Learning the Reward Model of Dialogue POMDPs from Data

Abdeslam Boularias, Hamid R. Chinaei, Brahim Chaib-draa
Department of Computer Science
Laval University
Quebec, QC, Canada
boularias,hrchinaei,chaib@damas.ift.ulaval.ca

Abstract

Spoken language communication between human and machines has become a challenge in research and technology. In particular, enabling the health care robots with spoken language interface is of great attention. Recently due to uncertainty characterizing dialogues, there has been interest for modelling the dialogue manager of spoken dialogue systems using Partially Observable Markov Decision Processes (POMDPs). In this context, we would like to learn the reward model of dialogue POMDPs from expert's data.

In a previous paper, we used an unsupervised learning method for learning the states, as well as the transition and observation functions of the dialogue POMDPs based on human-human dialogues. Continuing our objective of learning the components of dialogue POMDPs from data, here we introduce a novel inverse reinforcement learning (IRL) algorithm for learning the reward function of the dialogue POMDP model. Based on the introduced method, and from an available corpus of data, we construct a dialogue POMDP. Then, the learned dialogue policies, based on the learned POMDP, are evaluated. Our empirical evaluation shows that the performance of the learned POMDP is higher than expert performance in non, low, and medium noise levels, but the high noise level. At the end, current limitations and future directions are addressed.

1 Introduction

Currently, we are working on the project *robotic assistant for persons with disabilities*[1] with the goal of equipping *Smartwheeler* [10] with a robust language interface. Smartwheeler is a wheelchair available at Mcgill university which is used for research on increasing the autonomy and safety of individuals with severe mobility impairment[2]. In this context, we are interested in modelling the Dialogue Manager of Smartwheeler using methods of sequential decision making under uncertainty.

Recently, there has been a great interest for modelling the dialogue manager of spoken dialogue systems using Partially Observable Markov Decision Processes (POMDPs) [14]. In fact, POMDPs can be used to model uncertainty available in spoken language which is induced by Automatic Speech Recognition (ASR), and Natural Language Understanding (NLU). However, in POMDPs estimating the environment's dynamics is a significant issue; as it has direct impact on their applicability in the domain of interest. This is because of the fact that POMDP model can highly impact planned policies. For instance, [7] suggests learning the POMDP model from data for their proposed POMDP for assisting persons with dementia during hand washing.

[1] http://damas.ift.ulaval.ca/projet.php#i
[2] http://www.cs.mcgill.ca/~smartwheeler/

In a previous work [1], we learned a set of states, as well as transition and observation functions for a dialogue POMDP from expert dialogues [12]. By expert dialogues we mean some human-human dialogues generated through a (simulated) voice channel, where human at one side has the role of a dialogue manager (expert), and the one at the other side has the role of a user who is trying to interact with the machine to achieve her goal similar to SACTI dialogues [13] [12]. In this paper, first we describe our previous work for learning the states, transition and observation functions, for a dialogue POMDP using unannotated data available in [13].

Moreover, Inverse Reinforcement Learning (IRL) methods are used for learning the reward function of (PO)MDP models using expert dialogues. In fact, IRL methods suits us as our available data have been generated based on human human conversion in a wizard-of-oz setting. In this paper, we introduce a novel IRL algorithm for POMDPs to learn the reward function which the human expert, in the role of a dialogue manager, is maximizing.

In the rest of the paper, we briefly present POMDP background in Section 2, and describe our past work for learning states, as well as transition and observation functions in Section 3. Section 4 introduces IRL and our algorithm for learning the reward function in POMDPs followed by the experiments in Section 5. Finally, the conclusion and future directions are presented in Section 6.

2 Background

A POMDP can be represented by n-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{O}, T, R, \Omega, \gamma, H\}$ where $\mathcal{S}$ is the set of some discrete states, $\mathcal{A}$ is the set of some discrete actions, $R(s, a)$ is the reward of taking action a in the state s, and T the transition model which consists of the probability of state transitions:

$$T(s, a, s') = P(s_{t+1} = s' | a_t = a, s_t = s)$$

where s is for the current state of the system and s' for the next state of the system.

Moreover, $\mathcal{O}$ is the set of some discrete observations which is used for estimating the current state using the model Ω, with $\Omega(o, s, a) = P(o|a, s)$, i.e. the probability of observing o after taking the action a which results in the state s. The system's initial belief state is b_0, and the belief state at time t is derived from $b_t = P(s_t | b_0, a_0, o_1, \ldots, b_{n-1}, a_{n-1}, o_n)$, γ is a discount factor, and H is the planning horizon. The belief at time $t + 1$, b_{t+1}, can be computed from the previous belief, b_t, the last action a, and observation o, by applying Bayes rule:

$$b_{t+1}^{a,o}(s') = \frac{1}{P(o|b_t, a)} \Omega(o, s', a) \sum_{s \in \mathcal{S}} T(s, a, s') b_t(s)$$

where $P(o|b_t, a) = \sum_{s' \in \mathcal{S}} \Omega(s', a, o) \sum_{s \in \mathcal{S}} T(s, a, s') b_t(s)$ is the probability of observing o after doing action a in belief b_t, this acts as a normalizing constant such that b_{t+1} remains a probability distribution:

In POMDPs, the system's goal is to find an optimal policy, π, where $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$, that maximizes the expected discounted rewards. That is, the policy which has the maximum value. The value of policy π is defined as:

$$V^{\pi}(s) = \sum_{t=0}^{\infty} E_{s_t, a_t}[\gamma^t R(s_t, a_t) | s_0 = s, \pi]$$

If the environment dynamics are not known, i.e. transition probabilities, observation probabilities, or the reward function are not known in advance, the system has to assume the environment as an unknown POMDP and during interaction with the user tries to approximate the unknown model(s). Conversely, if the dynamics are known to the system then the classic dynamic programming techniques, such as point based value iteration [11], can be used to find an optimal policy π^*. In this work, we would like to learn the dialogue POMDP from data, to be able to learn the POMDP policy from available model-based algorithms.

3 Learning dialogue POMDP states, and transition and observation functions

In this section, we briefly describe our previous work for learning states, as well as transition and observation functions from dialogues [1]. First, we briefly explain Hidden Topic Markov Model (HTMM) [6] that we use to learn hidden intentions behind user's utterances.

HTMM is a method which combines Hidden Markov Model (HMM) and Latent Dirichlet Analysis (LDA) for obtaining some topics for documents [6]. HMM is a framework for obtaining the hidden states based on some observations in Markovian domains such as part-of-speech tagging [3]. In LDA, similar to Probabilistic Latent Semantic Analysis (PLSA), the observations are explained by groups of latent variables. For instance, if we consider observations as uttered words in a dialogue, then the dialogue is considered as a bag of words with mixture of some intentions, where intentions are represented by the words with higher probabilities. In LDA as opposed to PLSA, the mixture of intentions is generated from a Dirichlet prior mutual to all dialogues in a dialogue set. Since HTMM adds the Markovian property inherited in HMM to LDA, in HTMM the dependency between successive words is regarded, and the dialogue is no longer seen as a bag of words. Notice that HTMM can be considered as a clustering method such as LSA, however, it is probabilistic similar to PLSA, and moreover, the Markovian property between the utterances are considered as opposed to PLSA or LSA.

Hidden intentions can be used as user's hidden states in a dialogue POMDP [4]. For each dialogue in the corpus, we assign its hidden state as the maximum likely state, then we estimate the transition function using maximum likelihood with a smoothing method to make it more robust. To construct an observation function for the dialogue POMDP, we use the learned hidden states for each utterance as its meta observation. This meta observations are used as the POMDP observation set rather than the whole words in the utterance to be able to reduce the size of observation set significantly. However, this model is a fully observable Markov Decision Process (MDP). To do it in POMDPs, the observations are reduced to the number of states, and the observation function is estimated by taking average over belief of states given each action and state. This allows that each meta observation be allowed in other states with a small probability.

In HTMM, the special form of the transition matrix enables a reduced time complexity of $O(|d|N^2)$, where $|d|$ is the length of the dialogue d, and N is the number of desired user's intentions, given to the algorithm [5]. The small time complexity of the algorithm enables the dialogue system to apply it at any time to update the observation function based on its recent observations.

4 Learning dialogue POMDP Reward Function

The key insight behind Inverse Reinforcement Learning (IRL) is that the reward function is the most succinct hypothesis for explaining a behaviour. Thus, an observed behaviour can be better generalized to new contexts when a reward function is known. IRL refers to the problem of finding a reward function that explains the human's demonstrated actions. In this section, we first present the background on IRL in MDPs, then we describe an efficient new algorithm for IRL in Partially Observable MDPs (POMDP).

4.1 Inverse Reinforcement Learning in MDPs

We assume the existence of a feature vector of size k, $\phi_i \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$, such that the reward function R is a linear combination of these features with *unknown* weights $w_i \in \mathbb{R}$:

$$\forall (s,a) \in \mathcal{S} \times A : R(s,a) = \sum_{i=1}^{k} w_i \phi_i(s,a)$$

The expected frequency of a feature i when the agent starts executing policy π in state s is defined as:

$$V_i^{\pi}(s) = \sum_{t=0}^{\infty} E_{s_t,a_t}[\gamma^t \phi_i(s_t,a_t)|s_0 = s, \pi]$$

Input: A POMDP model $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, \Omega, b_0, \gamma, H)$ without a reward function, and a set $\mathcal{D}$ of trajectories $a_1^m o_1^m \dots a_{t-1}^m o_{t-1}^m a_t^m, t \leq H$ demonstrated by a human;
Output: Reward weights $w_i \in \mathbb{R}$;

Extract the human's stochastic policy π from the demonstration;
Initialize the set of variables $\mathcal{V}$ with the weights w_i;
Initialize the set of linear constraints $\mathcal{C}$ with $\{\forall (s,a) \in \mathcal{S} \times \mathcal{A} : R_{min} \leq \sum_{i=1}^{k} w_i \phi_i(s,a) \leq R_{max}\}$;
for $t = H$ **to** *1* **do**
  foreach $h \in \mathcal{D}$, *such that* h *is a trajectory of length* t, **do**
    Calculate b_h, the belief state at the end of trajectory h;
    foreach $(a, o) \in \mathcal{A} \times \mathcal{O}$ **do**
      Add the variable V_{hao} to $\mathcal{V}$ // is an approximation of the value of π at b_{hao} defined below;
      if $h_{ao} \notin \mathcal{D}$ *and* $t < H$ **then**
        Let b_{hao} be the belief corresponding to the trajectory hao;
        Calculate the belief states b_{h_i} corresponding to the trajectories in $\mathcal{D}$ of length $t+1$;
        Find a list of weights α_i such that $b_{hao} = \sum_{i=0}^{n} \alpha_i b_{h_i}$;
        Add to $\mathcal{C}$ the constraint $V_{hao} = \sum_{i=0}^{n} \alpha_i V_{h_i}$;
        // V_{hao} is an approximation of the value of π at the belief corresponding to the trajectory hao
      end
      if $h_{ao} \notin \mathcal{D}$ *and* $t = H$ **then**
        Add the constraint $V_{hao} = 0$ to the set $\mathcal{C}$;
      end
    end
    Add the variable V_h to $\mathcal{V}$ // the value of π at b_h;
    Add to $\mathcal{C}$ the constraint $V_h = \sum_{a \in \mathcal{A}} \pi(h,a)[\sum_{s \in \mathcal{S}} b_h(s,a) \sum_{i=1}^{k} w_i \phi_i(s,a) + \gamma \sum_{o \in \mathcal{O}} Pr(o|h,a) V_{hao}]$;
    foreach $a \in \mathcal{A}$ **do**
      Add the variable V_h^a to $\mathcal{V}$ // the value of an alternative policy choosing a after the trajectory h ;
      Add to $\mathcal{C}$ the constraint $V_h^a = \sum_{s \in \mathcal{S}} b_h(s,a) \sum_{i=1}^{k} w_i \phi_i(s,a) + \gamma \sum_{o \in \mathcal{O}} Pr(o|h,a) V_{hao}$;
      Add the variable ϵ_h^a to the set $\mathcal{V}$;
      Add to $\mathcal{C}$ the constraint $V_h - V_h^a \geq \epsilon_h^a$;
    end
  end
end
Maximize $\sum_{h \in \mathcal{H}} \sum_{a \in \mathcal{A}} \epsilon_h^a$ subject to the constraints of set $\mathcal{C}$;

Algorithm 1: Point-based Inverse Reinforcement Learning for POMDPs.

Using this definition, the value of a policy π can be written as: $V^\pi(s) = \sum_{i=1}^{k} w_i V_i^\pi(s)$. Given a set $\mathcal{D}$ of M trajectories

$$s_1^m a_1^m \dots s_{t_m}^m a_{t_m}^m,$$

$m \leq M, t_m \leq H$, sampled by executing a policy π, the expected frequency of a feature i can be approximated by using the following Monte Carlo estimation:

$$\hat{V}_i^\pi = \frac{1}{M} \sum_{m=1}^{M} \sum_{t=1}^{t_m} \gamma^t \phi_i(s_t^m, a_t^m) \tag{1}$$

Assuming that the rewards are in the interval $[R_{min}, R_{max}]$, a reward function $\mathcal{R}$ can be found by solving the following linear program:

$$\max_{\{w_i\}} \min_{\pi' \neq \pi} \sum_{s \in \mathcal{S}} \sum_{i=1}^{k} w_i [\hat{V}_i^\pi(s) - V_i^{\pi'}(s)]$$

$$\text{s.t. } \forall (s,a) \in \mathcal{S} \times A : R_{min} \leq \sum_{i=1}^{k} w_i \phi_i(s,a) \leq R_{max}$$

Instead of comparing the value of the demonstrated policy π with that of each possible alternative policy π', [9] proposed to start with a randomly chosen policy π^0, and iterate between finding a reward function R^k by solving the linear program above with $\pi' = \pi^k$, and finding an optimal policy π^{k+1} given the reward R^k.

4.2 Inverse Reinforcement Learning in POMDPs

A POMDP can be seen as an MDP where the states correspond to belief states. Building upon this observation, [2] showed that the IRL algorithm introduced by [9] can be extended to POMDPs without making major changes. However, the number of possible belief states grows double exponentially with respect to the planning horizon and the number of observations. Moreover, the human's optimal action is unknown for most of these belief states. Therefore, the error introduced by the Monte Carlo estimator (Equation 1) can be significantly large when used for a POMDP.

In this section, we propose a robust point-based IRL algorithm for POMDPs. Our approach is described in Algorithm 1. The main idea of this algorithm is to gradually construct a linear program by adding variables corresponding to the values V_h of the human's stochastic policy π after each trajectory h, as well as the values of alternative policies V_h^a selecting a deterministic action at h, and following π for the remaining time-steps. These variables are subject to linear constraints corresponding to Bellman equation (steps 21 and 24). Whenever a trajectory hoa does not appear in the demonstration, the value of a policy after hoa, for $(o, a) \in \mathcal{O} \times A$, is approximated by a linear combination of the values of the human's policy at other trajectories h_i that appeared in the demonstration and have the same length as hoa (Step 13). Notice that, due to the piecewise linearity of the optimal value function, this latter approximation corresponds to the true value if the human's policy in the belief state b_{hoa} is the same as in the belief states b_{h_i} used in the linear combination. This condition is more likely to be verified when the beliefs b_{h_i} are closer to b_{hao}. Finally, we explicitly state that the value of the human's policy at any history h is higher than the value of any alternative policy by a margin ϵ_h^a (Step 26) that should be maximized (Step 30). An alternative policy is defined as a deterministic policy that selects an action a and then follows the human's policy for the upcoming time-steps (Step 24 in the algorithm).

This algorithm requires solving one linear program with $\mathcal{O}(|\mathcal{A}||\mathcal{O}|MH)$ variables and $\mathcal{O}(|\mathcal{A}|MH)$ constraints. Furthermore, the computational complexity of each linear approximation is $\mathcal{O}(M^3H^3)$, given that there are $\mathcal{O}(MH)$ belief states where the human's policy is known. Therefore, the overall complexity of the algorithm is polynomial in the parameters $|\mathcal{A}|$, $|\mathcal{O}|$, M and H.

5 Experiments

5.1 Learning the dialogue POMDP for SACTI1

This section describes the method that we use for learning POMDP states, as well as the transition and observation functions for SACTI1 dialogues [13]. Using HTMM [6], we designed a dialogue POMDP for SACTI1 dialogues which is publicly available at: `http://mi.eng.cam.ac.uk/projects/sacti/corpora/`. There are about 144 dialogues between 36 users and 12 experts who have the role of a dialogue manager for 24 total tasks. The utterances from users to human experts are first confused using a speech recognition error simulator [8].

Figure 1 shows the 3 captured user's intentions and their top 20 words with their probabilities learned by HTMM. For each intention, we have highlighted the keywords which best distinguish the intention. These intentions are for the user's intentions for request information about some visiting places, the transportation, and food places, respectively. The number of intentions (here 3) is set manually. Similarly, the highlighted words for indicating the meaning of each state has been done manually.

Without loss of generality, we can consider the user's intention as the system's state [4]. Based on the above captured intentions, we defined 3 primary states for the SACTI1 DM as follows: *visits (v)* , *transports (t)* , and *foods (f)*. Moreover, we defined two absorb states, i.e., *Success (S)* and *Failure (F)* for dialogues which end successfully and unsuccessfully, respectively. The Success and Failure states are identified by user. In SACTI1 corpus, after finishing each dialogue, the user assigns the level of precision and recall [13]. These are the only explicit feedback that we require from the user, which is used to define absorb states of dialogue POMDP. A dialogue is successful if its precision and recall is above a predefined threshold. The set of actions are coming directly from SACTI1 corpus.

The transition model is calculated using maximum likelihood with add-one smoothing to make a more robust transition model:

Intention 0: visits				**Intention 1: transports**				**Intention 2: foods**			
the	0.08	like	0.01	the	0.08	a	0.02	you	0.06	um	0.02
i	0.06	**hotel**	0.01	to	0.04	does	0.02	the	0.04	and	0.02
to	0.05	for	0.01	is	0.04	**road**	0.02	i	0.04	thank	0.01
um	0.02	would	0.01	how	0.03	and	0.01	a	0.03	to	0.01
is	0.02	i'm	0.01	um	0.02	on	0.01	me	0.03	of	0.01
a	0.02	**tower**	0.01	it	0.02	long	0.01	is	0.02	**restaurant**	0.01
and	0.02	**castle**	0.01	uh	0.02	of	0.01	uh	0.02	there	0.01
you	0.02	go	0.01	i	0.02	much	0.01	can	0.02	do	0.01
uh	0.02	do	0.01	from	0.02	**bus**	0.01	tell	0.02	could	0.01
what	0.01	me	0.01	**street**	0.02	there	0.01	please	0.02	where	0.01

Figure 1: Intentions learned by HTMM for SACTI1, with their 20-top words and their probabilities.

$$T(s_1, a_1, s_2) \quad = \quad \frac{Count(s_1, a_1, s_2) + 1}{Count(s_1, a_1) + K}$$

where $K = |\mathcal{S}|^2|\mathcal{A}|$.

For the choice of observation function, we assumed 5 observations, each one is specific for one state, i.e. user's hidden intention. we use the notation *O*= { *VO, TO, FO, SuccessO, FailureO* } for the observations for *visits, transports, foods, Success*, and *Failure*, respectively. For each user's intention, one can capture POMDP observations given each utterance $W = \{w_1, \ldots, w_{|W|}\}$ by taking a maximum likely state using HTMM learned observation model. Note that during learning time, for each hidden state s HTMM learns a vector $\beta_{w,s}$ which gives the probability of observing each word w_i. Then, during human-machine interaction, given any arbitrary user's utterance the POMDP observation o is captured as:

$$o = \text{argmax}_z \prod_i \beta_{w_i s} \tag{2}$$

Then, the observation function is estimated by taking average over belief of states given each action and state. The learned transition and observation functions, together with the dialogue trajectories have been used in our IRL algorithm to learn the reward function. The dialogue trajectories are in the form of observation action, oa, for non, low, med, and high noise dialogues.

Because of the small time complexity of the algorithm discussed earlier, the algorithm took real time (less than a second) to converge on the authors machine with 1.7 MHz cpu and 4 gigabytes memory. This fact suggests use of the algorithm after finishing some tasks by the system to update its model, and hopefully a better policy.

5.2 Experimental Results

We captured the components of the dialogue POMDP for SACTI1 corpus as described above. The generated dialogue POMDP consist of 5 states, 14 actions and 5 meta observations drawn by HTMM using 817 primitive observations (words). We solved the POMDP models, using ZMDP software available online at: `http://www.cs.cmu.edu/~trey/zmdp/`. We set a uniform distribution on the 3 primary states (*visits, transports, and foods*), and set the discount factor to 90%. Based on simulation, we evaluated the quality of the learned models based on the performance of their policies learned by ZMDP.

First, we evaluated use of HTMM for learning states, transition and observation function for SACTI1 dialogues. So, we fixed the reward function similar to a hand-crafted reward function. That is, a large positive reward for successfully dialogue completion, a large negative reward for unsuccessfully dialogue completion, and -1 reward for any other dialogue state. On the other hand, we applied our IRL algorithm on the expert dialogues to learn the reward function of our POMDP. Then, we evaluated the influence of the number of training dialogues on the quality of the learned transition and observation functions. To do so, we fixed the number of states and the reward function, and updated the transition and observation functions using 24, 48, 72, and 96 dialogues (labelled as

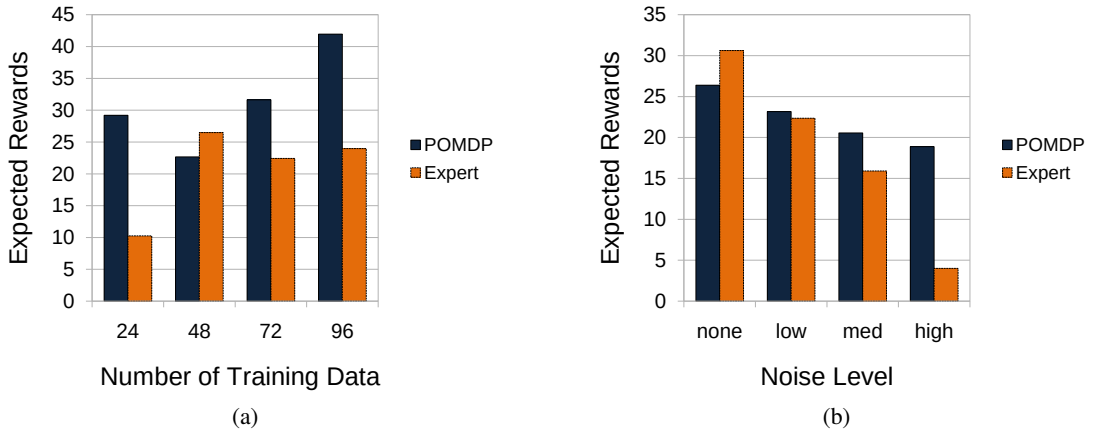

Figure 2: (a): Comparison of performance in dialogue POMDPs v.s. experts with respect to the number of expert dialogues. (b): Comparison of performance in dialogue POMDPs v.s. experts with respect to the noise level.

Number of Training Data in Figure 2(a)). Then, we solved each POMDP model using ZMDP. We measured the average gathered rewards for each learned POMDP based on simulation runs. The average rewards for each model was measured for 24, 48, 72, and 96 runs, respectively.

Figure 2 (a) shows that in overall by increasing expert dialogues, as training data, the updated POMDP models perform better. In other words, by increasing data the introduced method learns better dialogue POMDP models. This is in particular an advantage for us; as stated earlier the time complexity for learning the transition and observation functions is polynomial with respect to the length of all used dialogues. The exception in the quality of the learned model is when we used 48 dialogues where the dialogue POMDP performance decreased compared to when 24 dialogues were used. The reason could be the use of EM for learning the model which is prone to local optima. Moreover, EM depends on the priors α and η. In this work, we set the priors based on heuristic given in [6], and our trial and error experiments, which is indeed a drawback for use of parametric models in real applications.

Moreover, based on our simulations, we evaluated the behaviour of generated POMDP models to ASR noise since SACTI1 data has been produced based on a simulated ASR channel [8]. The transcribed dialogues from user to the human expert, as the dialogue manager, can contain no, low, medium, or high noise. For each noise level, we randomly took 24 expert dialogues and learned the transition and observation functions of the POMDP model. Then, for each POMDP model we learned its policy using ZMDP, and performed 24 simulations to calculate their average gathered rewards. Figure 2 (b) compares these averaged rewards to the average of gathered rewards in their corresponding expert dialogues. As the figure shows the dialogue POMDP models are more robust to ASR noise levels compared to their corresponding expert dialogues. That is, the relative performance of POMDP is much better in higher levels of noise, i.e. medium and high noise levels than non or low noise levels. This result is not surprising as in the higher noise levels, there is more uncertainty which POMDPs can show their performance better compared to lower levels of noise with no or little uncertainty.

Furthermore, we evaluated the performance of our IRL algorithm for learning the reward function of the dialogue POMDP. We used the four learned POMDPs mentioned-above as our baseline (learned using non, low, medium, and high noise dialogues, respectively, and with hand-crafted reward). On the other hand, we applied our IRL algorithm on the expert trajectories with non, low, medium, and high levels of noise to learn 4 reward functions. Using ZMDP, we learned the policy for the POMDPs with the 4 learned rewards, respectively. Then, for each dialogue turn in all the dialogues of the corpus we found the suggested actions by each of the POMDPs. Table 1 shows the number of mistakes for each dialogue POMDP with respect to the expert actions inside dialogues in the corpus. The results in the table suggest that POMDPs with learned rewards using non, low, and medium noise

levels performed much better than corresponding POMDPs with hand-crafted rewards. However, the number of mistakes increased drastically for the POMDP with learned rewards using high noise dialogues. This is because in a high level of noise, the belief states are almost uniform distributions in several time-steps. Therefore, a similar reward, of around 100, was attributed to many different actions by our IRL algorithm. This problem may be addressed, in a future work, by multiplying each error margin by an importance factor that is proportional to the entropy of the corresponding belief state.

Table 1: Number of mistakes for hand-crafted reward POMDPs, and learned reward POMDPs.

Noise level	non	low	med	high
Hand-crafted reward POMDP mistakes	1076	1088	1040	746
Learned reward POMDP mistakes	**546**	**546**	**1007**	1028

6 Conclusion

This paper presents our work on learning dialogue POMDP states, transition, observation and reward functions using data containing real human dialogues. In particular, we described our IRL algorithm for constructing a linear program with a polynomial time complexity. The POMDPs with learned rewards performed well in lower levels of noise. In fact, the rewards in lower levels of noise seem to capture the human's trade-off between enquiring information and making a shorter dialogue. In high noise levels however, the belief states have almost uniform distribution, and a similar reward of around 100, was attributed to many different actions. This problem may be solved by using more demonstrations.

In the future work, we are going to enhance our IRL algorithms for instance by using some informative features, as we did not use any feature of this kind in our current IRL algorithm. Moreover, we may incorporate domain-knowledge in form of priors on the reward function.

References

[1] Hamid R. Chinaei, Brahim Chaib-draa, and Luc Lamontagne. Learning user intentions in spoken dialogue systems. In *Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART)*, pages 107–114, Porto, Portugal, January 2009.

[2] Jaedeug Choi and Kee-Eung Kim. Inverse reinforcement learning in partially observable environments. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence (IJCAI)*, pages 1028–1033, 2009.

[3] Kenneth Ward Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of conference on Applied Natural Language Processing (ANLP)*, pages 136–143, Morristown, NJ, USA, 1988.

[4] Finale Doshi and Nicholas Roy. Spoken language interaction with model uncertainty: an adaptive human-robot interaction system. *Connection Science*, 20(4):299–318, 2008.

[5] Amit Gruber and Ashok Popat. Notes regarding computations in openhtmm, 2007.

[6] Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. Hidden topic markov models. In *Artificial Intelligence and Statistics (AISTATS)*, San Juan, Puerto Rico, 2007.

[7] Jesse Hoey, Axel von Bertoldi, Pascal Poupart, and Alex Mihailidis. Assisting persons with dementia during handwashing using a partially observable markov decision process. In *Proceedings of the International Conference on Vision Systems (ICVS), Biefeld, Germany*, pages 1028–1033, 2007.

[8] Jason D. Williams Matthew Stuttle and Steve Young. A framework for wizard-of-oz experiments with a simulated asr-channel. In *Proc Intl Conf on Speech and Language Processing (ICSLP)*, 2004.

[9] Andrew Ng and Stuart Russell. Algorithms for Inverse Reinforcement Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, pages 663–670, 2000.

[10] Joelle Pineau and Amin Atrash. Smartwheeler: A robotic wheelchair test-bed for investigating new models of human-robot interaction. In *AAAI Spring Symposium on Multidisciplinary Collaboration for Socially Assistive Robotics*, 2007.

[11] Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025 – 1032, August 2003.

[12] Karl Weilhammer, Jason D. Williams, and Steve Young. The SACTI-2 Corpus: Guide for Research Users, Cambridge University. Technical report, 2004.

[13] Jason D. Williams and Steve Young. The SACTI-1 Corpus: Guide for Research Users. Cambridge University Department of Engineering. Technical report, 2005.

[14] Jason D. Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21:393–422, 2007.