Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 00 (2014) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

6th International conference on Intelligent Human Computer Interaction, IHCI 2014

Imitating dialog strategies under uncertainty

Johannes Fonfara[a,*], Sven Hellbach[a], Hans-Joachim Böhme[a]

[a] *University of Applied Sciences Dresden, Friedrich-List-Platz 1, 01069 Dresden, Germany*

Abstract

We consider human-robot interaction involving a service robot and many different users in a public environment. The task is to learn a dialog policy that deals with changing user goals, can act under uncertainty, and is easy to apply in practice. Unlike reinforcement-learning-based systems, our simulator-free approach avoids common problems such as reward tuning and state space exploration: We apply imitation learning in order to mimic an expert's behavior based on a small number of Wizard-of-Oz experiments. A dynamic Bayesian Network is used to track hidden user goals. We evaluate our approach in a simulated environment and show that by using lifelong model updates it is possible to apply the expert's policy correctly even if the user behavior changes over time.

Peer-review under responsibility of the Scientific Committee of IHCI 2014.

Keywords: dialog system, imitation learning, lifelong learning, cognitive robotics, Bayesian network

1. Introduction

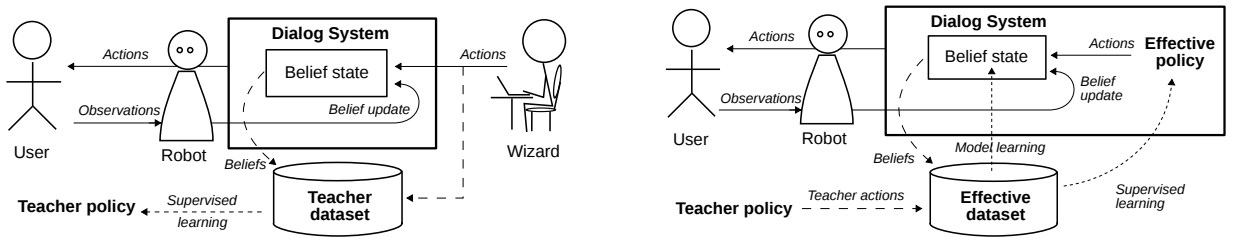
In the past decade, many dialog controllers have been successfully developed based on underlying (Partially Observable) Markov Decision Processes (POMDPs), with reinforcement learning being now the state of the art for finding dialog policies for problems with large state spaces[1]. The development of such a system usually comprises collecting a dialog corpus in Wizard-of-Oz experiments, building a user simulator base thereon, and have it interact with a learning dialog manager until a policy is found that maximizes some hand-crafted reward function (for related applications see, e.g.,[2,3,4,5]).

But not every dialog task is as simply described as "Fulfill all user goals in the least number of turns". For example in a museum tour guide scenario the task may be to entertain visitors, chat with them, and try not to bore them by presenting too much information. Not only would this scenario require a very complex and realistic user simulator, also the reward function must be hand-crafted in a trial-and-error manner until a satisfying policy is found that meets all the designer's demands. Conventional approaches explore the state space by interacting with a simulated user[6]. However, for human-robot dialogs (which are not necessarily goal-directed) it is very difficult to build a simulator that makes reasonable predictions of what a user's reaction would be in a rarely seen state, without making lots of psychoanalytical assumptions about the user's internal goals and beliefs. This renders exploration by simulation nearly impossible.

This paper addresses dialog scenarios where a user simulator is too complex to design because user behaviors vary widely, as is often the case in human-robot dialogs in public spaces. Our experience shows that, if in a Wizard-of-Oz

* Corresponding author.
E-mail address: fonfara@htw-dresden.de

Peer-review under responsibility of the Scientific Committee of IHCI 2014.



(a) The wizard demonstrates the interaction.

(b) The dialog controller can apply the policy autonomously.

Fig. 1. Overview of the learning task: Initially, the dialog is demonstrated in Wizard-of-Oz experiments and a teacher policy derived from the stored belief-action trajectories. At runtime, more dialog data is collected, from which model parameters are learned to improve belief updates while the effective policy is continuously adapted to the current user.

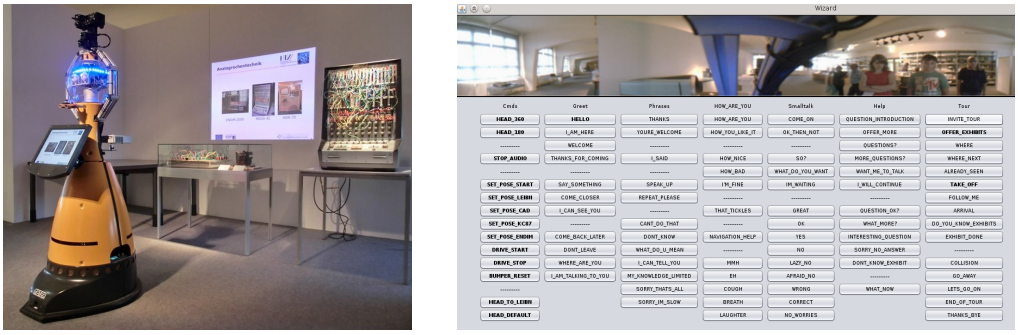
experiment we consequently apply the same interaction policy, large parts of the state space are never visited and therefore are irrelevant for policy learning. Furthermore, since we are already given a desirable policy by the wizard's demonstrations, it seems natural to imitate this behavior instead of exploring the state space in simulations made under vague assumptions.

Experimental scenario: The interaction considered in this paper takes place in a museum where a robot is deployed as an autonomous tour guide. Its task is to guide visitors through the exhibition, present information about the exhibits and answer their questions (for a detailed explanation, see Poschmann[7]). During Wizard-of-Oz experiments performed on the real system the remote operator had available a panel with an overwhelming multitude of buttons, each of which triggering a different speech act on the robot, as shown in Fig. 2(b). Essentially we aim to find a learning algorithm that can be trained simply by conducting Wizard-of-Oz experiments (even when the number of actions is very high), such that after a few demonstrations the system can generalize and apply the wizard's policy even in previously unseen states.

Lifelong learning approach: A possible problem with human-robot interaction is the varying behavior of users, posing difficulties to track the user state due to initially unknown system dynamics. While most state-of-the-art systems assume static model parameters, we attempt adapt to the user by using hand-crafted model parameters only for the initial dialog manager and let it interact with real users. Over time, the system collects more data and re-trains its model parameters based on observed dialogs. Thus, the dialog controller can recognize changes in the user behavior over time and still apply the wizard's policy correctly. The approach is outlined in Fig. 1. While we believe that reinforcement learning is an indispensable method for many goal-directed dialog problems, we show that our supervised imitation-learning approach works well for a dialog problem with about 10^5 states and 9 summary actions.

2. Related work

Imitation learning has been used in some fields of robotics, but mostly to clone human motion by motor control (e.g.[8,9]). Another early application was the autonomous car ALVINN[10] where a neural network was trained with road



(a) Our guide robot presents an exhibit

(b) Remote control panel used for Wizard-of-Oz experiments

Fig. 2. Set-up of the dialog scenario: A guide robot in a museum is set to interact with visitors.

images and a corresponding steering output. There are also model-based approaches such as inverse reinforcement learning [11] where the goal is to compute the teacher's hidden reward function from available demonstrations. Hence, a policy can be computed that optimizes the learned reward function and attains similar performance as the teacher.

A problem with inverse reinforcement learning is however that it strongly relies on behavioural features used to compare policies. Given a set of m teacher trajectories $\{s_0^{(i)}, s_1^{(i)}, ...\}_{i=1}^m$, a feature expectation μ_E is computed as $\mu_E = \frac{1}{m}\sum_{i=1}^m \sum_{t=0}^\infty \gamma^t \phi(s_t^{(i)})$ where $\phi(s)$ is a feature extractor and γ is a discount factor. However, features of dialog behavior of that form are difficult to find since it also matters how many times and at what time in the dialog an action was performed. For instance, in an uncertain situation the teacher may ask confirmation questions repeatedly, which may bias the performance.

Therefore in this paper we resort to supervised learning in order to map system beliefs to actions, and discuss the problem of missing data in unexplored states.

3. Dialog state tracking

Preliminaries: We cast the dialog system as a POMDP (without reward function) because it provides a principled framework that readily handles uncertainty [1]. A POMDP is defined by a set of states $\mathcal{S}$ from which an initial state s_0 is drawn, a set of system actions $\mathcal{A}$, and a set of observations $\mathcal{O}$ the system can perceive. The POMDP model for dialogs operates as follows: At each time step the world is in some hidden state $s \in \mathcal{S}$. Since that state is not known exactly, the system maintains a belief distribution over all states where $b = Bel(s)$ is the probability that the true state is s. Based on this belief the machine selects an action $a \in \mathcal{A}$ and applies it, leading to a transition of the current state s to a successor state s' according to the system dynamics $P(s'|s, a)$. The user's reaction $o \in \mathcal{O}$ is observed thereafter and the entire belief distribution is updated, which can be done efficiently using Bayesian Networks, as described below.

Action set: In a dialog system, the dialog manager is the component that takes care of higher-level decision making. Its output is an action class, triggering lower-level situation-dependent actions such as context-dependent speech outputs. The details are omitted here since they are not important for an understanding of the method.

Our experimental system has 9 actions available to control the course of a museum tour. However, many actions can be summarized into summary acts, such that the system dynamics have less parameters but the wizard still has the full action set available to respond differently in various situations, as in Fig. 2(b). The action set of our experimental system is listed in Table 1.

Table 1. Action set of the dialog manager in the experiments.

Dialog act	Action class	Summary act	Description
`present`	1.1	1	Read a short text about the current exhibit along the museum tour.
`resume_present`	1.2	1	Get back to the topic and present the next text.
`offer_more`	2	2	Ask the audience whether they want to hear more information.
`offer_questions`	3	3	Ask the audience whether they have particular questions about the exhibit.
`respond_on-topic`	4.1	4	Give a statement in response to an on-topic question.
`respond_off-topic`	4.2	4	Give a statement in response to an off-topic question.
`refuse_answer`	4.3	4	Say that this question won't be answered.
`ask_repeat`	5	5	Ask the user to rephrase the earlier statement or question.
`end`	6	6	Finish explaining the current exhibit and move on to the next one.

Observation set: As for the inputs of the dialog system, the robot has capabilities to recognize speech and has a perception when and how much the people in front of it are paying attention to it. Again, to the dialog manager it does not matter what exactly the user said, the speech input is mapped to an observation class in $\mathcal{O}$ to decide the further dialog strategy. The observation set is listed in Table 2.

Dialog state tracking: The two main challenges in dialog state modeling are I) a very large state space and II) uncertainty induced by sensor noise such as speech recognizers. We attempt to solve I) by dividing the state into several variables and making reasonable independence assumptions and II) we model the whole state as a Bayesian Network, as suggested by Thomson [12], maintaining a probability distribution over state variables. This also allows one to tractably update the belief even for a very large state space. We adopt the state factorization of Williams [5],

Table 2. Action set of the user in the experiments.

Dialog act	Description
`affirm`	User accepts a suggestion.
`negate`	User refuses to hear more, or says that he has no questions.
`request_more`	User wants more information about the exhibit.
`request_stop`	User wants no more information about the exhibit.
`ask_on-topic`	User asks a question about the current exhibit.
`ask_off-topic`	User asks a small talk question, not related to the exhibit.
`silence`	User says nothing.

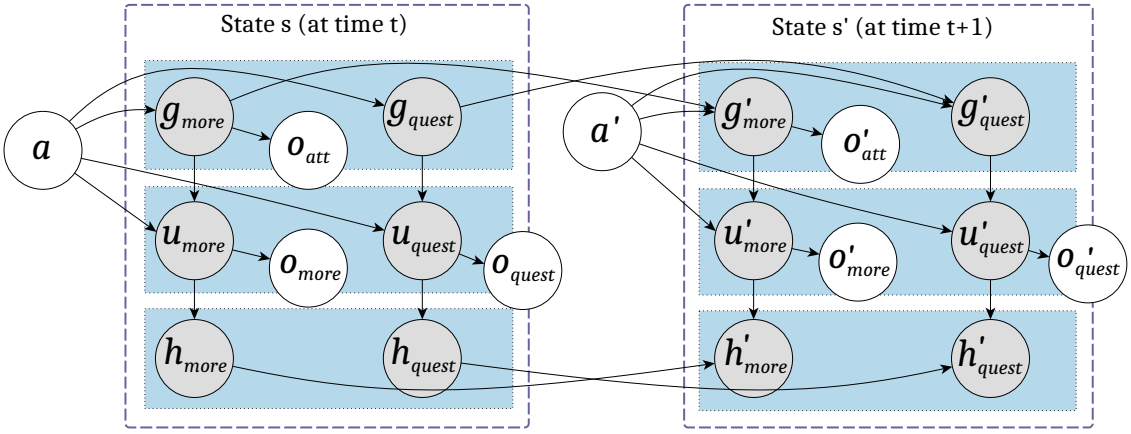


Fig. 3. Two time slices of the Bayesian Network used to track the belief state of the dialog system. Unobserved nodes are shaded, observed nodes are unshaded. This is the model of the experimental dialog system which tracks two different user goals: Hearing more information and asking questions. The components are the system action $a \in \mathcal{A}$, the state $s \in \mathcal{S} = (g, u, h)$ with $g = (g_{more}, g_{quest})$, $u = (u_{more}, u_{quest})$, and $h = (h_{more}, h_{quest})$, and the observations $o = (o_{att}, o_{more}, o_{quest})$.

who break down the dialog state into three components $s = (g, u, h)$ where g is the user goal, u is the user action, and h is a history value of previous user actions. The belief state is the joint distribution over these components: $Bel(s) = P(s) = P(g, u, h)$.

Nodes in the example system: In our experimental system we further divide the goal state into two binary goals $g = (g_{more}, g_{quest})$ which indicate the user's willingness to hear more information and ask a question, respectively. The user action variables are also factorized into $u = (u_{more,}u_{quest})$, where u_{more} has three possible values (`nothingSaid`, `more`, `noMore`) and u_{quest} has four possible values (`nothingSaid`, `noQuestion`, `onTopic`, `offTopic`). Similarly, the history variables h are factorized into $h = (h_{more}, h_{quest})$ and save the last value of the user actions.

Lastly, there are three observation variables: $o = (o_{more}, o_{quest}, o_{att})$. While the former two correspond to user action variables u and are set e.g. to a corresponding speech recognizer result class, the latter binary variable o_{att} is a visual cue of the user's attention. It is set to o_{att}=`True` if the people interacting with the robot are notably interested (not moving, face towards robot) and to o_{att}=`False` when they do otherwise, meaning the audience is bored. Thus, it gives a vague indication for the user interest goal g_{more}. The whole model is illustrated in Fig. 3.

Belief update: In order to update the belief state we need the following four probabilistic models. Firstly, the *goal model* $P(g'|a', g)$ indicates the change of goals over time. We assume that with every present-action of the robot, there is a slight change that the user gets bored, and equally there is a small chance that the user wants to ask a question. Secondly, the *user action model* $P(u|a, g)$ holds the probabilities of a user action given a goal and system action. These two models are initialized with hand-crafted parameters but later learned from real dialog data.

Conversely, the *history model* $P(h'|u', h)$ and *observation model* $P(o|u)$ are not learned from data and are therefore clamped to prevent changes during parameter learning. The history model is entirely deterministic and updates history values when new user actions are observed. Uncertain inputs are handled in the observation model which is usually a property of the speech recognizer.

With this Bayesian Network the entire belief can be updated tractably using standard algorithms. We use the junction tree algorithm and insert the values for a' (the action taken) and o'(observations made), as well as the last known distribution over g and h as evidence into the network in every time step. Hence, it is easy to compute the marginal distributions over the posterior system state $P(s') = P(g', u', h')$ which is the new belief state.

Model learning: Given a set of E dialogs, we take the observed variables $\left\{\{a_t^{(i)}, o_{att,t}^{(i)}, o_{more,t}^{(i)}, o_{quest,t}^{(i)}\}_{t=1}^{T}\right\}_{i=1}^{E}$ for every dialog sequence of length T. Inserting these values as evidence into the network while holding the history model and observation model fixed, we can now compute a maximum likelihood estimate over the hidden parameters $P(g'|a', g)$ and $P(u'|a', g')$ using the standard EM algorithm [13].

4. Lifelong user model learning and data aggregation for imitation learning

The simplest way to learn a policy imitating a teacher is to apply supervised learning, mapping beliefs to actions. The belief state of the system already provides a good set of features, but additionally we take the observations $P(o')$, the previous system action a, and a priori marginal distributions of the goals $P(g)$ and history $P(h)$ to describe the dialog situation. The whole feature vector ϕ is constructed as the concatenation of histograms:

$$\phi(Bel(s')) = [P(a), P(g), P(h), P(g'), P(u'), P(h'), P(o')]$$

where $P(\nu)$ is a column vector of the marginal probabilities of a variable $\nu \in \{a, g, h, g', u', h', o'\}$. Hence, our teacher dataset $\mathcal{D}_T$ consists of features of all dialog turns of all recorded dialogs and the corresponding teacher action. We can now train a policy $\pi : Bel(\mathcal{S}) \rightarrow \mathcal{A}$ using any supervised learning algorithm.

Since the true system dynamics may differ from the hand-crafted ones and user behavior can change over time, it is a good idea to update model parameters frequently. Fortunately, it is unlikely to visit entirely unseen states when always following the same policy, but it might occur since the dynamics model changes during runtime. In order to ensure the teacher dataset has good generalization properties, we propose the following procedure to collect training examples:

1. Initialize the system dynamics θ with hand-crafted parameters. Initialize teacher dataset $\mathcal{D}_T \leftarrow \{\emptyset\}$. Set $i = 1$.
2. Perform Wizard-of-Oz experiments with real users, record all state-action trajectories in a dataset $\mathcal{D}_T^{(i)}$. During the experiments, it is important to make decisions solely based on the belief state and act consistently.
3. If only few dialogs were recorded, generate more dialogs by sampling the model, reinforcing the teacher policy. Repeat until all desired states are visited sufficiently.
4. Add sampled trajectories to teacher dataset $\mathcal{D}_T \leftarrow \mathcal{D}_T \cup \mathcal{D}_T^{(i)}$.
5. Learn model parameters θ_i from $\mathcal{D}_T$ using the EM-algorithm.
6. Train classifier on entire dataset $\mathcal{D}_T$.
7. If the classifier result looks good, stop. Otherwise, set $i = i + 1$ and return to step 3.
8. Call the trained classifier π^* and the last set of learned parameters θ_0.

The teacher policy π^* provides a baseline, but there are no guarantees it performs well in the real world. As the policy is executed, previously unseen states might be repeatedly encountered that are insufficiently modeled.

A way to solve this is to collect more data automatically by executing π^*, as proposed by Ross [14]. Their data aggregation algorithm works as follows. In the first iteration, the teacher policy is executed for E episodes and a new dataset $\mathcal{D}$ is collected. Subsequently, a new policy $\hat{\pi}_2$ is trained that best mimics the expert on the new dataset. Iteratively, it executes the last policy $\hat{\pi}_i$ to collect more data, adds them to the dataset $\mathcal{D}$ and trains a new policy $\hat{\pi}_{i+1}$ in each iteration that best mimics the expert on the complete dataset. In other words, it accumulates a set of states that the learned policy is likely to encounter during its execution based on previous experience, and can therefore make decisions with higher certainty. Since the teacher policy labels all new data points, it is ensured that new policies can not "drift away" from the teacher.

Our lifelong learning approach (listed in Algorithm 1) is based on the work of Ross since it has strong performance guarantees and showed to produce high-accuracy policies even if the teacher classifier performs poorly. Our extension concerns the (possibly changing) underlying dynamics model. Therefore, we introduce a finite horizon H as the maximum number of dialogs in the dataset and discard the first entries from it if H is exceeded. Simultaneously, the

system dynamics are learned from the last H dialogs. Setting the horizon to large values will result in a less noisy dynamics model, but will adapt slower to changing user behavior.

Algorithm 1 Lifelong imitation learning

Initialize dataset $\mathcal{D} \leftarrow \{\emptyset\}$.
Initialize system dynamics model parameters $\theta_1 \leftarrow \theta_0$ to the parameters learned during teacher demonstrations.
Initialize $\hat{\pi}_1 = \pi^*$.
Repeat Forever
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$, such that with probability β_i it follows $\pi^*$and with probability $1 - \beta_i$ it follows $\hat{\pi}_i$.
  Interact with real users for E dialogs using π_i and system dynamics model parametrized by θ_i.
  Get dataset $\mathcal{D}_i = \{(b, \pi^*(b))\}$ of visited belief states by π_i and actions given by teacher.
  Aggregate datasets $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$.
  If $\|\mathcal{D}\| > H$
    Remove first elements from $\mathcal{D}$, such that its size is H at most.
  End If
  Learn model parameters θ_{i+1} from all trajectories in $\mathcal{D}$ using the EM-algorithm.
  Train effective classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
End Repeat

Setting the parameter $\beta_i > 0$ queries the teacher policy with probability β_i and can be seen as an "directed exploration rate", allowing to detect changes in the user behavior more quickly, other than by "forgetting" old data. It is useful in order to leverage the teacher's presence in the first few iterations while the dataset is being built up and can be decayed to zero as soon as the dataset contains H dialogs.

Equally important, the number of dialogs recorded between model updates E controls how quickly new dialogs are integrated into the model. Low values of E may result in a poor dynamics model in the first iterations, but a faster recognition of user behavior change, while high values will make model learning slower and less noisy.

5. Experimental results

The following evaluation was not conducted with real users but in a simulation to show the utility of the approach. Recall that in the real system, experiments are costly and dialogs may differ from the simulated ones, which is why they are not useful to train a policy for a reinforcement learning approach. However, they suffice for this evaluation.

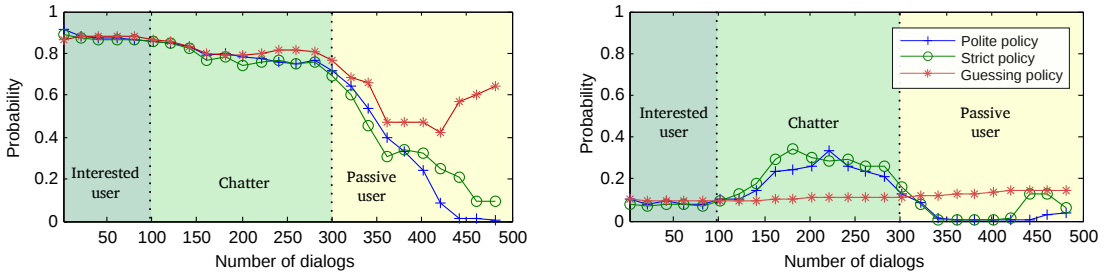
Before each dialog, our rule-based user simulator is initialized with a certain behavior in form of 15 probability distributions (e.g. the probability of responding to a question). Additionally, it is sampled how many information the user likes to hear about an exhibit and how the behavior changes when this is fulfilled. Also, every user exhibits a preference for the number and kinds of questions they may ask (on-topic vs. off-topic).

We simulate three different kinds of users commonly encountered in the museum scenario: I) An *interested* user who likes to listen to a lot of information and -when offered- will ask only on-topic questions, II) a *passive* user who just listens to very few information and will ask no questions at all, and III) a *chatter* who wants to hear some but not too much about exhibits and will ask a lot of off-topic questions.

Three different teacher policies were demonstrated for 21 training dialogs for each policy, by interacting with each user type for 7 dialogs. These teacher policies are:

1. Polite: Present information, then offer more when not sure about user interest. Answer all user questions.
2. Strict: As above, but refuse to answer off-topic questions and questions asked without permission.
3. Guessing: Do not ask whether a user is interested, but end the presentation when the probability of the user being interested gets too low.

As stated in Section 4, any supervised learning algorithm can be used to train the policies. In the experiments a linear SVM was used because it handles underrepresented classes well and is not very prone to overfitting.



(a) Probability of user being interested in the next turn $P(g'_{more} = \texttt{True} | g_{more} = \texttt{True}, a = \texttt{present})$

(b) Probability user wants to ask a question in the next turn $P(g'_{quest} = \texttt{True} | g_{quest} = \texttt{False}, a = \texttt{present})$

Fig. 4. Adaptation of model parameters during the lifetime of the dialog controller. The shown probabilities were learned from dialogs using the three policies: polite, strict, and guessing, keeping the last 80 dialogs in memory. Model parameters were estimated as expected: The *interested* user and the *chatter* have a high interest while the interest drops when the *passive* user sets in. The *chatter* is more likely to ask questions than the *interested* user, while the *passive* user's question asking likelihood is nearly zero. Depending on the applied policy, model parameters vary because it matters when and how often the user status is inquired by asking, as can be seen on the "guessing" policy.

For all tests we set $\beta_1 = 1$, $H = 80$, and $E = 20$ and let the algorithm interact with the simulator. At several points the user behavior abruptly changes (from interested to chatter to passive) in order to see how the new user behavior is learned and the policy reacts. Some of the model parameters learned are visualized in Fig. 4.

All three tested teacher policies were mimicked as expected over a total lifetime of 500 dialogs. Since user behavior was changed abruptly, for a short time the model did not reflect the true user behavior and therefore the dialog manager made mistakes until all data from the previous user were discarded from the history. However, we do not expect this to happen in the real world. We also tested the teacher policies on a mixture of all three user types, which is more realistic. This resulted in fairly stable user model parameters over the whole lifetime.

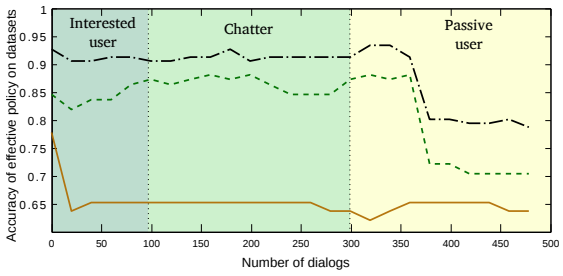
The trained classifiers mimicking the wizard showed very good performance throughout the dialog controller lifetime and for all the tested policies, even with a suboptimal teacher policy. The classifier performance for the "polite" policy is shown exemplarily in Table 3.

Table 3. Performance of the effective policies at dialog 1 (teacher dataset), and after learning the user models for the interested user (dialog 100), chatting user (dialog 300), and passive user (dialog 500), using the "polite" teacher dataset and lifelong model learning.

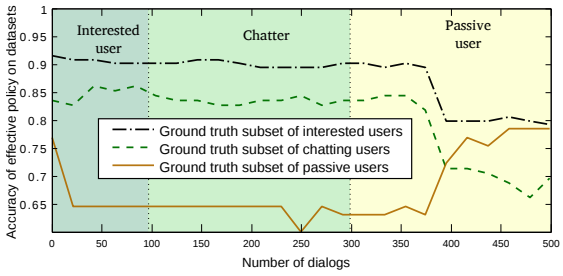
	Teacher dataset		Dataset at dialog 100		Dataset at dialog 300		Dataset at dialog 500	
Dialog act class	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall
`present`	0.95	0.99	0.99	0.99	0.99	0.99	1.0	0.99
`resume_present`	0.9	0.82	1.0	0.94	0.88	0.92	0.97	1.0
`offer_more`	0.77	0.77	0.96	0.93	0.97	0.92	0.98	0.99
`offer_questions`	0.62	0.87	1.0	0.99	0.97	0.94	1.0	0.91
`respond_on-topic`	1.0	1.0	0.98	1.0	1.0	1.0	-	-
`respond_off-topic`	1.0	1.0	0.89	1.0	0.98	1.0	-	-
`ask_repeat`	1.0	0.75	1.0	1.0	1.0	1.0	1.0	1.0
`end`	0.78	0.86	0.95	0.97	0.96	0.98	0.96	1.0

This result is not very surprising since the teacher policy will always choose the label for a new belief point according to its own decision boundaries and naturally group similar belief points in the same class. We therefore evaluate how well the ground truth (teacher) dataset is represented by the learned effective policies over the lifetime. We compare a dialog system with a hand-crafted model to a version with learned parameters. Since the ground truth dataset consists of dialogs with all three user types, we split it into these three subsets of 7 dialogs each, in order to see the adaptation to different user types. Results are shown in Fig. 5 and suggest that learning model parameters during runtime in fact improves adaptation to the current users.

Two things can be seen in Fig. 5: Firstly, a change of user behavior during runtime affects how well the ground truth datasets are classified by the effective policies since they are based on the current user behavior. In the static dialog manager, the passive user subset does not fit well into the dataset generated by the interested and chatting user



(a) Accuracy of the effective policies on the ground truth subsets (interested, chatter, passive) using hand-crafted model parameters.



(b) Accuracy of the effective policies on the ground truth subsets (interested, chatter, passive) using learned model parameters.

Fig. 5. Accuracy of the effective (continuously updated) policies during runtime on the three ground truth subsets of the "polite" teacher dataset. If the dialog controller learned to adapt to different users, the subset recorded with one user during teaching would attain the highest accuracy while interacting with the same respective user at runtime. Since the behavior of *interested* user and *chatter* are very similar, they respective datasets attain a similar accuracy throughout the lifetime and thus this does not indicate a loss of generality.

and vice versa, therefore the accuracy drops when the user type changes. Secondly, we can see how well the dialog controller adapts to different users by these accuracy values. While the passive user dataset attains rather steady accuracy (0.65) with the static dialog manager over the whole lifetime, as shown in Fig. 5(a), the dynamic dialog manager adapts better. This can be seen in Fig. 5(b) as the behavior changes from *chatter* to *passive*: The passive user subset reaches similar or better performance on the effective policies than the other subsets, indicating that the dialog manager learned to use more information from the passive user subset when interacting with passive users.

6. Conclusion and future work

The evaluation suggests that lifelong model updates improve the ability to mimic a teacher policy when user behavior changes over time. However, the executed policies strongly depend on teacher demonstrations, depending on the complexity of the task sufficient teacher demonstrations have to be recorded to cover all situations one wants to consider. We are aware that the provided results are merely a proof of concept, and for a serious evaluation real user studies have to be conducted. This will be provided in a future article and evaluated in more detail to demonstrate how well the approach works in real the real world.

Acknowledgements. This work was funded by ESF grant number 100130198.

References

1. Young, S., Gasic, M., Thomson, B., Williams, J.D.. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE* 2013;**101**:1160–1179.
2. Bui, T.H., Poel, M., Nijholt, A., Zwiers, J.. A tractable hybrid ddn–pomdp approach to affective dialogue modeling for probabilistic frame-based dialogue systems. *Natural Language Engineering* 2009;**15**:273–307.
3. Georgila, K., Traum, D.R.. Reinforcement learning of argumentation dialogue policies in negotiation. In: *INTERSPEECH*. 2011, .
4. Foster, M.E., Keizer, S., Lemon, O.. Towards action selection under uncertainty for a socially aware robot bartender. In: *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. 2014, .
5. Williams, J., Young, S.. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language* 2007;.
6. Schatzmann, J., Weilhammer, K., Stuttle, M., Young, S.. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The Knowledge Engineering Review* 2006;**21**:97–126.
7. Poschmann, P., Donner, M., Bahrmann, F., Rudolph, M., Fonfara, J., Hellbach, S., et al. Wizard of oz revisited: Researching on a tour guide robot while being faced with the public. In: *RO-MAN, 2012 IEEE*. 2012, .
8. Atkeson, C.G., Schaal, S.. Robot learning from demonstration. In: *ICML*. 1997, .
9. Englert, P., Paraschos, A., Peters, J., Deisenroth, M.. Probabilistic model-based imitation learning. *Adaptive Behavior Journal* 2013;.
10. Pomerleau, D.A.. Alvinn: An autonomous land vehicle in a neural network. Tech. Rep.; DTIC Document; 1989.
11. Abbeel, P., Ng, A.Y.. Apprenticeship learning via inverse reinforcement learning. In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, .
12. Thomson, B., Young, S.. Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems. *Computer Speech & Language* 2010;**24**:562–588.
13. Bishop, C.M.. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc.; 2006. ISBN 0387310738.
14. Ross, S., Gordon, G.J., Bagnell, J.A.. A reduction of imitation learning and structured prediction to no-regret online learning. *arXiv preprint arXiv:10110686* 2010;.