Dialogue Strategy Learning in Healthcare: A Systematic Approach for Learning Dialogue Models from Data

Hamid R. Chinaei
Hamidreza.Chinaei.1@ulaval.ca

Brahim Chaib-draa
Brahim.Chaib-Draa@ift.ulaval.ca

Abstract

We aim to build dialogue agents that optimize the dialogue strategy, specifically through learning the dialogue model components from dialogue data. In this paper, we describe our current research on automatically learning dialogue strategies in the healthcare domain. We go through our systematic approach of learning dialogue model components from data, specifically user intents and the user model, as well as the agent reward function. We demonstrate our experiments on healthcare data from which we learned the dialogue model components. We conclude by describing our current research for automatically learning dialogue features that can be used in representing dialogue states and learning the reward function.

1 Introduction

Cognitive assistive technologies provide support systems for the elderly, possibly with cognitive or physical disabilities, for instance people with dementia (such as Alzheimer's disease) (Boger et al., 2005; Pineau et al., 2011; Rudzicz et al., 2012). Such support systems can significantly reduce the costs of performing several tasks, currently done by family members or employed caregivers. In this context, (Rudzicz et al., 2012) are working on a computerized caregiver that assist individuals with Alzheimer's disease (AD) to complete daily tasks (e.g., preparing meals) using verbal communication. Thus, an important component of such technologies is the dialogue agent.

Table 1 (left) shows sample dialogues collected by SmartWheeler, an intelligent wheelchair for persons with disabilities (Pineau et al., 2011). In particular, SmartWheeler aims to minimize the physical and cognitive load required in steering it. SmartWheeler is equipped with a dialogue agent, thus the users can give their commands through the spoken language besides a joystick.

The first line denoted by u_1 shows the true user utterance, which is the one that has been extracted manually from user audio recordings. The following line denoted by $\tilde{u}_1$ is the recognized user utterances by automatic speech recognition (ASR). Finally, the line denoted by a_1 shows the performed action in response to the ASR output at the time of collecting the dialogues. First, the users may say a command in different ways. For instance for turning right, the user may say *turn right a little please*, *turn right*, *right a little*, or *right*.

Furthermore, challenges can occur when ASR output is imperfect. That is, for each dialogue utterance recognized by ASR, the dialogue agent must estimate the human user intent and then to perform the best action that satisfies the user intent. The recognized utterance by ASR is not reliable for decision making. For instance, the first utterance, u_1: "turn right a little", shows the true user utterance. The ASR output for this utterance is, $\tilde{u}_1$: [10 writer little]. As such, the action performed by SmartWheeler at this dialogue turn is, the general query action u_1: PLEASE REPEAT YOUR COMMAND.

We are working on automatically learning the dialogue strategy by learning the dialogue model components such as user intents, dialogue features, and the agent's reward function from data. Our data includes dialogues between human users and expert agents (such as caregivers).

2 Theoretical approach

Sequential Decision Making: Figure 1 shows the cycle of interaction between an agent (such as a dialogue agent or a caregiver) and its environment (such as a human user). The agent performs an action and receives an observation in return. The observation can be used by the agent, for in-

Proceedings of the 5th Workshop on Speech and Language Processing for Assistive Technologies (SLPAT), pages 13–19,
Baltimore, Maryland USA, August 26 2014.

u_1 :	turn right	u_1 :	turn right
$\tilde{u}_1$:	[turn right]	$\tilde{u}_1$:	[turn right]
a_1 :	TURN RIGHT	s_1 :	*turn-right-little, 60%*
u_2 :	turn right a little	u_2 :	turn right a little
$\tilde{u}_2$:	**[10 writer little]**	$\tilde{u}_2$:	**[10 writer little]**
a_2 :	**PLEASE REPEAT YOUR COMMAND**	s_2 :	***turn-right-little, 90%***
u_3 :	turn right a little	u_3 :	turn right a little
$\tilde{u}_3$:	[turn right to lead a]	$\tilde{u}_3$:	[turn right to lead a]
a_3 :	TURN RIGHT A LITTLE	s_3 :	*turn-right-little, 90%*
u_4 :	stop	u_4 :	stop
$\tilde{u}_4$:	[stop]	$\tilde{u}_4$:	[stop]
a_4 :	STOP	s_4 :	*stop, 90%*

Table 1: Left: A sample from the SmartWheeler dialogues (Pineau et al., 2011). Right: results of learning human user intents from patients' noisy dialogues.

stance to update its state and reward. The reward works as reinforcement from the environment that shows how well the agent performed. In sequential decision making, the agent is required to make decision for sequence of states rather than making a one-shot decision. Then, the sequential decision making is performed with the objective of maximizing the long term rewards. The sequence of actions is called a strategy, and the major question in sequential decision making is how to find a near optimal strategy.

Reinforcement learning (RL): RL in (partially observable) Markov decision processes, so called the (PO)MDPs, is a learning approach in sequential decision making. In particular, (PO)MDPs have been successfully applied in dialogue agents (Roy et al., 2000; Zhang et al., 2001; Williams, 2006; Thomson and Young, 2010; Gašić, 2011). The (PO)MDP framework is a formal framework to represent uncertainty explicitly while supporting automated strategy solving. Specifically, it is an optimization framework that supports automated strategy solving by maximizing a "reward function".

3 Objective

SDS (Spoken dialogue system) researchers have addressed several practical challenges of applying (PO)MDPs to SDS (Williams, 2006; Paek and Pieraccini, 2008). Specifically, estimating the user model and the reward function is a significant challenge since these model components have a direct impact on the optimized dialogue strategy. Furthermore, the reward function is perhaps the most hand-crafted aspect of the optimization frameworks such as (PO)MDPs (Paek and Pieraccini, 2008). Using *inverse reinforcement learning* (IRL) techniques, a reward function can be determined from expert actions (such as caregiver actions) (Ng and Russell, 2000). Fortunately, learning the reward function using IRL methods have already been proposed for the general (PO)MDP framework (Ng and Russell, 2000; Kim et al., 2011), paving the way for investigating its use for dialogue (PO)MDPs. In this context, the IRL algorithms require dialogue features (for instance ASR recognitions with their confidence scores) for representing the reward function. Extracting relevant dialogue features is important since the dialogue features and their representation highly affect the learned reward function and finally the optimized strategy.

Thus, our goals include building (PO)MDP-based dialogue technologies that optimizes the dialogue strategy through learning user intents and the user model, and reward function from dialogue data, as follows:

1. Learning user intents and the user model from collected dialogues, i.e., ASR recognitions, or directly from acoustic data.

2. Learning the reward function.

   (a) Learning useful dialogue features.

   (b) Representing features in IRL for learning the reward function.

Recall Figure 1 that shows the cycle of interaction between an agent (such as a dialogue agent or a caregiver) and its environment (such as a human user). In this figure, circles represent the learned models. The model denoted by (PO)MDP includes the (PO)MDP model components, without

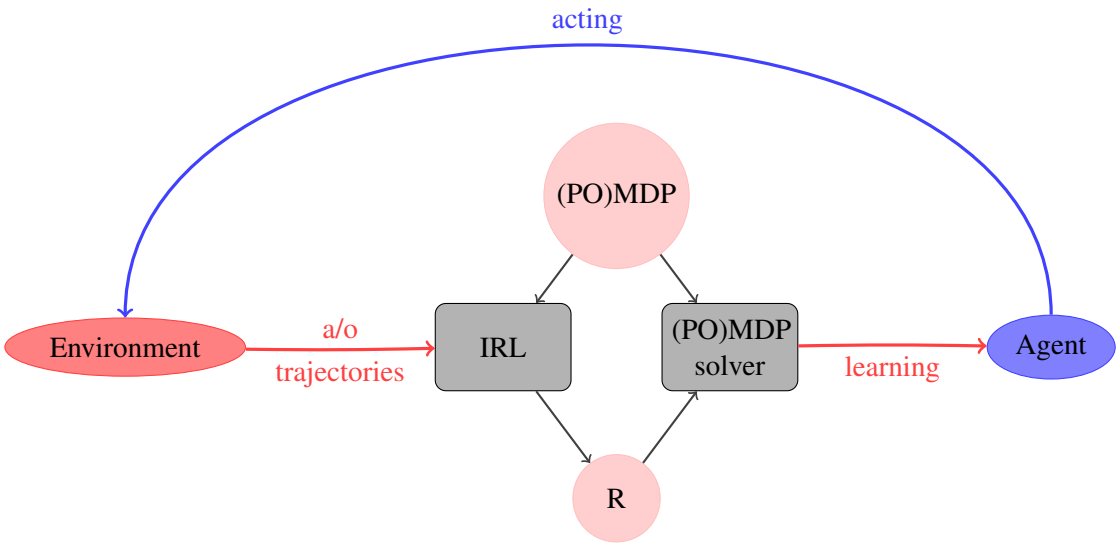

Figure 1: The cycle of acting/learning between the agent and environment. The circles represent the models. The model denoted by (PO)MDP includes the (PO)MDP model components, without a reward function, learned from step 1 in the objective section. The learned (PO)MDP model together with expert action/observation trajectories are used in IRL to learn the reward function denoted by R, in step 2 in the objective section. The learned (PO)MDP and reward function are used in the (PO)MDP solver to learn/update the strategy.

a reward function, which have been learned from step 1 above. The learned (PO)MDP together with action/observation trajectories are used in IRL to learn the reward function, denoted by R. Then, the learned (PO)MDP and the reward function are used in a (PO)MDP solver to learn/update the optimal strategy.

4 SmartWheeler data

The SmartWheeler project aims to build an intelligent wheelchair for persons with disabilities (Pineau et al., 2011). In particular, SmartWheeler aims to minimize the physical and cognitive load required in steering it. This project has been initiated in 2006, and a first prototype, shown in Figure 2, was built in-house at McGill's Center for Intelligent Machines.

We used the dialogues collected by SmartWheeler to develop dialogue (PO)MDPs, learned primarily from data. The data includes eight dialogues with healthy users and nine dialogues with target users of SmartWheeler (Pineau et al., 2011). The dialogues with target users, who are the elderly, are somehow more noisy than the ones with healthy users. More specifically, the average word error rate (WER) equals 13.9%

Figure 2: The SmartWheeler robot platform (Pineau et al., 2011).

for the healthy user dialogues and 18.5% for the target user dialogues. In order to perform our experiments on a larger amount of data, we used all the healthy and target user dialogues. In total, there are 2853 user utterances and 422 distinct words in the SmartWheeler dialogues.

5 Learning user intents from data

We learned the (PO)MDP states by learning the user intents occurred in the dialogue set using a topic modeling approach, i.e., Hidden Topic

Markov Model (HTMM) (Gruber et al., 2007). HTMM is a variation of Latent Dirichlet Allocation (LDA) which learns topics from text based on co-occurrence of words and using Dirichlet distribution for generating the topics of text documents (Blei et al., 2003). HTMM adds Markovian assumption to the LDA model in order to exploit the Markovian property between sentences in the documents. Thus, HTMM can be seen both as a variation of Hidden Markov Model (HMM) and a variation of LDA.

Our experimental results showed that HTMM learns proper user intents that can be used as dialogue states, and is able to exploit the Markovian property between dialogue utterances, adequately. The learned states, using our proposed methods, from SmartWheeler data are as follows: s_1 : *move-forward-little*, s_2 : *move-backward-little*, s_3 : *turn-right-little*, s_4 : *turn-left-little*, s_5 : *follow-left-wall*, s_6 : *follow-right-wall*, s_7 : *turn-degree-right*, s_8 : *go-door*, s_9 : *set-speed*, s_{10} : *follow-person*, s_{11} : *stop*. Table 3 shows the learned user intents, five of them, with their top-four words, i.e., the intent *keywords*.

Table 1 (right) shows results of HTMM application on SmartWheeler for the example shown in Table 1 (left). For instance, the second utterance shows that the user actually uttered *turn right a little*, but it is recognized as *10 writer little* by ASR. The most probable intent returned by HTMM for this utterance is s_3 *: turn-right-little* with 90% probability. This is because HTMM considers Markovian property for deriving intents. As a result, in the second turn it estimates correctly the true user intent based on the user intent in the first turn.

The list of all SmartWheeler actions are shown in Table 2. Each action is the right action of one state (the user intent for a specific command). So, ideally, there should be 24 states for SmartWheeler dialogues (There are 24 actions other than the general query action: REPEAT). However, we only learned 11 of the states, mainly because of the number of dialogues. That is, not all of the states appeared in the data frequently enough. There are also states that do not appear in dialogues at all.

6 Learning reward functions from data

In this section, we experiment our implementation of the trajectory-based MDP-IRL algorithm proposed by (Ng and Russell, 2000). The IRL experiments are designed to verify if the introduced IRL methods are able to learn a reward function for the expert strategy, where the expert strategy is represented as a (PO)MDP strategy. That is, the expert strategy is the strategy that the underlying (PO)MDP framework optimizes. The MDP expert strategy for each of the (PO)MDP state is represented in Table 4. This strategy suggests performing the right action of each state.

a_1	DRIVE FORWARD A LITTLE
a_2	DRIVE BACKWARD A LITTLE
a_3	TURN RIGHT A LITTLE
a_4	TURN LEFT A LITTLE
a_5	FOLLOW THE LEFT WALL
a_6	FOLLOW THE RIGHT WALL
a_7	TURN RIGHT DEGREE
a_8	GO THROUGH THE DOOR
a_9	SET SPEED TO MEDIUM
a_{10}	FOLLOW THE WALL
a_{11}	STOP
a_{12}	TURN LEFT
a_{13}	DRIVE FORWARD
a_{14}	APPROACH THE DOOR
a_{15}	DRIVE BACKWARD
a_{16}	SET SPEED TO SLOW
a_{17}	MOVE ON SLOPE
a_{18}	TURN AROUND
a_{19}	PARK TO THE RIGHT
a_{20}	TURN RIGHT
a_{21}	DRIVE FORWARD METER
a_{22}	PARK TO THE LEFT
a_{23}	TURN LEFT DEGREE
a_{24}	PLEASE REPEAT YOUR COMMAND

Table 2: The list of the possible actions, performed by SmartWheeler.

6.1 MDP-IRL learned rewards

We applied the MDP-IRL algorithm on SmartWheeler dialogue MDP described above using the introduced keyword features in Table 5. The algorithm was able to learn a reward function in which the strategy equals the expert strategy for all states, (the expert strategy shown in Table 4). Table 6 shows the learned reward function. Note that, for instance for state s_3: *turn-right-little*, the reward of performing both actions a_3: TURN RIGHT A LITTLE and a_4: FOLLOW THE RIGHT WALL is close to 1. Nevertheless, the optimized strategy for this reward function suggest the correct action, i.e., TURN RIGHT A LITTLE for this state (*turn-right-little*).

intent 1		*intent 2*		*intent 3*		*intent 4*		...	*intent 11*	
forward	18.0%	**backward**	38.0%	**right**	20.9%	**left**	18.9%		**stop**	94.2%
move	16.1%	drive	33.3%	turn	17.1%	turn	17.1%		stopp	02.2%
little	11.4%	little	10.9%	little	13.1%	little	13.8%		scott	00.7%
drive	08.1%	top	01.7%	bit	07.4%	right	09.0%	... ...	but	00.2%
...	...	...	...	...	...	...	...		...	...

Table 3: The learned user intents from the SmartWheeler dialogues and their top words. Each percentage shows the probability of each word given the intent.

state	state description	expert action	expert action description
s_1	*move-forward-little*	a_1	DRIVE FORWARD A LITTLE
s_2	*move-backward-little*	a_2	DRIVE BACKWARD A LITTLE
s_3	*turn-right-little*	a_3	TURN RIGHT A LITTLE
s_4	*turn-left-little*	a_4	TURN LEFT A LITTLE
s_5	*follow-left-wall*	a_5	FOLLOW THE LEFT WALL
s_6	*follow-right-wall*	a_6	FOLLOW THE RIGHT WALL
s_7	*turn-degree-right*	a_7	TURN RIGHT DEGREES
s_8	*go-door*	a_8	GO THROUGH THE DOOR
s_9	*set-speed*	a_9	SET SPEED TO MEDIUM
s_{10}	*follow-wall*	a_{10}	FOLLOW THE WALL
s_{11}	*stop*	a_{11}	STOP

Table 4: The learned strategy using the learned dialogue MDP from SmartWheeler dialogues.

	forward	backward	**right**	**left**	turn	go	for	top	stop
s_1	1	0	0	0	0	0	0	0	0
s_2	0	1	0	0	0	0	0	0	0
s_3	0	0	**1**	0	0	0	0	0	0
s_4	0	0	0	**1**	0	0	0	0	0
s_5	0	0	0	**1**	0	0	0	0	0
s_6	0	0	**1**	0	0	0	0	0	0
s_7	0	0	0	0	1	0	0	0	0
s_8	0	0	0	0	0	1	0	0	0
s_9	0	0	0	0	0	0	1	0	0
s_{10}	0	0	0	0	0	0	0	1	0
s_{11}	0	0	0	0	0	0	0	0	1

Table 5: Keyword features for the SmartWheeler dialogues.

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	...	REPEAT
s_1	1.0	0	0	0	0	0	0	0	0	0	0	0	...	0
s_2	0	1.0	0	0	0	0	0	0	0	0	0	0	...	0
s_3	0	0	**1.0**	0	0	**1.0**	0	0	0	0	0	0	...	0
s_4	0	0	0	**1.0**	**1.0**	0	0	0	0	0	0	0	...	0
s_5	0	0	0	**1.0**	**1.0**	0	0	0	0	0	0	0	...	0
s_6	0	0	**1.0**	0	0	**1.0**	0	0	0	0	0	0	...	0
s_7	0	0	0	0	0	0	1.0	0	0	0	0	0	...	0
s_8	0	0	0	0	0	0	0	1.0	0	0	0	0	...	0
s_9	0	0	0	0	0	0	0	0	1.0	0	0	0	...	0
s_{10}	0	0	0	0	0	0	0	0	0	1.0	0	0	...	0
s_{11}	0	0	0	0	0	0	0	0	0	0	1.0	0	...	0

Table 6: The learned reward function for the learned dialogue MDP from SmartWheeler dialogues using keyword features.

6.2 Choice of features

IRL needs features to represent the reward function. We propose *keyword* features for applying IRL on the learned dialogue MDP/POMDP from SmartWheeler. The keyword features are automatically learned as the top-one words for each user intent (see Table 3). There are nine learned keywords:

forward, backward, right, left, turn, go, for, top, stop.

The keyword features for each state of SmartWheeler dialogue POMDP are represented in a vector, as shown in Table 5. The figure shows that states s_3, (*turn-right-little*) and s_6 (*follow-right-wall*) share the same features, i.e., *right*. Moreover, states s_4 (*turn-left-little*) and s_5 (*follow-left-wall*) share the same feature, i.e., *left*. In our experiments, we used *keyword-action-wise* feature representation. Such features include an indicator function for each pair of state-keyword and action. Thus, the feature size for SmartWheeler equals $216 = 9 \times 24$ (9 keywords and 24 actions).

Note that the choice of features is application dependent. The reason for using keywords as state features is that in the intent-based dialogue applications the states are the dialogue intents, where each intent is described as a vector of k-top words from the domain dialogues. Therefore, the keyword features are relevant features for the states.

7 Conclusion

In this paper, we described our our systematic approach for learning dialogue (PO)MDP model components from unannotated dialogues. In our approach, we start by learning the dialogue (PO)MDP states, i.e., the learned user intents from data. The learned states were then used for learning the user model. Building off these model components, we learned the agent's reward function by implementing a model-based IRL algorithm. We demonstrated our experiments on data collected in a healthcare domain to learn the dialogue model components solely from data.

8 Ongoing work

We are working on a variation of MDP-IRL algorithm, that is a model-free trajectory-based MDP-IRL algorithm. In the model-free MDPs, the states are usually presented using features (and thus there is no defined/learned transition model). Then, model-free MDP algorithms are used for estimating the optimal strategy of such MDPs. Model-free MDPs can be used in the place of POMDPs where state features are analogous to observations.

In this context, data analysis for feature selection is highly important. Dialogue features can be used to represent dialogue situations (as well as the observations in the dialogue POMDPs). Moreover, the IRL algorithms require (dialogue) features for representing the reward function. As mentioned earlier, the reward function of (PO)MDPs highly affects the optimized strategy. A relevant reward function to the dialogue agent and users can only be learned by studying and extracting relevant features from the dialogue domain. We would like to learn the relevant and proper features that are suitable for both state features as well as the reward representation. In particular, we are going to use the experts' (caregivers') strategies in the place of a (PO)MDP strategy in order to learn a reward function that accounts for caregivers' strategies.

9 Acknowledgment

The authors thank Jason D. Williams and Suhrid Balakrishnan for helpful discussions in the early development of this work. The authors also thank Joelle Pineau for providing them with the Smartwheeler data. The dataset has been collected with contributions from researchers at McGill University, École Polytechnique de Montréal, Université de Montréal, and the Centre de réadaptation Lucie-Bruneau and Constance-Lethbridge. The authors thank Ethan Selfridge for his help in proofreading. Last but not least, many thank to FQRNT (Fonds Québécois de la recherche sur la nature et les technologies) for partial financial support of this work.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jennifer Boger, Pascal Poupart, Jesse Hoey, Craig Boutilier, Geoff Fernie, and Alex Mihailidis. 2005. A decision-theoretic approach to task assistance for persons with dementia. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, pages 1293–1299, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Milica Gašić. 2011. *Statistical Dialogue Modelling*. Ph.D. thesis, Department of Engineering, University of Cambridge.

Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. 2007. Hidden topic Markov models. In *Artificial Intelligence and Statistics (AISTATS'07)*, San Juan, Puerto Rico, USA.

D. Kim, J. Kim, and K.E. Kim. 2011. Robust performance evaluation of POMDP-based dialogue systems. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):1029–1040.

Andrew Y. Ng and Stuart J. Russell. 2000. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning (ICML'00), Stanford, CA, USA*.

T. Paek and R. Pieraccini. 2008. Automating spoken dialogue management design using machine learning: An industry perspective. *Speech Communication*, 50(8):716–729.

Joelle Pineau, Robert West, Amin Atrash, Julien Villemure, and Francois Routhier. 2011. On the feasibility of using a standardized test for evaluating a speech-controlled smart wheelchair. *International Journal of Intelligent Control and Systems*, 16(2):124–131.

Nicholas Roy, Joelle Pineau, and Sebastian Thrun. 2000. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (ACL'00), Hong Kong*.

Frank Rudzicz, Rozanne Wilson, Alex Mihailidis, Elizabeth Rochon, and Carol Leonard. 2012. Communication strategies for a computerized caregiver for individuals with alzheimer's disease. In *Proceedings of the Third Workshop on Speech and Language Processing for Assistive Technologies, (SLPAT'12)*, pages 47–55, Montreal, Quebec, Canada. Association for Computational Linguistics.

Blaise Thomson and Steve Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech and Language*, 24(4):562–588.

Jason D. Williams. 2006. *Partially Observable Markov Decision Processes for Spoken Dialogue Management*. Ph.D. thesis, Department of Engineering, University of Cambridge.

Bo Zhang, Qingsheng Cai, Jianfeng Mao, and Baining Guo. 2001. Planning and acting under uncertainty: A new model for spoken dialogue system. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI'01), Seattle, Washington, USA*, August.