Learning Goal-oriented Dialogue Policy with Opposite Agent Awareness

Zheng Zhang[†], **Lizi Liao**[‡], **Xiaoyan Zhu**[†], **Tat-Seng Chua**[‡],
Zitao Liu[§], **Yan Huang**[§], **Minlie Huang**[†*]
[†] Department of Computer Science and Technology, Institute for Artificial Intelligence,
State Key Lab of Intelligent Technology and Systems,
Beijing National Research Center for Information Science and Technology, Tsinghua University
[‡] School of Computing, National University of Singapore, Singpore [§] TAL Education Group
[†]zhangz.goal@gmail.com {zxy-dcs,aihuang}@tsinghua.edu.cn
[‡]liaolizi.llz@gmail.com chuats@comp.nus.edu.sg [§]{liuzitao,galehuang}@100tal.com

Abstract

Most existing approaches for goal-oriented dialogue policy learning used reinforcement learning, which focuses on the target agent policy and simply treats the opposite agent policy as part of the environment. While in real-world scenarios, the behavior of an opposite agent often exhibits certain patterns or underlies hidden policies, which can be inferred and utilized by the target agent to facilitate its own decision making. This strategy is common in human mental simulation by first imaging a specific action and the probable results before really acting it. We therefore propose an opposite behavior aware framework for policy learning in goal-oriented dialogues. We estimate the opposite agent's policy from its behavior and use this estimation to improve the target agent by regarding it as part of the target policy. We evaluate our model on both cooperative and competitive dialogue tasks, showing superior performance over state-of-the-art baselines.

1 Introduction

In goal-oriented dialogue systems, dialogue policy plays a crucial role by deciding the next action to take conditioned on the dialogue state. This problem is often formulated using reinforcement learning (RL) in which the user serves as the environment (Levin et al., 1997; Rieser and Lemon, 2011; Lemon and Pietquin, 2012; Young et al., 2013; Fatemi et al., 2016; Zhao and Eskenazi, 2016; Dhingra et al., 2016; Su et al., 2016; Li et al., 2017; Williams et al., 2017; Liu and Lane, 2017; Lipton et al., 2018; Liu et al., 2018; Gao et al., 2019; Takanobu et al., 2019, 2020; Jhunjhunwala et al., 2020). However, different from symbolic-based and simulation-based RL tasks, such as chess (Silver et al., 2016) and video games (Mnih et al., 2015), which can get vast amounts of training interactions in low cost, dialogue systems require to learn directly from real users, which is too expensive.

Therefore, there are some efforts using simulation methods to provide an affordable training environment. One principle direction for mitigating this problem is to leverage human conversation data to build a user simulator, and then to learn the dialogue policy by making simulated interactions with the simulator (Schatzmann et al., 2006; Li et al., 2016; Gür et al., 2018).

However, there always exist discrepancies between simulated users and real users due to the inductive biases of the simulation model, which can lead to a sub-optimal dialogue policy (Dhingra et al., 2016).

Another direction is to learn the dynamics of dialogue environment during interacting with real user, and concurrently use the learned dynamics for RL planning (Peng et al., 2018; Su et al., 2018; Wu et al., 2018; Zhang et al., 2019b). Most of these works are based on Deep Dyna-Q (DDQ) framework (Sutton, 1990), where a world model is introduced to learn the dynamics (which is much like a simulated user) from real experiences. The target agent's policy is trained using both real experiences via direct RL and simulated experiences via a world-model.

In the above methods, both the simulated user and world model facilitate target policy learning by providing more simulated experiences and remain a black box for the target agent. That is, the target agent's knowledge about the simulated agents is still passively obtained through interaction and implicitly learned by the policy model updating as indirect try-and-error with real user. However, we argue that from the angle of a target agent, actively exploring the world with proper estimation would not only make user simulation

*Corresponding author.

Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 122–132

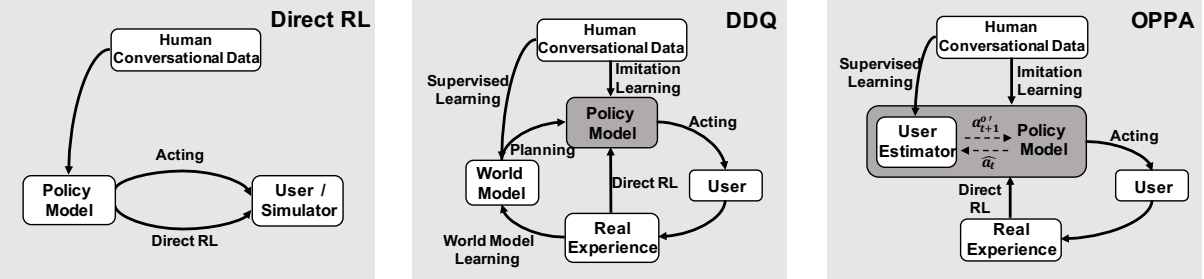

Figure 1: A comparison of dialogue policy learning a) with real/simulated user, b) with real user via DDQ and c) with real user guided by active user estimation.

more efficient but also improve the target agent's performance. In agreement with the findings from cognitive science, humans often maintain models of other people they interact with to capture their goals (Harper, 2014; Premack and Woodruff, 1978). And humans manage to use their mental process to simulate others' behavior (Gordon, 1986; Gallese and Goldman, 1998). Therefore, to carefully treat and model the behaviors of other agents would be full of potential. For example, in competitive tasks such as chess, the player often sees a number of moves ahead by considering the possible reaction of the other player. In goal-oriented dialogues for a hotel booking task, the agent can reduce interaction numbers and improve user experience by modeling users as business travellers with strict time limit or backpackers seeking adventure.

In this paper, we propose a new dialogue policy learning method with OPPosite agent Awareness (OPPA), where the agent maintains explicit modeling of the opposite agent or user for facilitating its own policy learning. Different from DDQ, the estimated user model is not utilized as a simulator to produce simulated experiences, but as an auxiliary component of the target agent's policy to guide the next action. Figure 1(c) shows the framework of our model. Specifically, whenever the system needs to output an action, it foresees a candidate action $\hat{a}_t$ and consequently estimates the user's response behavior ${a^o_{t+1}}'$. On top of this estimation, as well as the dialogue context, it makes better decisions with a dynamic estimation of the user's strategy. To further regulate the behavior of the system agent, we mitigate the difference between the real system action a_t and the sampled action $\hat{a}_t$ with decay for better robustness and consistency. Without any constraint on the type of agents (either competitive or cooperative), the proposed OPPA method can be applied to both cooperative and non-cooperative goal-oriented dialogues.

To summarize, our contributions are three-fold:

- We propose a new dialogue policy learning setting where the agent shifts from passively learning to actively estimating the opposite agent or user for more efficient simulations, thereby obtaining better performance.
- We mitigate the difference between real system agent action and the sampled action with decay to further enhance estimated system agent behaviors.
- Extensive experiments on both cooperative and competitive goal-oriented dialogues indicate that the proposed model can achieve better dialogue policies than baselines.

2 Related Work

2.1 RL-based Dialogue Policy Learning

Policy learning plays a central role in building goal-oriented dialogue systems by deciding the next action, which is often formulated using the RL framework. Early methods used probabilistic graph model, such as partially observable Markov decision process (POMDP), to learn dialogue policy by modeling the conditional dependences between observation, belief states and actions (Williams and Young, 2007). However, these methods require manual work to define features and state representation, which leads to poor domain adaptation. More recently, deep learning methods are applied in dialogue policy learning, including DQN (Mnih et al., 2015) and Policy Gradient (Sutton et al., 2000) methods, which mitigate the problem of domain adaptation through function approximation and representation learning (Zhao and Eskenazi, 2016).

Recently, there are some efforts focused on multi-domain dialogue policy. An intuitive way is to learn independent policies for each specific domain and aggregate them (Wang et al., 2014; Gašić

et al., 2015; Cuayáhuitl et al., 2016). There are also some works using hierarchical RL, which decomposes the complex task into several sub-tasks (Peng et al., 2017; Casanueva et al., 2018) according to pre-defined domain structure and cross-domain constraints. Nevertheless, most of the above works regard the opposite agent as part of the environment without explicitly modeling its behavior.

Planning based RL methods are also introduced to make a trade-off between reducing human interaction cost and learning a more realistic simulator. Peng et al. (2018) proposed to use Deep Dynamic Q-network, in which a world model is co-trained with the target policy model. By training the world model with the real system-human interaction data, it consistently approaches the performance of real users, which provides better simulated experience for planning. Adversarial methods are applied to dynamically control the proportion of simulated and real experience during different stages of training (Su et al., 2018; Wu et al., 2018). Still, these methods work from the opposite agents' angle.

2.2 Dialogue User Simulation

In RL-based dialogue policy learning methods, a user simulator is often required to provide affordable training environments due to the high cost of collecting real human corpus. Agenda-based simulation (Schatzmann et al., 2007; Li et al., 2016) is a widely applied rule-based method, which starts with a randomly generated user goal that is unknown to the system. During a dialogue session, it remains a stack data structure known as *user agenda*, which holds some pending user intentions to achieve. In the stack update process, machine learning or expert-defined methods can be applied. There are also some model-based methods that learn a simulator from real conversation data. The seq2seq framework has recently been introduced by encoding dialogue history and generates the next response or dialogue action (Asri et al., 2016; Kreyssig et al., 2018). By incorporating a variational step to the seq2seq network, it can introduce meaningful diversity into the simulator (Gür et al., 2018). Our work tackles the problem from a different point of view. We let the target agent approximate an opposite agent model to save user simulation efforts.

3 Model

In this section, we introduce our proposed OPPA model. There are two agents in our framework, one is the system agent we want to optimize, and the other is the user agent. We refer to these two agents as *target* and *opposite* agents in the following sections. Note that the proposed model works at dialog act level, and it can also work at natural language level when equipped with natural language understanding (NLU) and natural language generation (NLG) modules.

3.1 Overview

As shown in Figure 2, the proposed model consists of two key components: a target agent Q-function $Q(s, a)$ and an opposite agent policy estimator $\pi^o(s, a)$. Specifically, each time before the target agent needs to take an action, the model samples a candidate action $\hat{a}_t$. Then the opposite estimator π_o estimates the opposite agent's response behavior ${a^o_{t+1}}'$, which is then aggregated with the original dialog state s_t to generate a new state $\hat{s}_t$. On top of this new state, the target policy $Q(s, a)$ gets the next target action. In more detail, a brief script of our proposed OPPA model is shown in Algorithm 1.

3.2 Opposite Action Estimation

One essential target of the opposite estimator is to measure how the opposite agent reacts given its preceding target agent action and state. In OPPA, we implement the opposite estimation model using a two-layer feed-forward neural network followed by a softmax layer. It takes as input the current state s_t, a sampled target action $\hat{a}_t$, and predicts an opposite action ${a^o_{t+1}}'$ as below:

$$ {a^o_{t+1}}' = \pi^o(s_t, \hat{a}_t). \tag{1} $$

Note that we regard the opposite action estimation task as a classification problem, and ${a^o_{t+1}}'$ is an action label. It has been shown effective in other studies like Su et al. (2018). We also carried out preliminary experiments on other more complicated designs such as Weber et al. (2017). However, results have shown MLP's superior performance in our dialogue policy learning task.

3.3 Opposite Aware Q-Learning

After obtaining the estimated opposite reaction ${a^o_{t+1}}'$, it serves as an extra input to the DQN-based policy component besides the original dialogue state representation s_t. Therefore, we form a new state representation $\hat{s}_t$ as below:

$$ \hat{s}_t = [s_t, E^o {a^o_{t+1}}'], \tag{2} $$

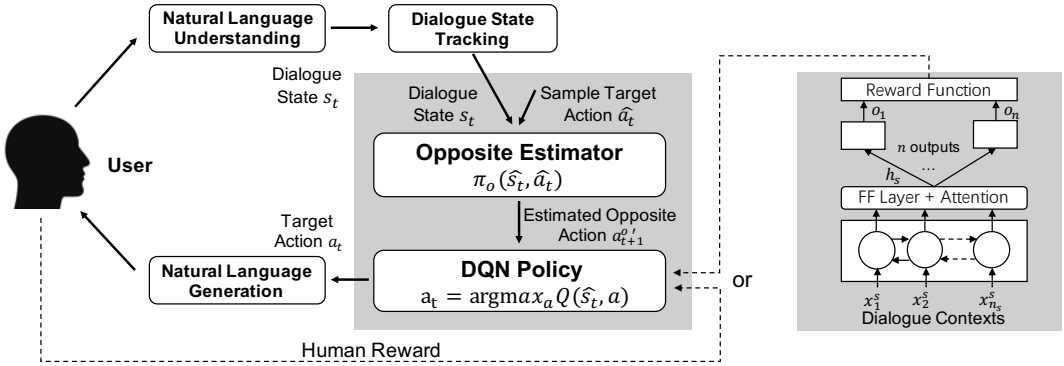

Figure 2: The proposed OPPA model and the reward function. Note that the reward for policy model can be either from real user or the reward function depending on whether real reward is available.

in which $E^o {a^o_{t+1}}'$ introduces the knowledge of opposite agent into our policy learning. E^o is the opposite action embedding matrix which maps the action into specific vector representation. Given the output a_t of $\text{argmax}_{a'} Q(\hat{s}_t, a')$, the agent chooses an action to execute using an ϵ-greedy policy that selects a random action with probability ϵ or otherwise follows the output a_t. We update the Q-function by minimizing the mean-squared loss function, which is defined as

$$\mathcal{L}_1(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}^L}[(y_i - Q(s,a))^2], \quad (3)$$

$$y_i = r + \gamma \max_{a'} Q'(s', a'), \quad (4)$$

where $\gamma \in [0, 1]$ is a discount factor, $\mathcal{D}^L$ is the replay buffer and y_i represents the expected reward computed based on the transition.

3.4 Target Action Sampling

In this subsection, we explain how the action $\hat{a}_t$ is sampled utilizing the above modules. For generating the true target action a_t, we predict it using a deep Q-network which takes as input an estimated opposite action ${a^o_{t+1}}'$ and the dialogue state h^e_t. However, we cannot get ${a^o_{t+1}}'$ without $\hat{a}_t$. Therefore, we further leverage this Q-network at hand. Specifcally, we feed an constant opposite action placeholder a^o to the Q-function:

$$\hat{a}_t = \text{argmax}_{a'} Q([h^e_t, E^o a^o], a') \quad (5)$$

where a^o serves as a constant opposite action. In our experiment, a^o corresponds to the general actions which do not influence business logic, such as *Hello* and *Thanks*.

3.5 Action Regularization with Decay

In our method, $\hat{a}_t$ is sampled from a distribution. At the very beginning of training, since the model is not well trained, the sampled $\hat{a}_t$ may perform badly, which would lead to slow convergence. Therefore, we apply action regularization to mitigate the difference between $\hat{a}_t$ and real a_t. As the training progress goes on, such guidance becomes less effective, and we hope to encourage the model to explore more in the action space. Therefore, we adopt a decay mechanism inspired by (Zhang et al., 2019a). The regularization term is defined as the cross entropy of $\hat{a}_t$ and real action:

$$\mathcal{L}_2(\theta) = -\beta \sum_t a_t log(\hat{a}_t), \quad (6)$$

where β is the decay coefficient. The value of β decreases along with time by applying a discount factor γ in each epoch. As a consequence, a strict constraint on the sampled action is applied to avoid large action sampling performance drop at the beginning stage. After that, the constraint is continuously relaxed so that the model can explore more actions for better strategy.

To sum up, the final loss function for training our OPPA model is the weighted sum of the DQN loss and action regularization loss:

$$\mathcal{L}(\theta) = \mathcal{L}_1(\theta) + \lambda \mathcal{L}_2(\theta). \quad (7)$$

where λ is an adjustable hyperparameter.

3.6 Reward Function

When a dialogue session is completed, what we get are several dialogue acts or natural language utterances (when paired with NLU and NLG). For most goal-oriented dialogues, the reward signal can be obtained from the user simulator or real user ratings. However, when that reward is not available, an output prediction model is required which takes as input the whole dialogue session

Algorithm 1 OPPA for Dialogue Policy Learning

Require: ϵ, C

initialize $\pi^o(s, a; \theta_\pi)$ and $Q(s, a; \theta_Q)$ by supervised and imitation learning
initialize $Q'(s, a, \theta_{Q'})$ with $\theta_{Q'} = \theta_Q$
initialize replay buffer D
for each *iteration* **do**
  $user$ acts a^u
  initialize state s
  while not $done$ **do**
    $e = \text{random}(0, 1)$
    if $e < \epsilon$ **then**
      select a random action a
    else
      sample $\hat{a}_t$
      est. user action ${a^o_{t+1}}' = \pi^o(s, \hat{a}_t)$
      $\hat{s} = [s, E^o {a^o_{t+1}}']$
      $a = \text{argmax}_{a'} Q(\hat{s}, a'; \theta_Q)$
    end if
    execute a
    get user response a^o and reward r
    state updated to s'
    store (s, a, r, s') to D
  end while
  sample minibatches of (s, a, r, s') from D
  update θ_Q according to Equation 4
  each every C iterations set $\theta_{Q'} = \theta_Q$
end for

$X = \{x^s_1, x^s_2, ..., x^s_n\}$ where X is a sequence of tokens, and outputs structured result to calculate the reward.

We use a bi-directional GRU model with an attention mechanism to learn a summarization h^s of the whole session:

$$h^o_j = \text{BiGRU}(h^o_{j-1}, [Ex^s_j, h_j]), \quad (8)$$

$$h^a_j = W^a[\tanh(W^h h^o_j)], \quad (9)$$

$$\alpha_j = \frac{\exp(w \cdot h^a_j)}{\sum_{t'} \exp(w \cdot h^a_{j'})}, \quad (10)$$

$$h^s = \tanh(W^s[h^g, \sum_j \alpha_j h_j]). \quad (11)$$

Note that in this process, we concatenated all the utterances by time order, and the subscript j indicates the index of word in the concatenated sequence. In addition, there may be multiple aspects of the output. For example, in a negotiation goal-oriented dialogue with multiple issues (we denote the book or hat items to negotiate on as issues), we need to get the output of each issue to calculate the total reward. Therefore, for each issue o_i, a specific softmax classifier is applied:

$$p_\theta(o_i | x_{0...T}, g) = \text{softmax}(W^{o_i} h^s). \quad (12)$$

After the structured output is predicted, we can obtain the final reward by applying the task-specific reward function on the output.

$$r = f^R(o_1, o_2, ..., o_{N_o}), \quad (13)$$

where N_o is the number of output aspects and f^R is the reward function which is often manually defined according to the task.

4 Experiment

Depending on the task, dialogues can be divided into cooperative and competitive ones. In a cooperative task, the aim can be reducing unnecessary interactions by inferring the opposite person's intention. While in competitive tasks, the aim is usually to maximize their own interests by considering the opposite agents' possible reactions. To test our method's wide suitability, we evaluated it on both cooperative and competitive tasks.

4.1 Dataset

For the cooperative task, we used MultiWOZ (Budzianowski et al., 2018), a large-scale linguistically rich multi-domain goal-oriented dialogue dataset, which contains 7 domains, 13 intents and 25 slot types. There are 10,483 sessions and 71,544 turns, which is at least one order of magnitude larger than previous annotated task-oriented dialogue dataset. Among all the dialogue sessions, we used 1,000 each for validation and test. Specifically, in the data collection stage, the user follows a specific goal to converse with the agent but is encouraged to change his/her goal dynamically during the session, which makes the dataset more challenging.

For the competitive task, we used a bilateral negotiation dataset (Lewis et al., 2017), where there are 5,808 dialogues from 2,236 scenarios. In each session, there are two people negotiating to divide some items, such as books, hats and balls. Each kind of item is of different value to each person, thus they can give priority to valuable items in the negotiation. For example, a hat may worth 5 for person A and 3 for person B, so B can give up some hat in order to get other valuable items. To conduct our experiment, we further labeled the dataset with system dialogue actions.

4.2 Experimental Settings

We implemented the model using PyTorch (Paszke et al., 2017). The hyper-parameters were decided using validation set. The dimension of GRU_o hidden state is 256, and the hidden state size of GRU_g and GRU_w are 64 and 128 respectively. The size of h_s is 256. As for the Q-function, the size of s_t is 256. ϵ-greedy is applied for exploration. The buffer size of D is set to 500 and the update step C is 1.

Note that due to the complexity of MultiWOZ, the error propagation problem caused by NLU and NLG is serious. Therefore, the cooperative experiment is conducted on the dialogue act level. In the experiment, our proposed model interacts with a robust rule-based user simulator, which appends an agenda-based model (Schatzmann et al., 2007) with extensive manual rules. The simulator gives user response, termination signal and goal-completion feedback during training. For the competitive task, the experiment is on natural language level. Following (Lewis et al., 2017), we built a seq2seq language model for the NLU and NLG module, which is pre-trained on the negotiation corpus.

Our proposed model was first pre-trained with supervised learning (SL). Specifically, we pre-trained the opposite estimator π_o and the Q-function $Q(s, a)$ via supervised learning and imitation learning. We then fine-tuned the model using reinforcement learning (RL). The reward of the MultiWOZ experiment consists of two parts: a) a small negative value in each turn to encourage shorter sessions and b) a large positive reward when the session ends successfully. Note that the task completion signal is obtained from the user. For the negotiation experiment, the reward is the total value of item items that the agent finally got. In the negotiation dataset, the reward is given by the proposed output model described in the Reward Function section.

4.3 Baselines

To demonstrate the effectiveness of our proposed model, we compared it with the following baselines. For the MultiWOZ task, we compared with:

- **DQN**: The conventional DQN (Mnih et al., 2015) algorithm with a 2-layer fully-connected network for Q-function.
- **REINFORCE**: The REINFORCE algorithm (Williams, 1992) with a 2-layer fully-connected policy network.
- **PPO**: Proximal Policy Optimization (Schulman et al., 2017), a policy-based RL algorithm using a constant clipping mechanism.
- **DDQ**: The Deep Dyna-Q (Peng et al., 2018) algorithm which introduced a world-model for RL planning.

Note that the DQN can be seen as our proposed model without opposite estimator (**OPPA w/o OBE**). For the negotiation task, we compared with:

- **SL RNN**: A supervised learning method that is based on an RNN language generation model.
- **RL RNN**: The reinforcement learning extension of SL RNN by refining the model parameters after SL pretraining.
- **ROL**: SL RNN with goal-based decoding in which the model first generates several candidate utterances and chooses the one with the highest expected overall reward after rolling out several sessions.
- **RL ROL**: The extension of RL RNN with rollout decoding.
- **HTG**: A hierarchical text generation model with planning (Yarats and Lewis, 2018), which learns explicit turn-level representation before generating a natural language response.

Note that the rollout mechanism used in ROL and RL ROL also endows them with the ability of "seeing ahead" in which the candidate actions' rewards are predicted using a random search algorithm, while our OPPA explicitly models the opposite's behavior. RL RNN, RL ROL and HTG used the REINFORCE (Williams, 1992) algorithm for reinforcement learning on both strategy and language level, while in OPPA we used the DQN (Mnih et al., 2015) algorithm only on strategy level. To further examine the effectiveness of our proposed action regularization with decay, we did an ablation study by removing the regularization with decay part in Equation 6 (**OPPA w/o A**).

4.4 Evaluation Metric

For the evaluation of experiments on MultiWOZ, we used the number of turns, inform F1 score, match rate and success rate. The **Number of turns** is the averaged number on all sessions, and less turns in cooperative goal-oriented task can promote user satisfaction. **Inform F1** evaluates whether all the slots of an entity requested by the user have been successfully informed. We use F1 score because it considers both the precision and recall so

that a policy which greedily informs all slot information of an entity won't get a high score. **Match rate** evaluates whether the booked entities match the goals in all domains. The score of a domain is 1 only when its entity is successfully booked. Finally, a session is considered **successful** only if all the requested slots are informed (recall = 1) and all entities are correctly booked.

For the negotiation task, we used the averaged scores (total values of items) of all the sessions and those with an agreement as the primary evaluation metrics following Lewis et al. (2017). The percentage of agreed and Pareto optimal* sessions are also reported.

Method	#Turn	Inform F1	Match	Success
DQN	10.50	78.23	60.31	51.7
REINFORCE	9.49	81.73	67.41	58.1
PPO	9.83	83.34	69.09	59.0
DDQ	9.31	81.49	63.10	62.7
OPPA w/o A	8.19	88.45	77.18	75.2
OPPA	8.47	91.68	79.62	81.6
Human	7.37	66.89	95.29	75.0

Table 1: The results on MultiWoZ dataset, a large scale multi-domain task-oriented dialog dataset. We used a rule-based method for DST and Agenda-based user simulator. The DQN method can be regard as OPPA w/o OBE. Human-human performance from the test set serves as the upper bound.

4.5 Cooperative Dialogue Analysis

The results on MultiWOZ dataset are shown in Table 1. OPPA shows superior performance on task success rate than other baseline methods due to the considerable improvement in Inform F1 and Match rate. By first infer the next action of the opposite agent, the target agent policy can make better choices to match the reward signal during training. When compared with human performance, OPPA even achieves a higher success rate, although the number of turns is still higher. This might be due to the fact that the user is sensitive to the dialogue length. When a dialogue becomes intolerably long, many user will leave without completing the dialogue. By taking actions in account of the inferred opposite action, the target agent can also make the dialogue more efficiently by avoiding some lengthy interactions, which is extremely important in applications where the user is sensitive to dialogue length.

Meanwhile, DDQ achieves higher task success rate than other baseline models since it also models the behavior of opposite agent through world model. However, it makes use of the learned world model by providing more simulated experiences, which does not give a direct hint on how to act in the middle of a session. Therefore, in its experiments, it still gets longer dialogue sessions and a lower success rate than OPPA.

If we remove the action regularization mechanism, we can see an obvious decline on performance, which is as expected. The action regularization is introduced to mitigate the difference between sampled $\hat{a}_t$ and real a_t, so there can be a large discrepancy between the sampled and real actions if we remove it at the early training stage. When the $\hat{a}_t$ is not reliable, the consequent estimated opposite action a'_{t+1} also becomes noisy, which leads to performance drop.

4.6 Competitive Dialogue Analysis

Table 2 shows the scores for all sessions and for only agreed ones. When comparing with the seq2seq models, OPPA achieves significantly better results. This can be attributed to the hierarchical structure of OPPA. The sequence models only take as input (and outputs) the word-level natural language utterances, without explicitly modeling turn-level dialogue actions. In this way, the parameters for linguistic and strategy functions are tangled together, and the back-propagation errors can influence both sides. As for the two ROL models, although they can predict the value of a candidate action in advance, they still cannot beat OPPA. The reason is that the rollout method did not explicitly maintain an estimation of the opposite agent as our OPPA did. Instead, it just estimates the candidate acitons' rewards based on Monte Carlo search by using its own model for predicting future movements. Therefore, when the opposite model's behavior is not very familiar to the target agent, the estimated reward becomes unreliable.

The HTG model also used a hierarchical framework by learning an explicit turn-level latent representation. By doing this, it obtains higher scores than the seq2seq models. However, it does not make any assumptions about the opposite agent. Therefore, its scores are still lower than OPPA, although the discrepancy narrows down.

By removing the opposite estimator, we find that the performance of OPPA w/o OBE drops significantly compared to that of OPPA. This ablation study directly verifies the effectiveness of our proposed opposite behavior estimator. There fore,

*A dialogue is Pareto optimal if neither agent's score can be improved without lowering the other's score.

Method	vs. SL RNN		vs. RL RNN		vs. ROL		vs. RL ROL	
	All	Agreed	All	Agreed	All	Agreed	All	Agreed
SL RNN	5.4 vs. 5.5	6.2 vs. 6.2	-	-	-	-	-	-
RL RNN	7.1 vs. 4.2	7.9 vs. 4.7	5.5 vs. 5.6	5.9 vs. 5.8	-	-	-	-
ROL	7.3 vs. 5.1	7.9 vs. 5.5	5.7 vs. 5.2	6.2 vs. 5.6	5.5 vs. 5.4	5.8 vs. 5.9	-	-
RL ROL	8.3 vs. 4.2	8.8 vs. 4.5	5.8 vs. 5.0	6.5 vs. 5.5	6.2 vs. 4.9	7.0 vs. 5.4	5.9 vs. 5.8	6.4 vs. 6.3
HTG	8.7 vs. 4.4	8.8 vs 4.5	6.0 vs. 5.1	6.9 vs. 5.5	6.5 vs. 5.0	6.9 vs. 5.3	6.5 vs. 5.6	7.0 vs. 6.3
OPPA w/o OE	8.2 vs. 4.2	8.8 vs. 4.7	6.1 vs. 5.2	6.8 vs. 5.6	6.5 v.s. 4.8	7.0 v.s. 5.3	5.7 v.s. 5.8	6.5 v.s. 6.4
OPPA w/o A	8.7 vs. 4.1	8.9 vs. 4.3	6.3 vs. 5.0	7.2 vs. 5.4	6.5 vs. 4.8	7.2 vs. 5.4	6.5 vs. 6.1	7.1 vs. 6.8
OPPA	8.8 vs. 3.9	9.0 vs. 4.1	6.7 vs. 4.6	7.3 vs. 5.2	6.8 vs. 4.2	7.4 vs. 5.1	6.7 vs. 6.0	7.2 vs. 6.6

Table 2: The results of our proposed OPPA and the baselines on the negotiation dataset. *All* and *Agreed* indicates averaged scores for all sessions and only the agreed sessions respectively.

Method	vs. SL RNN		vs. RL RNN		vs. ROL		vs. RL ROL	
	Agreed(%)	PO(%)	Agreed(%)	PO(%)	Agreed(%)	PO(%)	Agreed(%)	PO(%)
SL RNN	87.9	49.6	-	-	-	-	-	-
RL RNN	89.9	58.6	81.5	60.3	-	-	-	-
ROL	92.9	63.7	87.4	65.0	85.1	67.3	-	-
RL ROL	94.4	74.8	85.7	74.6	71.2	76.4	67.5	77.2
HTG	94.8	75.1	88.3	75.4	83.2	77.8	66.1	73.2
OPPA w/o OBE	94.6	74.6	87.9	75.2	79.3	78.2	73.7	77.9
OPPA w/o A	95.6	77.9	91.9	77.4	82.4	78.8	78.0	79.5
OPPA	95.7	77.7	91.4	77.2	82.3	79.1	78.2	79.7

Table 3: The proportion of agreed and Pareto optimal (PO) sessions for our proposed OPPA and the baselines on the negotiation dataset.

modeling the opposite policy in one's mind is a crucial source to achieve better results in competitive dialogue policy learning.

When comparing with OPPA w/o A which removed action regularization, we can see that the OPPA model gets better results. This verifies the importance of regularizing the action sampling. By controlling the difference between real and model generated actions, we can keep the opposite model consistent with the real opposite agent at the early training stage.

The percentage of agreed and Pareto optimal session are shown in Table 3. As we can see, the percentage of Pareto optimal increases in our method, showing that the OPPA model can explore the solution space more effectively. However, the agreement rate decreases when the opposite model gets stronger. This phenomenon is also found in Lewis et al. (2017) when they change the opposite agent from SL RNN to real human. This can be attributed to the aggressiveness of the agent: when both agents act aggressively, they are less likely to reach an agreement. The SL RNN model simply imitates the behavior in the dialogue corpus, while the ROL and RL mechanisms both help the agent to explore more spaces, which makes them more aggressive on action selection.

4.7 Human Evaluation

To better validate our propositions, we further conducted human evaluation by making our model conversing with real user. We only conducted human evaluation on the negotiation task since the MultiWOZ model is implemented on the dialogue act level. We tested the models on a total of 1,000 dialogue sessions. In the evaluation, the users conversed with the agent, and the total item values are used as the evaluation metrics. The results are shown in Table 4. We can see that our proposed OPPA outperforms the baseline models. The system score are lower than that in Table 2, and the discrepancy between *All* and *Agreed* results is large. This can be due to the high intelligence and aggressiveness of real humans who want to get as more values as possible and do not make compromises easily. Due to this reason, the sessions become considerably lengthy, and the target agent exceeds our length limit before reaching an agreement.

Method	All	Agreed
RL ROL	4.5 vs. 5.2	7.8 vs. 7.1
HTG	4.8 vs. 4.7	8.0 vs. 7.2
OPPA w/o A	4.7 vs. 4.9	8.4 vs. 6.7
OPPA	5.2 vs. 5.1	8.2 vs. 6.5

Table 4: The rewards of each model vs. human user.

5 Conclusion

In this work, we present an opposite agent-aware dialogue policy model which actively estimates the

opposite agent instead of doing passive learning from experiences. We have shown that it is possible to harvest a reliable model of the opposite agent through more efficient dialogue interactions. By incorporating the estimated model output as part of dialogue state, the target agent shows significant improvement on both cooperative and competitive goal-oriented tasks. As future work, we will explore multi-party dialogue modeling in which multi-agent learning techniques can be applied.

Acknowledgments

This work was jointly supported by the NSFC projects (key project with No. 61936010 and regular project with No. 61876096), and the Guoqiang Institute of Tsinghua University with Grant No. 2019GQG1. This work was also supported by Beijing Academy of Artificial Intelligence, BAAI and Beijing Nova Program (Z201100006820068) from Beijing Municipal Science & Technology Commission. We thank THUNUS NExT Joint-Lab for the support.

References

Layla El Asri, Jing He, and Kaheer Suleman. 2016. A sequence-to-sequence model for user simulation in spoken dialogue systems. *arXiv preprint arXiv:1607.00070*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Inigo Casanueva, Paweł Budzianowski, Pei-Hao Su, Stefan Ultes, Lina Rojas-Barahona, Bo-Hsiang Tseng, and Milica Gašić. 2018. Feudal reinforcement learning for dialogue management in large domains. *arXiv preprint arXiv:1803.03232*.

Heriberto Cuayáhuitl, Seunghak Yu, Ashley Williamson, and Jacob Carse. 2016. Deep reinforcement learning for multi-domain dialogue systems. *arXiv preprint arXiv:1611.08675*.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2016. Towards end-to-end reinforcement learning of dialogue agents for information access. *arXiv preprint arXiv:1609.00777*.

Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. *arXiv preprint arXiv:1606.03152*.

Vittorio Gallese and Alvin Goldman. 1998. Mirror neurons and the simulation theory of mind-reading. *Trends in cognitive sciences*, 2:493–501.

Jianfeng Gao, Michel Galley, Lihong Li, et al. 2019. Neural approaches to conversational ai. *Foundations and Trends® in Information Retrieval*, 13(2-3):127–298.

M Gašić, N Mrkšić, Pei-hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2015. Policy committee for adaptation in multi-domain spoken dialogue systems. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 806–812. IEEE.

Robert M Gordon. 1986. Folk psychology as simulation. *Mind & Language*, 1(2):158–171.

Izzeddin Gür, Dilek Hakkani-Tür, Gokhan Tür, and Pararth Shah. 2018. User modeling for task oriented dialogues. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 900–906. IEEE.

Mary Harper. 2014. Learning from 26 languages: Program management and science in the babel program. In *COLING*, page 1.

Megha Jhunjhunwala, Caleb Bryant, and Pararth Shah. 2020. Multi-action dialog policy learning with interactive human teaching. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 290–296, 1st virtual meeting. Association for Computational Linguistics.

Florian Kreyssig, Inigo Casanueva, Pawel Budzianowski, and Milica Gasic. 2018. Neural user simulation for corpus-based policy optimisation for spoken dialogue systems. *arXiv preprint arXiv:1805.06966*.

Oliver Lemon and Olivier Pietquin. 2012. *Data-driven methods for adaptive spoken dialogue systems: Computational learning for conversational interfaces*. Springer Science & Business Media.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 1997. Learning dialogue strategies within the markov decision process framework. In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 72–79. IEEE.

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning of negotiation dialogues. In *EMNLP*.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. *arXiv preprint arXiv:1703.01008*.

Xiujun Li, Zachary C Lipton, Bhuwan Dhingra, Lihong Li, Jianfeng Gao, and Yun-Nung Chen. 2016. A user simulator for task-completion dialogues. *arXiv preprint arXiv:1612.05688*.

Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and Li Deng. 2018. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *AAAI*.

Bing Liu and Ian Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 482–489. IEEE.

Bing Liu, Gokhan Tur, Dilek Hakkani-Tur, Pararth Shah, and Larry Heck. 2018. Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. *arXiv preprint arXiv:1804.06512*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS 2017 Workshop Autodiff*.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Kam-Fai Wong. 2018. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. In *ACL*.

Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *EMNLP*.

David Premack and Guy Woodruff. 1978. Does the chimpanzee have a theory of mind? *Behavionral Linguistics*, page 1.

Verena Rieser and Oliver Lemon. 2011. *Reinforcement learning for adaptive dialogue systems: a data-driven methodology for dialogue management and natural language generation*. Springer Science & Business Media.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *NAACL*, pages 149–152.

Jost Schatzmann, Karl Weilhammer, Matt Stuttle, and Steve Young. 2006. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The knowledge engineering review*, 21(2):97–126.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484.

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Continuously learning neural dialogue management. *arXiv preprint arXiv:1606.02689*.

Shang-Yu Su, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Yun-Nung Chen. 2018. Discriminative deep dyna-q: Robust planning for dialogue policy learning. In *EMNLP*.

Richard S Sutton. 1990. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, pages 216–224. Elsevier.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*.

Ryuichi Takanobu, Runze Liang, and Minlie Huang. 2020. Multi-agent task-oriented dialog policy learning with role-aware reward decomposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 625–638, Online. Association for Computational Linguistics.

Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. 2019. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. *arXiv preprint arXiv:1908.10719*.

Zhuoran Wang, Hongliang Chen, Guanchun Wang, Hao Tian, Hua Wu, and Haifeng Wang. 2014. Policy learning for domain selection in an extensible multi-domain spoken dialogue system. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 57–67.

Théophane Weber, Sébastien Racanière, David P Reichert, Lars Buesing, Arthur Guez, Danilo Jimenez Rezende, Adria Puigdomenech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, et al. 2017. Imagination-augmented agents for deep reinforcement learning. *arXiv preprint arXiv:1707.06203*.

Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv preprint arXiv:1702.03274*.

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

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

Yuexin Wu, Xiujun Li, Jingjing Liu, Jianfeng Gao, and Yiming Yang. 2018. Switch-based active deep dyna-q: Efficient adaptive planning for task-completion dialogue policy learning. *arXiv preprint arXiv:1811.07550*.

Denis Yarats and Mike Lewis. 2018. Hierarchical text generation and planning for strategic dialogue. In *ICML*, pages 5587–5595.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019a. Bridging the gap between training and inference for neural machine translation. In *ACL*, pages 4334–4343.

Zhirui Zhang, Xiujun Li, Jianfeng Gao, and Enhong Chen. 2019b. Budgeted policy learning for task-oriented dialogue systems. *arXiv preprint arXiv:1906.00499*.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*.