data points visited during all training dialogues. Hence, we use the kernel span sparsification method described in [21] to reduce the computational complexity and make the algorithm tractable. This method can be elegantly incorporated into the on-line GP-Sarsa algorithm [19].

To apply GP policy optimisation, a kernel function must be defined on both the summary state space $\mathcal{C}$ and the summary action space $\mathcal{A}$. Since the summary space typically consists of both continuous and discrete values, the kernel function is defined as the sum of the kernels of the continuous and the discrete values. For the kernel on the continuous parts of the space we use the Gaussian kernel function, defined as:

$$k_\mathcal{C}(\mathbf{c},\mathbf{c}';p,\sigma_k)=p^2\exp\left(-\frac{\|\mathbf{c}-\mathbf{c}'\|^2}{2\sigma_k^2}\right), \quad (5)$$

where σ_k determines how close the points have to be for the values of the function to be correlated, and p defines the prior variance at each data point since $k(\mathbf{c},\mathbf{c})=p^2$. The main advantage of the Gaussian kernel is that it is a dot product of an infinite vector of feature functions [18], which gives it the potential to model covariances better. For the kernel on discrete parts of the space as well as for the kernel on summary action space, we use the δ-kernel, defined as:

$$k(a,a')=1-\delta_a(a'). \quad (6)$$

Since the algorithm only depends on the kernel function, a space of arbitrary size can be used provided that the kernel function can be efficiently calculated.

3. EXPERIMENTS

3.1. Bayesian Update of Dialogue State dialogue manager

The Bayesian Update of Dialogue State (BUDS) dialogue manager is a POMDP-based dialogue manager [9] which factorises the dialogue state into conditionally dependent elements. These elements are arranged into a dynamic Bayesian network, which allows for their marginal probability distributions to be updated during the dialogue. Thus, the belief state of the BUDS dialogue manager consists of the marginal posterior probability distribution over hidden nodes in the Bayesian network. The hidden nodes in the BUDS system consist of the history nodes and the goal nodes for each concept in the dialogue. For instance in a restaurant information domain these include *area*, *food-type*, *address*. The history nodes define possible dialogue histories for a particular concept, eg. *system-informed*, *user-informed*. The goal nodes define possible values for a particular concept, eg. *Chinese*, *Indian*.

The summary space of the BUDS system consists of both continuous and discrete features of the belief state. The continuous features represent the entropy of the marginal distribution for each hidden node, as well as the probability of the highest probable value and the probability of the next value for each hidden node. In addition, for each hidden node, there is a binary feature indicating whether the user does not care about the value of that concept. There are also global features which refer to the whole time slice of the dynamic Bayesian network. Some of them correspond to history nodes, e.g. if the highest probability method that the user used to inform the system was *by name of the venue* or if the highest probability discourse act used was *repeat*. Also, there is a count of the number of top probability goal values that are greater than 0.8. Finally, there is a list which defines the order in which the goal slots can be accepted.

3.2. TopTable domain

The TopTable domain consists of the restaurants in Cambridge, UK that are automatically extracted from the TopTable web service [22]. There are about 150 restaurants and each restaurant has 8 attributes – slots. This results in the belief space that consists of 25 concepts where each concepts takes from 3 to 150 values and each value has a probability in $[0, 1]$ and the summary space consists of 129 features. The summary action space consists of 16 summary actions.

3.3. The agenda-based simulated user

The agenda-based user simulator [23, 24] factorises the user state into an *agenda* and a *goal*. The goal ensures that the user simulator exhibits consistent, goal-directed behaviour. The role of the agenda is to elicit the dialogue acts that are needed for the user simulator to fulfil the goal.

In addition, an error model was used to add confusions to the simulated user input such that it resembles those found in real data [25]. The length of the N-best list was set to 10 and the confusion rate was set to 15%, which means that 15% of time the true hypothesis is not in the N-best list. An intermediate experimentation showed that these confusion rates are typical of real data.

The reward function was set to give a reward of 20 for successful dialogues, zero otherwise. In addition, 1 is deducted for each dialogue turn to encourage shorter dialogues. The discount factor γ is set to 1 and the dialogue length is limited to 30 turns.

3.4. Amazon MTurk service

In order to evaluate or train a policy with real people we used crowdsourcing via the Amazon Mechanical Turk service in a set-up similar to [15]. The BUDS dialogue manager was incorporated in a live telephone-based spoken dialogue system. The Mechanical Turk users were assigned specific tasks in the TopTable domain. They were asked to find restaurants that have particular features as defined by the given task. To elicit more complex dialogues, the users were sometimes asked to find more than one restaurant, and in cases where such a restaurant did not exist they were required to seek an alternative, for example find a Chinese restaurant instead of a Vietnamese one. After each dialogue the users filled in a feedback form indicating whether they judged the dialogue to be successful or not. Based on that binary rating, the subjective success was calculated as well as the average reward. The objective rating can also be obtained by comparing the system outputs with the predefined task.

3.5. Policy design

The GP-Sarsa algorithm was configured with Gaussian kernel parameters $\sigma_k = 5$ and $p = 4$, and the noise parameter σ (see Eq. 2) was set at 5. Intermediate experimentation showed that these provide a good configuration. We compared three stochastic models with different variance scale factors by setting $\eta = 1$, $\eta = 2$ and $\eta = 3$[1]. Results during training and testing are given in Figs. 2 and 3 respectively. It is clear that increasing the variance scale factor η has a significant influence on performance. Without scaling, the system learns quickly, but also converges to a local optimum. Scaling the variance avoids this problem. Fig. 3 shows that when exploration is disabled[2], the stochastic policy model provides only small improvements in the eventual performance compared to the standard ϵ-greedy policy. However, as shown in Fig. 2 during training when exploration is enabled, the performance is significantly better. This is particularly valuable when training in interaction with real users since it is important during optimisation to avoid distressing users with poor performance and unpredictable behaviour.

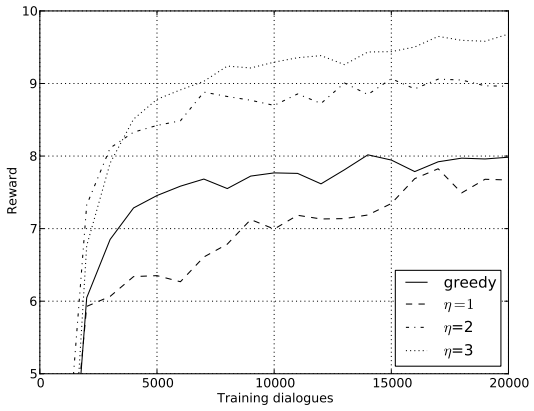

Fig. 2. Different policy models – training

In order to examine the performance of the GP policy on real users, we conducted an evaluation using the Amazon Mechanical

[1]Higher values did not produce an improvement in performance.

[2]By setting $\epsilon = 0$ in the ϵ-greedy model and $\eta = 1$ in the stochastic model.

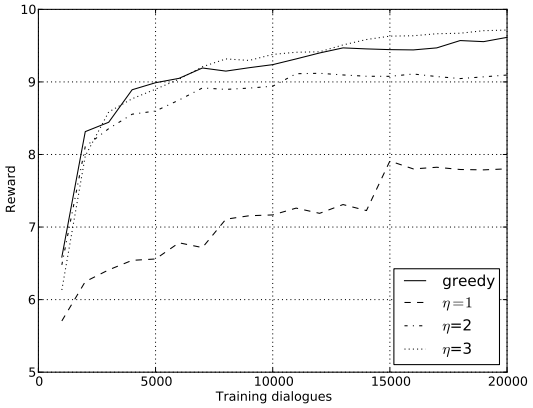

Fig. 3. Different policy models – testing

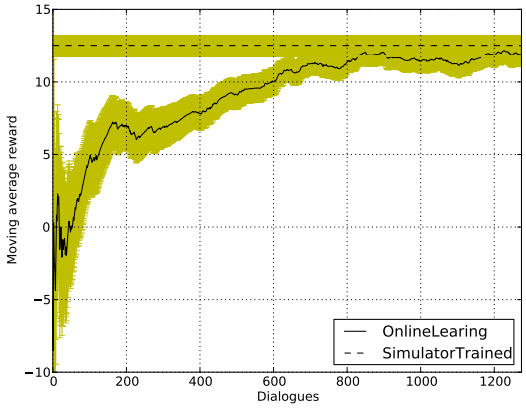

Fig. 4. On-line learning with real users

Turk service. We compared two policies: a partially trained GP policy trained with $10,000$ simulated dialogues, and a fully trained GP policy trained using $100,000$ simulated dialogues. The results are given in Table 1 and they suggest that it is not necessary to train the GP policy with more than $10,000$ dialogues. Indeed, there is some evidence that performance actually degrades with further training, perhaps due to over-fitting on the simulator.

Table 1. Human evaluation of simulator trained policies

	GP-PartlyTrained	GP-FullyTrained
#of dialogues	400	397
Reward	12.5 ± 0.3	11.6 ± 0.4
Success rate (%)	93.5 ± 1.2	91.2 ± 1.4

3.6. Training in interaction with humans

In this section, we investigate on-line optimisation of the Bayesian Update of Dialogue State dialogue system using GP-Sarsa and the stochastic policy model described in the previous section with η set to 3. As noted in the introduction, our earlier attempts at on-line policy optimisation with real users recruited via crowd-sourcing had encountered problems in achieving consistent user feedback for driving the reward function [16]. In an attempt to mitigate these problems, we now utilise a validation procedure to check the subjective feedback from users.

To train a policy from scratch, human users were assigned specific tasks in the TopTable domain using the Amazon Mechanical Turk service as described in Section 3.4. At the end of each call, users were asked to press 1 if they were satisfied (i.e. believed that they had been successful in fulfilling the assigned task) and 0 otherwise. Since experience has previously shown that this feedback can be unreliable, at the end of each call the objective success was also calculated by comparing the predefined task given to the user with the information that the system provided. The dialogue was then only included in the GP-sarsa optimisation if the user rating agreed with the objective measure of success. This *filtered* reward function was motivated by the findings in [16] where it was shown that a policy trained on a corpus filtered in the same way substantially outperformed a policy trained on the full corpus.

The performance achieved during on-line learning on the initial 1274 dialogues was compared to the performance of the policy trained on $10,000$ dialogues using the simulated user (GPPartlyTrained policy from the previous section). The results are given in Fig. 4 where the moving average reward is given for a window of 400 dialogues along with its one standard error. Out of 1274 dialogues 1081 had the same subjective and objective success and they took part in learning. It can be seen from Fig. 4 that the policy trained on-line reaches the same asymptotic reward as the policy trained using the simulated user but in substantially fewer training dialogues. Moreover, the learning curve is smooth and it does not exhibit inconsistencies as was the case in [16].

Finally, the policy trained on 1000 dialogues with real users was tested under the same conditions as for the simulator trained policies (i.e. the scaling factor $\eta = 1$). Comparing Table 2 with Table 1, it can be seen that the policy trained on real users significantly outperforms the simulator trained policies both in terms of success rate and average reward.

Table 2. Human evaluation of the policy trained online

	GP-OnlineTrained
#of dialogues	410
Reward	13.4 ± 0.3
Success rate (%)	96.8 ± 0.9

4. CONCLUSION

This paper has described how Gaussian processes can be deployed to rapidly optimise the policy of a Bayesian network based dialogue system in direct interaction with human users. To minimise the impact of disruptive system exploration during the early stages of learning, a novel stochastic policy is used and to ensure smooth learning, a robust reward function is applied which filters user feedback via a validation procedure. The combination of these two innovations provides smooth convergence towards an optimal policy within 1000 training dialogues. Furthermore, when tested in human trials, the policy trained with real users significantly outperforms policies trained with a user simulator. This new optimisation framework avoids the need for building costly application-dependent user simulators and yields improved run-time performance, opening the way for building adaptive dialogue systems which can learn to handle new domains on-line via interaction with real users. Future work will include developing adaptation strategies where a policy trained on simulated user can be adapted to real people.

5. REFERENCES

[1] E Levin, R Pieraccini, and W Eckert, "Using Markov Decision Processes for Learning Dialogue Strategies," in *Proceedings of ICASSP*, 1998.

[2] SJ Young, "Talking to Machines (Statistically Speaking)," in *Proceedings of ICSLP*, 2002.

[3] RS Sutton and AG Barto, *Reinforcement Learning: An Introduction*, Adaptive Computation and Machine Learning. MIT Press, Cambridge, Massachusetts, 1998.

[4] E Levin, R Pieraccini, and W Eckert, "A Stochastic Model of Human-Machine Interaction for Learning Dialog Strategies," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 11–23, 2000.

[5] S Singh, DJ Litman, M Kearns, and M Walker, "Optimizing Dialogue Management with Reinforcement Learning," *Journal of Artificial Intelligence Research*, vol. 16, pp. 105–133, 2002.

[6] N Roy, J Pineau, and S Thrun, "Spoken dialogue management using probabilistic reasoning," in *Proceedings of ACL*, 2000.

[7] B Zhang, Q Cai, J Mao, E Chang, and B Guo, "Spoken Dialogue Management as Planning and Acting under Uncertainty," in *Proceedings of Eurospeech*, 2001.

[8] Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu, "The Hidden Information State model: A practical framework for POMDP-based spoken dialogue management," *Computer Speech and Language*, vol. 24, no. 2, pp. 150–174, 2010.

[9] B Thomson and S Young, "Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems," *Computer Speech and Language*, vol. 24, no. 4, pp. 562–588, 2010.

[10] M Gašić, S Keizer, F Mairesse, J Schatzmann, B Thomson, K Yu, and S Young, "Training and evaluation of the HIS-POMDP dialogue system in noise," in *Proceedings of SIGDIAL*, 2008.

[11] F Jurčíček, B Thomson, and S Young, "Natural actor and belief critic: Reinforcement algorithm for learning parameters of dialogue systems modelled as POMDPs," *ACM Transactions on Speech and Language Processing*, 2011.

[12] Y Engel, S Mannor, and R Meir, "Bayes Meets Bellman: The Gaussian Process Approach to Temporal Difference Learning," in *Proceedings of ICML*, 2003.

[13] M Gašić, F Jurčíček, S Keizer, F Mairesse, J Schatzmann, B Thomson, K Yu, and S Young, "Gaussian Processes for Fast Policy Optimisation of POMDP-based Dialogue Managers," in *Proceedings of SIGDIAL*, 2010.

[14] L. Daubigney, M. Gašić, S. Chandramohan, M. Geist, O. Pietquin, and S. Young, "Uncertainty management for online optimisation of a POMDP-based large-scale spoken dialogue system," in *Proceedings of Interspeech*, 2011.

[15] F Jurčíček, S Keizer, M Gašić, F Mairesse, B Thomson, K Yu, and S Young, "Real user evaluation of spoken dialogue systems using Amazon Mechanical Turk," in *Proceedings of Interspeech*, 2011.

[16] M Gašić, F Jurčíček, B Thomson, K Yu, and S Young, "Online policy optimisation of spoken dialogue systems via live interaction with human subjects," in *Proceedings of ASRU*, 2011.

[17] M Gašić, M. Henderson, B Thomson, P. Tsiakoulis, and S Young, "Policy optimisation of POMDP-based dialogue systems without state space compression," in *Proceedings of SLT*, 2012.

[18] CE Rasmussen and CKI Williams, *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, Massachusetts, 2005.

[19] Y Engel, S Mannor, and R Meir, "Reinforcement learning with Gaussian processes," in *Proceedings of ICML*, 2005.

[20] M Geist and O Pietquin, "Managing Uncertainty within the KTD Framework," in *Proceedings of the Workshop on Active Learning and Experimental Design*, Sardinia (Italy), 2011.

[21] Y Engel, *Algorithms and Representations for Reinforcement Learning*, PhD thesis, Hebrew University, 2005.

[22] TopTable, "TopTable," 2012, *https://www.toptable.com*.

[23] J Schatzmann, *Statistical User and Error Modelling for Spoken Dialogue Systems*, Ph.D. thesis, University of Cambridge, 2008.

[24] S Keizer, M Gašić, F Jurčíček, F Mairesse, B Thomson, K Yu, and S Young, "Parameter estimation for agenda-based user simulation," in *Proceedings of SIGDIAL*, 2010.

[25] B Thomson, M Gašić, M Henderson, P Tsiakoulis, and S Young, "N-Best error simulation for training spoken dialogue systems," in *Proceedings of SLT*, 2012.