Preference-learning based Inverse Reinforcement Learning for Dialog Control

Hiroaki Sugiyama, Toyomi Meguro, Yasuhiro Minami

NTT Communication Science Laboratories, Kyoto, Japan

{sugiyama.hiroaki,meguro.toyomi,minami.yasuhiro}@lab.ntt.co.jp

Abstract

Dialog systems that realize dialog control with reinforcement learning have recently been proposed. However, reinforcement learning has an open problem that it requires a reward function that is difficult to set appropriately. To set the appropriate reward function automatically, we propose preference-learning based inverse reinforcement learning (PIRL) that estimates a reward function from dialog sequences and their pairwise-preferences, which is calculated with annotated ratings to the sequences. Inverse reinforcement learning finds a reward function, with which a system generates similar sequences to the training ones. This indicates that current IRL supposes that the sequences are equally appropriate for a given task; thus, it cannot utilize the ratings. In contrast, our PIRL can utilize pairwise preferences of the ratings to estimate the reward function. We examine the advantages of PIRL through comparisons between competitive algorithms that have been widely used to realize the dialog control. Our experiments show that our PIRL outperforms the other algorithms and has a potential to be an evaluation simulator of dialog control.

1. Introduction

Dialog control is a major topic in dialog system research area that decides appropriate system actions for user states. Previous studies realize the dialog control with rule-based approaches that a human defines system actions for each user state (i.e., rules) [1]; however, if the number of rules increases, it is difficult to define the rules consistently. To avoid this difficulty, recent studies [2, 3] adopt reinforcement learning (RL) to realize the dialog control. RL automatically decides the appropriate system actions for user states in order to maximize the total expected rewards received from a reward function designed by a human; therefore, if a reward function meets objectives of a dialog, RL automatically decides the optimal system actions for the objectives. This condition is easy to meet when the objective is represented with an obvious goal (i.e., goal-oriented dialog) such as troubleshooting [2]. However, when an obvious goal does not exist (i.e., less-goal oriented dialog) the reward function is difficult to design to meet the objectives. For example, considering building a dialog system that aims to provide counseling treatment, we don't know where the obvious goal is. For this kind of task, some studies annotate ratings to dialog corpus and utilize them as reward function [3, 4], However, the ratings have an ambiguity that evaluators annotate individual ratings even if they intend to the same appropriateness for the objectives. Thus, the reward function does not meet the objectives.

To set an appropriate reward function automatically, inverse reinforcement learning (IRL) has been proposed [5, 6] and is adopted for dialog control [7, 8]. IRL finds a reward function, with which a system generates similar sequences as the training ones in corpora. This indicates that all the sequences are assumed as equally optimal in the current IRL studies. Therefore, if the training sequences contain admissible (but not optimal) ones, we have to discard them in advance; otherwise (i.e., including the admissible sequences), non-optimal sequences will be generated with the estimated reward function. If the evaluation of the sequences are clearly different between successful and failure cases, this discard will not have harmful effects to the reward estimation. However, when we cannot decide the obviously distinguished evaluation, this discard causes a critical problem that the IRLs cannot distinguish admissible and fatal sequences; besides, it makes it difficult to estimate negative values of the reward function.

In this paper, we propose a preference-learning based inverse reinforcement learning (PIRL) that estimates a reward function from sequences with ratings. Preference-learning is a subfield of supervised learning that learns a preference (order relations) model from observed preference information. We use only the preference of the ratings instead of the absolute values of the ratings since the relation between the absolute values of the ratings and their appropriateness are inconsistent between annotators; on the other hand, the preferences are robust to this ambiguity and are consistent between the annotators.

2. Preference-learning based Inverse Reinforcement Learning

In this section, we define RL for dialog control and explain our preference-learning based IRL.

2.1. Reinforcement Learning for Dialog Control

RL and IRL are generally represented using a Markov decision process $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, R)$, where $\mathcal{S}$ is a finite set of user states; $\mathcal{A}$ is a finite set of system actions; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ is a transition function; $\gamma \in (0, 1]$ is a discount factor of future rewards; and $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{R}$ is the reward function representing a target task. In addition, we define a dialog sequence ζ_i that consists of $\{s_{i,t}, a_{i,t} : 0 \leq t < T\}$; i.e., the sequence of pairs of a user state and a system action. RL aims to learn a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ in order to maximize total expected rewards. The policy is generally defined as $\pi(s) = \arg\max_a Q(s, a)$, where $Q(s, a)$ is an action-value function that represents the appropriateness of a system action a at a user state s; thus, the difficulty of RL is in the estimation of $Q(s, a)$.

2.2. Preference-learning based Inverse Reinforcement Learning

Most less-goal oriented dialog corpora have ratings for each dialog sequence [3, 4]. The conventional IRL studies [6, 7, 8] assume that each training sequence is equally appropriate for a given task; thus, conventional IRL are not supposed to adopt to sequences with ratings. To estimate a reward function using sequences with ratings, we propose a preference-learning based inverse reinforcement learning (PIRL). Our algorithm estimates a reward function, with which a system ranks training sequences $\boldsymbol{\zeta}$ with the same preferences as the training one $\boldsymbol{o}^*$. In this study, we adopt pairwise preference [9] to represent the preference model for the sake of simplicity of implementation. We define the pairwise preference using the magnitude relations of the training ratings $\boldsymbol{e}^*$ as $o^*_{i,j} = \frac{e^*_i - e^*_j}{|e^*_i - e^*_j|} = \{-1, 0, 1\}$. We model the pairwise preference learning as a binary classification using only the pairs that have a relation $o^*_{i,j} = 1$ (i.e., $e^*_i > e^*_j$) as

$$L(\boldsymbol{\theta}) = P(\boldsymbol{o}^*|\boldsymbol{\zeta}, \boldsymbol{\theta}) = \sum_{i,j:e^*_i > e^*_j} \frac{(1 + o^{\boldsymbol{\theta}}_{i,j})^{\frac{1+o^*_{i,j}}{2}} \cdot (1 - o^{\boldsymbol{\theta}}_{i,j})^{\frac{1-o^*_{i,j}}{2}}}{2M}, \quad (1)$$

where $o^{\boldsymbol{\theta}}_{i,j} = \frac{e^{\boldsymbol{\theta}}_i - e^{\boldsymbol{\theta}}_j}{|e^{\boldsymbol{\theta}}_i - e^{\boldsymbol{\theta}}_j|}$ is a simulated-preference of sequences ζ_i and ζ_j under an estimated reward function $\boldsymbol{\theta}$, and M is the number of training sequence pairs. Here, we define simulated-ratings under $\boldsymbol{\theta}$ as

$$e^{\boldsymbol{\theta}}_i = \sum_t Q^{\boldsymbol{\theta}}(s_{i,t}, a_{i,t}), \quad (2)$$

where $Q^{\boldsymbol{\theta}}(s_{i,t}, a_{i,t})$ is a simulated action-value function under $\boldsymbol{\theta}$ explained as follows.

Basically, our PIRL iteratively calculates the simulated-preferences of each training sequence pairs with a current reward function $\boldsymbol{\theta}^n$ and updates it with derivation $\frac{\partial L_{i,j}}{\partial \boldsymbol{\theta}^n}$ calculated for each pair that has different preferences (i.e., $o^*_{i,j} \neq o^{\boldsymbol{\theta}}_{i,j}$). Its details are illustrated in algorithm 1.

The algorithm's input data are a set of sequences $\boldsymbol{\zeta}$ with their preferences $\boldsymbol{o}^*$. In Step 1, it calculates current action-value function $Q^{\boldsymbol{\theta}^n}(s, a)$ with current reward function $\boldsymbol{\theta}^n$. Since our PIRL requires the derivation of $Q^{\boldsymbol{\theta}^n}(s, a)$ for updating reward $\boldsymbol{\theta}^n$, we define the action-value function with an approximate version of value iteration algorithm as

$$Q^{\boldsymbol{\theta}^n}(s, a) = \sum_{s'} \{\theta^n(s, s') P_T(s'|s, a) + \max_{a'} \sum_{s''} P_T(s'|s, a) \theta^n(s', s'') P_T(s''|s', a')\}, \quad (3)$$

where $\theta(s, s')$ is a reward value when the user state is transitioned from s to s' and $P_T(s'|s, a)$ is a transition probability from s to s' with system action a. Next, the system calculates simulated-ratings $\boldsymbol{e}^{\boldsymbol{\theta}^n}$ with (2) and simulated-preferences $o^{\boldsymbol{\theta}^n}_{i,j}$. In Step 3, if preferences $o^*_{i,j}$ and $o^{\boldsymbol{\theta}^n}_{i,j}$ are different, it calculates the derivation of the reward function for the pair ζ_i and ζ_j with

$$\frac{\partial L_{i,j}}{\partial \boldsymbol{\theta}^n} \propto o^*_{i,j} \{\sum_{s_{i,t}, a_{i,t} \in \zeta_i} \frac{\partial Q^{\boldsymbol{\theta}^n}(s_{i,t}, a_{i,t})}{\partial \boldsymbol{\theta}^n} - \sum_{s_{j,t}, a_{j,t} \in \zeta_j} \frac{\partial Q^{\boldsymbol{\theta}^n}(s_{j,t}, a_{j,t})}{\partial \boldsymbol{\theta}^n}\}. \quad (4)$$

Each factor of $\frac{\partial Q(s,a)}{\partial \boldsymbol{\theta}}$ is formulated as

$$\frac{\partial Q(s, a)}{\partial \theta(s_1, s_2)} = \delta_{s,s_1} P_T(s_2|s_1, a) + P_T(s_1|s, a) P_T(s_2|s_1, a'_{s_1}),$$

where $a'_s = \arg\max_a \sum_s \theta(s, s') P_T(s'|s, a)$ and $\delta_{s,s1}$ is a Kronecker delta.

The algorithm sums up the derivation as $\frac{\partial L}{\partial \boldsymbol{\theta}^n} = \sum_{i,j:o^*_{i,j} \neq o^{\boldsymbol{\theta}^n}_{i,j}} \frac{\partial L_{i,j}}{\partial \boldsymbol{\theta}^n}$ and iteratively updates the reward function with the L-BFGS algorithm [10].

input : Training sequences $\boldsymbol{\zeta}$ with their preferences $\boldsymbol{o}^*$
output: Estimated reward function $\boldsymbol{\theta}$

0. Initialize reward function $\boldsymbol{\theta}^0$.

for *n* **to** *N* **do**
  1. Calculate action-value function $Q^{\boldsymbol{\theta}^n}(s, a)$ with current reward function $\boldsymbol{\theta}^n$ (3).
  2. Evaluate training sequences $\boldsymbol{\zeta}$ with $Q^{\boldsymbol{\theta}^n}(s, a)$ (2) and calculate simulated-preferences $\boldsymbol{o}^{\boldsymbol{\theta}^n}$.
  foreach $\{i, j | i < j, e^*_i \neq e^*_j\}$ **do**
    if $o^*_{i,j} \neq o^{\boldsymbol{\theta}^n}_{i,j}$ **then**
      3. Calculate $\frac{\partial L_{i,j}}{\partial \boldsymbol{\theta}^n}$ (4).
    end
  end
  4. Evaluate convergence with L.
  5. Update $\boldsymbol{\theta}^n$ using the L-BFGS algorithm.
end

Algorithm 1: Preference-learning based Inverse Reinforcement Learning

3. Experiments

Our PIRL estimates an appropriate reward function from dialog sequences with preferences calculated with the annotated ratings. In this section, we examine the effectiveness of our algorithm through the comparison between

the following three algorithms: Maximum Entropy IRL, RL with profit sharing, and our PIRL.

Maximum Entropy IRL is a state-of-the-art IRL algorithm [11] that estimates reward function using only high-rated sequences. Through the comparison with this, we examine the influence of the data sparseness caused by discarding low-rated sequences.

RL with profit sharing is a popular approach to estimate action-value function using annotated ratings as reward function [12]. Through the comparison with this, we examine the influence of the ambiguity of rating-annotation, since this approach utilizes the absolute values of the ratings as a reward function.

3.1. Dialog Data

We used the dialog data collected in our previous study [3]. The study aims to build a listening-oriented dialog system that attentively listens to other dialog participant. We collected 1259 listening-oriented dialogs using human subjects who consisted of ten listeners (five males and five females) and 37 speakers (18 males and 19 females) and labeled each sentence of the collected data using 32 dialog-act tags (totally, 67801 dialog-act tags are contained in the corpus). The collected dialogs were evaluated using two third-party participants (annotators), who were neither listeners nor speakers in our dialog data collection. The annotators evaluated each dialog sequence in terms of how they would have felt "being heard" after the dialog if they had been the speaker of the dialog in question. They provided ratings on a 7-point Likert scale for each dialog.

We used the speaker's dialog-act tags as user state space $\mathcal{S}$, and the listener's dialog-act tags as system action space $\mathcal{A}$. When one utterance contains several sentences, the algorithms blend action-value functions of plural user states as $Q'(\boldsymbol{s}, a) = \frac{1}{|\boldsymbol{s}|} \sum_{s' \in \boldsymbol{s}} Q(s', a)$. On the other hand, the algorithms can generate only one system action for each utterance.

3.2. Evaluation Criteria

To examine whether each algorithm can generate the optimal sequences, an agreement rate of sequences between the testing and the generated by the system seems a straightforward criterion. However, since several system actions appear for each user state in the training sequences, and several system actions are suitable for each user state, it is infeasible to generate the system actions in the testing sequences accurately. Therefore, we argue that the agreement rate of the sequences is not adequate criterion to evaluate the algorithms.

To compare the algorithms, we defined three criteria: agreement rates of the preferences, correlation coefficients of the preference, and "expected-ratings". The agreement rates and the correlation coefficients of the preferences are consistent criteria between algorithms since they are robust to the ambiguity of rating-annotation. If an agreement rate and a correlation coefficient of the preferences between the algorithm and an annotator are equivalent to those between annotators, we can utilize the algorithm as an evaluation simulator. In addition, as an intuitive criterion, we add "expected-ratings" that algorithms are expected to gain in real dialog. While we would like the annotators to evaluate the sequences, the evaluation is difficult even for annotators since the sequences contain only user state and system actions instead of sentences. Thus, we calculate the expected-ratings as the averaged ratings of the sequences that gain top-n highest/lowest simulated-ratings. We assume that the sequences with top-n highest simulated-ratings resemble those that algorithms will generate in real dialog; thus, we believe that the expected-ratings from the sequences with the highest simulated-ratings (i.e., high expected-ratings) resemble ratings that algorithms will gain in real dialog. On the other hand, expected-ratings from sequences with the lowest simulated-ratings (i.e., low expected-ratings) are also important to examine whether the algorithms can inhibit generating inappropriate system actions.

3.3. Experiment Settings

We divide data (sequences and their ratings) into training, development, testing sets with two settings: all-data and selected-data. In the all-data setting, we make training and development sets with one annotator's data, and testing set with another's data. Since the data consists of the same sequences between the annotators, the training, development and testing sets are divided so that their sequences have no overlaps each other.

In the selected-data setting, at first we select data that has similar ratings between the annotators (the difference is equal to or lower than 1); and then, we divide this data into the 300 for training, 300 for development, and 111 for testing sets at random. We added 200 randomly generated sequences with the lowest-ratings to the training set for the sake of increasing variation of the data since our data were expected to contain few fatal sequences since the dialogs were performed by humans. See Table 1 for the statistics.

	All-data	Selected-data
# train sequences	700 (ME:113)	500 (ME:88)
# train pairs	149515	75414
# dev. sequences	300 (ME:300)	300 (ME:150)
# dev. pairs	18177	16768
# test sequences	459	111
# test pairs	55310	2236

Table 1: Statistics of the data sets. The "ME:*" means the case of Maximum Entropy IRL that utilizes only data with high ratings (equal to or higher than 4).

3.4. Results and Discussion

Figure 1 shows the agreement rates of the preferences between the algorithms and an annotator. This illustrates

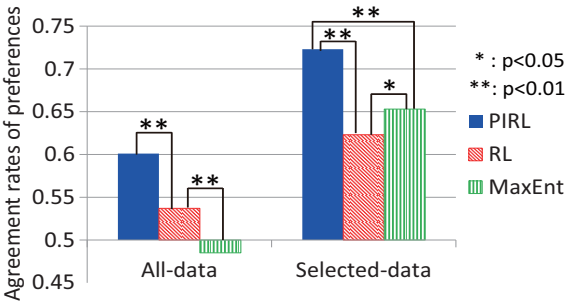

Figure 1: Comparison of the agreement rates of preferences. The agreement rate between the annonators was 0.632 in the all-data setting and 0.925 in the selected-data setting, and the random baseline was 0.5.

that our PIRL significantly outperforms the other algorithms in both settings. The reason why the agreement rates of the selected-data setting are higher than that of the all-data setting is that we can remove the data that have opposite ratings between the annotators.

	All-data			Selected-data		
	Corr.	High	Low	Corr.	High	Low
PIRL	**0.200**	**4.66**	**3.46**	**0.363**	**5.06**	3.86
RL	0.069	4.46	4.13	0.218	4.46	**3.40**
MaxEnt	-0.001	3.80	3.60	0.285	4.66	3.93
Annotator	0.211	4.90	2.86	0.817	6.80	1.33

Table 2: Correlation coefficients and high/low expected-ratings. Annotator's coefficient is 0.211 in the all-data setting, and the average of the all-data ratings is 0.402 and the selected-data setting is 0.458. The higher/lower expected-ratings mean better performance in "High"/"Low" column.

Table 2 illustrates the correlation coefficients of the ratings and the high/low expected-ratings with the highest/lowest 15 sequences. Our PIRL shows higher correlation coefficients and high expected-ratings than the others, and lower expected-ratings in the all-data setting. This suggests that our PIRL estimates an appropriate reward function, with which a system with the reward function generates appropriate system actions. Besides, the agreement rate of the preferences and the correlation coefficients of our PIRL are similar to the annotator's ones, our PIRL has a potential to be an evaluation simulator.

4. Conclusions

We proposed a preference-learning based inverse reinforcement learning (PIRL) that estimates a reward function for dialog control from dialog sequences with ratings. The contribution of our study is to extend the range of applications of inverse reinforcement learning (IRL) from sequences with single appropriateness to sequences with various appropriateness; thus, our PIRL can utilize non-optimal data, which is discarded in previous IRL, using pairwise preference information calculated with the ratings. Besides, our experiments show that our PIRL has a potential to be an evaluation simulator.

There are some studies that have the similar objective as our present study. Silva et al. proposed IRL with evaluation that utilizes pairwise preferences; however, this study force annotators to evaluate each pair of simulated sequences in each learning iterations [13]. While Cheng et al. proposed reinforcement learning based on preference-learning, this requires comparisons for each action pair [14]. The evaluation costs taken in these studies are infeasible; on the other hand, our PIRL requires the rating evaluation only one time to each sequence pair; thus, it is easier to introduce than the other conventional algorithms.

Much work still remains. Since we examined the effectiveness of our PIRL with offline evaluation, we plan to evaluate our PIRL using online evaluation. Besides, it is very interesting topic to extend our discrete user state and system action spaces to continuous distributions. A promising idea for this purpose is a topic model like Hidden Topic Markov Models [8], which is used in dialog control with IRL.

5. References

[1] R. Wallace, "The Anatomy of A.L.I.C.E." *A.L.I.C.E. Artificial Intelligence Foundation, Inc.*, 2004.

[2] J. D. Williams, "Applying POMDPs to dialog systems in the troubleshooting domain," in *Proc. Workshop on Bridging the Gap*, 2007.

[3] T. Meguro, R. Higashinaka, Y. Minami, and K. Dohsaka, "Controlling Listening-oriented Dialogue using Partially Observable Markov Decision Processes," in *Proc. Coling*, 2010.

[4] J. D. Williams and S. Young, "The SACTI-1 corpus: Guide for research users", *Technical report, University Of Cambridge*, 2005.

[5] A. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Proc. ICML*, pp. 663–670, 2000.

[6] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proc. ICML*, 2004.

[7] S. Chandramohan, M. Geist, F. Lefevre, and O. Pietquin, "User Simulation in Dialogue Systems using Inverse Reinforcement Learning," in *Proc. Interspeech*, 2011.

[8] A. Boularias, H. Chinaei, and B. Chaib-draa, "Learning the Reward Model of Dialogue POMDPs from Data," in *Proc. NIPS 2010 Workshop on Machine Learning for Assistive Technologies (MLAT-2010)*, 2010.

[9] R. Herbrich, T. Graepel, and K. Obermayer, "Large margin rank boundaries for ordinal regression," in *Proc. Advances in Large Margin Classifiers*, MIT Press, vol. 88, no. 2, pp. 115–132, 2000.

[10] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical programming*, vol. 45, pp. 503–528, 1989.

[11] B. Ziebart, A. Maas, J. Bagnell, and A. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI*, pp. 1433–1438, 2008.

[12] J. Grefenstette, "Credit assignment in rule discovery systems based on genetic algorithms," *Machine Learning*, vol. 3, no. 2, pp. 225–245, 1988.

[13] V. Freire da Silva, A. Reali Costa, and P. Lima, "Inverse reinforcement learning with evaluation," in *Proc. ICRA*, 2006.

[14] W. Cheng, J. Fürnkranz, E. Hüllermeier, and S. Park, "Preference-based policy iteration: leveraging preference learning for reinforcement learning," in *Proc. ECML-PKDD*, pp. 312–327, 2011.