Meta Dialogue Policy Learning

Yumo Xu*
School of Informatics
University of Edinburgh, UK
yumo.xu@ed.ac.uk

Chenguang Zhu
Microsoft Cognitive Services Research
Redmond, WA, USA
chezhu@microsoft.com

Baolin Peng
Microsoft Research
Redmond, WA, USA
bapeng@microsoft.com

Michael Zeng
Microsoft Cognitive Services Research
Redmond, WA, USA
nzeng@microsoft.com

Abstract

Dialog policy determines the next-step actions for agents and hence is central to a dialogue system. However, when migrated to novel domains with little data, a policy model can fail to adapt due to insufficient interactions with the new environment. We propose Deep Transferable Q-Network (DTQN) to utilize shareable low-level signals between domains, such as dialogue acts and slots. We decompose the state and action representation space into feature subspaces corresponding to these low-level components to facilitate cross-domain knowledge transfer. Furthermore, we embed DTQN in a meta-learning framework and introduce Meta-DTQN with a dual-replay mechanism to enable effective off-policy training and adaptation. In experiments, our model outperforms baseline models in terms of both success rate and dialogue efficiency on the multi-domain dialogue dataset MultiWOZ 2.0.

1 Introduction

Task-oriented dialogue systems aim to assist users to efficiently accomplish daily tasks such as booking a hotel or reserving dinner at a restaurant. Complex systems like Alexa and Siri often contain thousands of task domains. However, a successful model on one task often requires hundreds or thousands of carefully labelled domain-specific dialogue data, which consumes a large amount of human effort. Therefore, how to agilely adapt an existing dialogue system to new domains with a scant number of training samples is an essential task in task-oriented dialogues.

In this paper, we investigate dialogue policy, or dialogue management, which lies in the center of a task-oriented dialogue system. Dialogue policy determines the next-step action of the agent given dialogue states and the user's goals. As a dialogue is composed of multiple turns, the feedback to a dialogue policy's decision is often delayed until the end of the conversation. Therefore, Reinforcement Learning (RL) is usually leveraged to improve the efficiency and success rate in dialogue policy learning [1].

There have been a number of methods applying dialogue policy in multi-domain settings [2–4]. These models usually employ an all-in-one multi-hot representation for dialogue states. The state embedding vector is a concatenation of multiple segments, each as a multi-hot vector for the states in one domain. However, when there are unseen domains at inference time, the corresponding parameters of its dialogue acts and slots are not optimized. This significantly limits the adaptation performance of policy models.

*Work done during an internship at Microsoft.

Preprint. Under review.

To alleviate this problem, we note that there is often shareable low-level information between different domains. For instance, suppose the source domain is taxi-booking and the target domain is hotel-booking. Although the two domains have different ontologies, both domains share certain dialogue slots (e.g. *start time* and *location*) and dialogue acts (e.g. *request* and *inform*). These shared concepts bear a lot of similarities both in textual representation and corresponding agent policies. Thus, it is feasible to transfer domain knowledge via these commonalities in ontologies.

To this end, we propose a **D**eep **T**ransferable **Q**-**N**etwork (DTQN), based on Deep Q-Network (DQN) [5] in reinforcement learning, which learns to predict accurate Q-function values given dialogue states and system actions. In DTQN, we factorize the dialogue state space into a set of lower-level feature spaces. Specifically, we hierarchically model cross-domain relations at domain-level, act-level and slot-level. State representations are then composed of several shareable sub-embeddings. For instance, slots like *start time* in different domains will now share the same slot-level embedding. Furthermore, instead of treating actions as independent regression classes as in DQN, we decompose the dialogue action space and our model learns to represent actions based on common knowledge between domains.

To adapt DTQN to few-shot learning scenarios, we leverage the meta-learning framework. Meta-learning aims to guide the model to rapidly learn knowledge from new environments with only a few labelled samples [6, 7]. Previously, meta-learning has been successfully employed in the Natural Language Generation (NLG) module in dialogues [8]. However, NLG is supervised learning by nature. Comparatively, there has been little work on applying meta-learning to the dialogue policy, as it is known that applying RL under meta-learning, a.k.a. meta-RL, is a much harder problem than meta supervised learning [9].

To verify this fact, we train the DTQN model under the Model-Agnostic Meta-Learning (MAML) framework [6]. However, we find through experiments that the canonical MAML fails to let the policy model converge because the task training phase leverages *off-policy* learning while the task evaluation and meta-adaptation phase employ an *on-policy* strategy. Thus, the model initially receives very sparse reward signals, especially on complex composite-domain tasks. As a result, the dialogue agent is prone to overfit the on-policy data and to get stuck at the local minimum in the policy space.

Therefore, we further propose Meta-DTQN with a *dual-replay* mechanism. To support effective off-policy learning in meta dialogue policy optimization, we construct a task evaluation memory to cache dialogue trajectories and prefill it with rule-based experiences in task evaluation. This dual-replay strategy ensures the consistency of off-policy strategy in both meta-training and meta-adaptation, and provides richer dialogue trajectory records to enhance the quality of the learned policy model. Empirical results show that the dual-replay mechanism can effectively increase the success rate of DTQN while reducing the dialogue length, and Meta-DTQN with dual replay outperforms strong baselines on the multi-domain task-oriented dialogue dataset MultiWOZ 2.0 [10].

2 Related Work

Dialogue Policy Learning Dialogue policy, also known as the dialogue manager, is the controlling module in task-oriented dialogue that determines the agent's next action. Early work on dialogue policy is constructed on manual rules [11]. As the outcome of a dialogue does not emerge until the end of the conversation, dialogue policy is often trained via Reinforcement Learning (RL) [12]. For instance, deep RL is proven useful for strategic conversations [13], and a sample-efficient online RL algorithm is proposed to learn from only a few hundred dialogues [14]. Towards more effective completion on complex tasks, hierarchical RL is employed to learn a multi-level policy either through temporal control [2], or subgoal discovery [15]. Model-based RL also helps a dialogue agent to plan for the future during conversations [1]. While RL for multi-domain dialogue policy learning has attracted increasing attention from researchers, dialogue policy transfer remains under-studied.

Meta-Learning Meta-learning is a framework to adapt models to new tasks with a small number of data [16]. It can be achieved either by finding an effective prior as initialization for new task learning [16], or by a meta-learner to optimize the model which can quickly adapt to new domains [17]. Particularly, the model-agnostic meta-learning (MAML) [6] framework applies to any optimizable system. It associates the model's performance to its adaptability to new systems, so that the resulting model can achieve maximal improvement on new tasks after a small number of updates.

In dialogue systems, meta-learning has been applied to response generation. The domain adaptive dialog generation method (DAML) [8] is an end-to-end dialogue system that can adapt to new domains with a few training samples. It places the state encoder and response generator into the MAML framework to learn general features across multiple tasks.

3 Problem Formulation

Reinforced Dialogue Agent Task-oriented dialogue management is usually formulated as a Markov Decision Process (MDP): a dialogue agent interacts with a user with sequential actions based on the observed dialogue states s to fulfill the target conversational goal. At step t, given the current state s_t of the dialogue, the agent selects a system action a_t based on its policy π, i.e., $a_t = \pi(s_t)$, and receives a reward r_t from the environment[2]. The expected total reward of taking action a under the state s is defined as a function $Q(s, a)$:

$$Q(s,a) = \mathbb{E}_\pi \left[\sum_{k=0}^{T-t} \gamma^k r_{t+k} | s_t = s, a_t = a \right] \tag{1}$$

where T is the maximum number of turns in the dialogue, and $\gamma \in [0, 1]$ is a discount factor. The policy π is trained to find the optimal Q-function $Q^*(s, a)$ so that the expected total reward at each state is maximized. The optimal policy is to greedily act as $\pi^*(s) = \text{argmax}_{a \in \mathcal{A}} Q^*(s, a)$.

To better explore the action space, an ϵ-greedy policy is employed to select the action based on the state s: with probability ϵ, a random action is chosen; with probability $1 - \epsilon$, a greedy policy $a = \text{argmax}_{a'} Q^*(s, a'; \theta_Q)$ is taken. Here, the Q-function is modeled by Deep Q-Network (DQN) [18] with parameters θ_Q. To train this network, state-action transitions (s_t, a_t, r_t, s_{t+1}) are stored in a *replay buffer* $\mathcal{M}$. At each training step, a batch of samples is sampled from $\mathcal{M}$ to update the policy network via 1-step temporal difference (TD) error implemented with the mean-square error loss:

$$\mathcal{L}(\theta_Q) = \mathbb{E}_{(s,a,r,s') \sim \mathcal{M}} \left[(y - Q(s, a; \theta_Q))^2 \right] \tag{2}$$

$$y = r + \gamma \text{max}_{a'} Q'(s', a'; \theta_{Q'}) \tag{3}$$

where Q' is the *target network* that is only periodically replaced by Q to stabilize training.

Environment and Domain The dialogue environment typically includes a database that can be queried by the system, and a *user-simulator* that mimics human actions to interact with the agent. At the beginning of a conversation, the user-simulator specifies a dialogue *goal*, and the agent is optimized to accomplish it. Dialogue goals are generated from one or multiple *domain(s)*. For instance, in the benchmark multi-domain dialogue dataset MultiWoz [10], there are a total of 7 domains and 25 domain compositions. Each domain composition consists of one or more domains, e.g., `{hotel}` and `{hotel, restaurant, taxi}`. We split all domains into *source* domains and *target* domains to fit the meta-learning scenario (see Section 5 for details).

State Representation We show the dialogue state representation for classic DQN in Figure 1(A). After receiving a system action a, the environment responds with a user action, which is then fed into a *dialogue state tracker* (DST) to update the dialogue agenda. The DST maintains the entire dialogue records with a state dictionary, and the DQN has a *state encoder* to embed the dictionary into a state vector. In detail, this state encoder represents states with multi-hot state vectors including six primary feature categories [4], e.g., *request* and *inform*. As shown in the bottom-left corner of Figure 1(A), each category is encoded as the concatenation of a few domain-specific multi-hot vectors from its relevant domains, and the concatenation of the six category representations forms a binary state representation (see Appendix A for details).

We argue that two major issues in the classic DQN system prohibit its generalization to unseen domains: (1) the input states adopt multi-hot representations where no inter-state relation is considered and (2) given the state input, actions in different domains are modeled as independent regression classes. However, there is a considerable amount of domain knowledge that can be shared across

[2]Reward r measures the degree of success of a dialogue. In ConvLab [4], for example, success leads to a reward of $2 * L$ where L is the maximum number of turns in a dialogue (set to 40 in default), failure to a reward of $-L$. To encourage shorter dialogues, the agent also receives a reward of -1 at each turn.

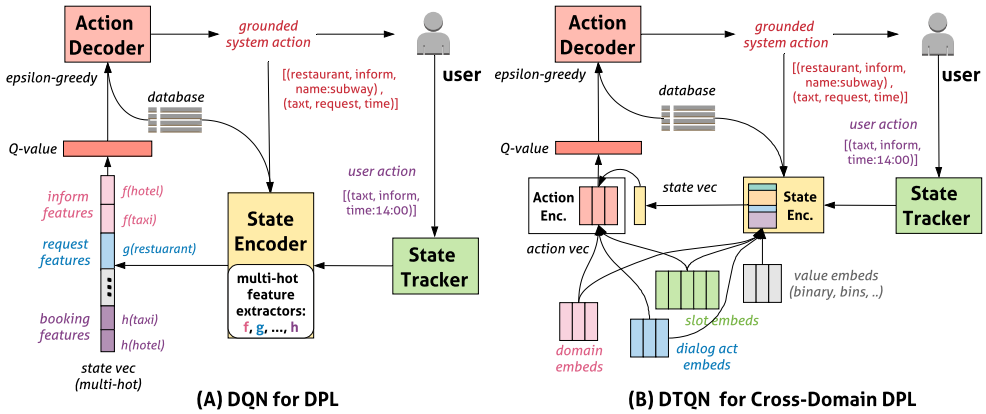

Figure 1: Framework of (A) classic DQN for Dialogue Policy Learning and (B) our Deep Transferable Q-Network (DTQN) for Cross-Domain Dialogue Policy Learning.

actions and states, e.g., both taxi-booking and hotel-reserving tasks share dialogue slots such as *start time* and *location* and dialogue acts such as *request*. These types of information elicit similar text representation and policy handling.

4 Framework

4.1 Deep Transferable Q-Network

To enable effective knowledge learning and transfer across different domains, we reformulate cross-domain dialogue policy learning as a state-action matching problem. As shown in in Figure 1(B), we propose DTQN, a **D**eep **T**ransferable **Q**-**N**etwork that jointly optimizes the policy network and domain knowledge representations.

Driven by the structure of dialogue knowledge, we assume that the dialogue state space $\mathcal{S}$ and the system action space $\mathcal{A}$ are factorized by a set of lower-level feature spaces. Based on this hypothesis, we aim at modeling cross-domain relations at different levels in DTQN: domain-level, act-level and slot-level. To this end, we hierarchically decompose the state and actions into four embedding subspaces, shared across all dialogue sessions: domains $\mathcal{D}$, dialogue acts $\mathcal{C}$, slots $\mathcal{O}$, and values $\mathcal{V}$. Both the states and actions are encoded by joining different sets of subspace embeddings.

We retain the existing categorization of dialogue state features mentioned in Section 3 considering its effectiveness in dialogue management [1, 4]. We first represent each feature category as a dense vector $s_h, h \in [1, H]$ and then concatenate the $H = 6$ categories of feature vectors into the state representation $s \in \mathbb{R}^{d_s \times 1}$:

$$s = \text{ReLU}\left(W_s[s_1, s_2, ..., s_H]\right). \tag{4}$$

Note that each $s_h, h \in [1, H]$ consists of $|\mathcal{D}_h|$ domain-specific features, each corresponding to a domain. In the classic DQN state representation [4], few features are shared across domains. As a result, an agent cannot generalize its policy from source domains to a target domain if the target state space remains unseen and the target action space is mostly unexplored. Besides, the length of state representations grows linearly with $|\mathcal{D}_h|$.

Here, we propose to use a fixed-length state vector s_h to represent the h-th feature category. To do that, we use cross-domain features to aggregate state information from different domains. In detail, the i-th domain-specific component of s_h is denoted by $\hat{s}_{h,i}$. For example, for the *inform* category,

$$\hat{s}_{h,i} = [d_{h,i}, \overline{o \odot v}, u_{h,i}] \tag{5}$$

where $d_{h,i}$ denotes embedding of domain. $\overline{o \odot v}$ is the average of the inner product between general slot embeddings and their value embeddings. The binary feature $u_{h,i}$ tracks whether the corresponding domain is active, i.e., essential domain slots are already filled.

Algorithm 1 Meta Dialogue Policy Learning

function METAPOLICYLEARNING
  Initialize $Q(s, a; \theta_Q)$ and $Q'(s, a; \theta_{Q'})$ with $\theta_{Q'} \leftarrow \theta_Q$ ▷ *Policy network and target network*
  Initialize experience replay memory $\mathcal{M}_{tr}$ and $\mathcal{M}_{ev}$ using Reply Buffer Spiking (RBS) ▷ *Dual replay*
  Set K domains $\{d_k\}_{k=1}^K$ and gather the domain compositions $\{\mathcal{T}_{d_k}\}_{k=1}^K$, where $d_k \in \mathcal{T}_{d_k}, 1 \leq k \leq K$.
  for $n \leftarrow 1 : N$ **do** ▷ *Outer loop for meta-training*
    Generate K dialogue goals $\{t_1, ..., t_K\}$ from d_k or Uniform($\mathcal{T}_{d_k}$) ▷ *Single or composite domain*
    Initialize meta-training loss $\mathcal{L} \leftarrow 0$
    for $k \leftarrow 1 : K$ **do** ▷ *Inner loop for task data collection and training*
      $\theta' \leftarrow \theta$ and load agent with θ'
      ENVINTERACT(t_k, $\mathcal{M}_{tr}$, B_{tr}) ▷ *Task training data collection*
      Sample random minibatches of (t_k, s, a, r, s') from $\mathcal{M}_{tr}$
      Update θ' via Z-step minibatch SGD
      ENVINTERACT(t_k, $\mathcal{M}_{ev}$, B_{ev}) ▷ *Task evaluation data collection*
      Sample random minibatches of (t_k, s, a, r, s') from $\mathcal{M}_{ev}$
      Forward pass with the minibatches and obtain $\mathcal{L}_{t_k}$
      $\mathcal{L} \leftarrow \mathcal{L} + \mathcal{L}_{t_k}$
    end for
    Load agent with θ_Q and update with respect to $\mathcal{L}$ via minibatch SGD
    Every C steps reset $\theta_{Q'} \leftarrow \theta_Q$ ▷ *Target network update*
  end for
end function

To obtain the fixed-length representation for a feature category, we aggregate its domain features from all relevant domains via a non-linear transformation with a residual connection:

$$s_h = \frac{1}{|\mathcal{D}_h|} \sum_{i=1}^{|\mathcal{D}_h|} \left(\hat{s}_{h,i} + \text{ReLU}\left(W_h \hat{s}_{h,i} + b_h\right)\right) \tag{6}$$

where W_h projects domain-specific features into a shared feature space across domains and we acquire the final state representation s via Equation (4).

Different from DQN, which encodes only dialogue states and incorporates no prior information of actions, we explicitly model the structural information of system actions with an *action encoder* in DTQN to maximize knowledge sharing across domains. Action encoding follows analogous procedures to the state encoding except that it does not use value space $\mathcal{V}$. For each system action a, the domains that contain this action form a set $\mathcal{D}_a$. We encode its ℓ-th domain feature ($1 \leq \ell \leq |\mathcal{D}_a|$) as $\hat{a}_\ell = [d_\ell, c_\ell, \overline{o}]$, where c_ℓ is embedding for dialogue act, e.g., *request* or *booking*, and $\overline{o}$ is the average of slot embeddings. We then obtain the system action embedding: $a = 1/|\mathcal{D}_a| \sum_{\ell=1}^{|\mathcal{D}_a|} \left(\hat{a}_\ell + \text{ReLU}\left(W_a \hat{a}_\ell + b_a\right)\right)$.

All embedding tables are shared between state and action encoders. We stack all action vectors and denote the action matrix as $A \in \mathbb{R}^{|\mathcal{A}| \times d_a}$, which is then used to produce the Q-values:

$$Q(s, a) = \frac{1}{\sqrt{d_a}} A W_q s \in \mathbb{R}^{|\mathcal{A}|} \tag{7}$$

where $W_q \in \mathbb{R}^{d_a \times d_s}$ is a parameter matrix.

4.2 Meta Reinforcement Learning with Dual Replay

To adapt Q-network to few-shot learning scenarios, we propose to use a meta-learning framework [6] and present an instantiation of this framework with the DTQN as the policy network Q and target network Q'. Algorithm 1 shows the pseudocode for our methodology for meta dialogue policy learning.

At the beginning of each outer-loop of meta-training, we first sample K dialogue goals as training tasks. For the kth inner-loop step, the agent interacts with the environment using task t_k to collect trajectories and stores them in the replay buffer $\mathcal{M}_{tr}$ (see Appendix B for details of function ENVINTERACT). Then, we sample from $\mathcal{M}_{tr}$ a minibatch of experiences of task t_k: $\mathcal{B}_{tr}^{t_k}$. The loss

Systems	Hotel			Train			Police			Average		
	Success	Reward	Turns	Success	Reward	Turns	Success	Reward	Turns	Success	Reward	Turns
Few-Shot Models												
DQN-1K	0.00	-55.70	17.71	2.15	-56.02	20.60	**100.00**	76.62	5.38	14.66	8.58	13.68
DTQN-1K	53.90	15.06	11.62	61.00	24.84	10.36	**100.00**	**79.28**	**2.72**	71.63	39.73	8.23
Adaptive Models												
VANILLADQN	28.10	-20.18	15.89	0.00	-59.00	21.00	24.30	-25.95	17.11	17.47	-35.04	18.00
DQN	36.00	-9.70	14.90	32.20	-14.53	15.17	**100.00**	76.62	5.38	56.07	17.46	11.82
DTQN	**62.70**	**27.99**	**9.25**	82.65	54.80	6.38	**100.00**	79.24	2.76	81.78	54.01	6.13
META-DTQN-SR	47.50	5.86	13.14	50.35	10.26	12.16	**100.00**	**79.28**	**2.72**	65.95	31.80	9.34
META-DTQN	61.90	26.28	10.00	**87.45**	**61.44**	**5.50**	**100.00**	79.24	2.76	**83.12**	**55.65**	**6.09**

Table 1: System performance in the single-domain setting on 2000 dialogues in the target domains.

function $\mathcal{L}_{t_k}$ is from Equation (2). We compute task-specific updated parameters $\theta_Q^{(k)}$ from θ_Q:

$$\theta_Q^{(k)} = \theta_Q - \alpha \nabla_{\theta_Q} \mathcal{L}_{t_k} \left(\theta_Q; \mathcal{B}_{tr}^{t_k} \sim \mathcal{M}_{tr}\right) \text{ where} \tag{8}$$

$$\nabla_{\theta_Q} \mathcal{L}_{t_k} \left(\theta_Q\right) = \mathbb{E}_{(s,a,r,s') \sim \mathcal{B}_{tr}^{t_k}} \left[\left(r + \gamma \max_{a'} Q'\left(s', a'; \theta_{Q'}\right) - Q\left(s, a; \theta_Q\right)\right) \nabla_{\theta_Q} Q\left(s, a; \theta_Q\right) \right]. \tag{9}$$

With the updated parameters $\theta_Q^{(k)}$, the agent interacts with the environment and obtains trajectory $\mathcal{B}_{ev}^{t_k}$. According to MAML [6], the task evaluation loss $\mathcal{L}_{t_k}(\theta_Q^{(k)}; \mathcal{B}_{ev}^{t_k})$ should be directly used to update θ_Q with learning rate β:

$$\theta_Q \leftarrow \theta_Q - \beta \nabla_{\theta_Q} \sum_{k=1}^{K} \mathcal{L}_{t_k}(\theta_Q^{(k)}; \mathcal{B}_{ev}^{t_k}). \tag{10}$$

However, this *on-policy* learning suffers from very sparse rewards especially at the initial learning stage. This is due to the inherent difficulties in cross-domain dialogue learning: i) the state-action space to explore is much larger, and ii) the conversation required to complete the task is often longer [2]. As a result, the dialogue agent is prone to overfit with on-policy data and to get stuck at the local minimum in the policy space.

To alleviate this problem, we propose a dual-replay framework to support efficient *off-policy* learning in meta-RL. Apart from the main replay buffer $\mathcal{M}_{tr}$ for meta-training, we construct a task evaluation memory $\mathcal{M}_{ev}$. We note that it is essential to separate $\mathcal{M}_{tr}$ and $\mathcal{M}_{ev}$ since the task replay buffer is for the evaluation purpose for each task and should not be seen during task training.

Moreover, we adopt a variant of imitation learning, Replay Buffer Spiking (RBS) [19] to warm up the learning process. Before our agent interacts with the environment, we employ a rule-based agent crafted for MultiWoz to initialize both $\mathcal{M}_{tr}$ and $\mathcal{M}_{ev}$. Then, in steps 14-16, we collect new trajectories with our agent and push them into $\mathcal{M}_{ev}$. We uniformly sample from $\mathcal{M}_{ev}$ a mini-batch $\mathcal{B}_{ev}^{t_k}$, which can be a mixture of on-policy and relevant off-policy data, to calculate the task evaluation loss $\mathcal{L}_{t_k}$. As a result, θ_Q is updated as:

$$\theta_Q \leftarrow \theta_Q - \beta \nabla_{\theta_Q} \sum_{k=1}^{K} \mathcal{L}_{t_k}(\theta_Q^{(k)}; \mathcal{B}_{ev}^{t_k} \sim \mathcal{M}_{ev}). \tag{11}$$

During test, for an unseen domain, we adopt a similar off-policy approach for meta-adaptation. This train-test consistency circumvents the known difficulty in on-policy meta-adaptation with off-policy meta-training [7]. In fact, classic MAML for RL can be seen as a special case of our dual-replay architecture by setting the task evaluation memory $\mathcal{M}_{ev}$ to $|\mathcal{B}_{ev}^{t_k}|$.

5 Experiment

5.1 Setup

Dataset and task settings We use the benchmark multi-domain dialogue dataset MultiWoz 2.0 [10] for the evaluation. We adopt `attraction`, `restaurant`, `taxi` and `hospital` as source domains for

Systems	Hotel			Train			Average		
	Success	**Reward**	**Turns**	**Success**	**Reward**	**Turns**	**Success**	**Reward**	**Turns**
Few-Shot Models									
DQN-1K	0.00	-50.38	12.38	2.15	-56.02	20.60	1.08	-53.20	16.49
DTQN-1K	2.55	-47.77	12.84	8.90	-45.96	18.64	5.73	-46.87	15.74
Adaptive Models									
VANILLADQN	0.05	-58.93	20.99	0.00	-59.00	21.00	0.03	-58.97	21.00
DQN	3.90	-53.74	20.42	9.85	-45.52	19.34	6.88	-49.63	19.88
DTQN	4.15	-53.32	20.30	15.40	-37.93	18.41	9.78	-45.63	19.36
META-DTQN-SR	1.15	-57.41	20.79	4.35	-53.01	20.23	2.75	-55.21	20.51
META-DTQN	**11.45**	**-43.68**	**19.42**	**19.30**	**-32.81**	**17.97**	**15.38**	**-38.25**	**18.70**

Table 2: System performance in the composite-domain setting on 2,000 dialogues in the target domains. We show results in `Hotel` and `Train` as `Police` has only single-domain dialogue goal.

Systems	Single			Composite			Average		
	Success	**Reward**	**Turns**	**Success**	**Reward**	**Turns**	**Success**	**Reward**	**Turns**
DQN	90.20	65.98	4.26	40.40	-3.00	13.48	65.30	31.49	8.87
DTQN	91.70	68.13	3.91	74.85	42.92	8.90	83.28	55.53	6.41
META-DTQN-SR	83.80	57.00	5.57	33.95	-11.81	14.55	58.88	22.59	10.06
META-DTQN	**93.00**	**69.84**	**3.76**	**80.30**	**49.95**	**8.41**	**86.65**	**59.90**	**6.09**

Table 3: System performance on 2,000 dialogues in the training domains.

training (source task size $K = 4$), and use `hotel`, `train` and `police` as target domains for adaptation. This split makes sure that both train and test splits have domains with various frequency levels (see Appendix C for details). We propose two experiment settings: *single-domain* and *composite-domain*. For the *single-domain* setting, agents are trained and tested with only single-domain dialogue goals. In the *composite-domain* setting, for each task in meta-training, we first select a seed domain d^* and then sample domain composition which contains d^*. The trained model is then adapted and evaluated in various domain compositions containing d^*.

Systems We developed different baseline task-oriented dialogue systems: DQN is standard deep Q-learning which uses binary state representations and DTQN, which is our proposed model without meta-learning framework. We also build VANILLADQN without Replay Buffer Spiking (RBS) [19] to show the warm-up effects in adaptation from rule-based off-policy data. In addition, we build META-DTQN-SR with only one single replay buffer to show the effects of the dual-replay mechanism we proposed. During adaptation to target domains, we simulate the data scarcity scenario by using only 1,000 frames (i.e., 10% of the training data). Besides, to examine the effects of the two-stage paradigm of training-and-adaptation, we also report the results of two few-shot models, DQN-1K and DTQN-1K. Both models are trained from scratch with the 1,000 frames in the target domains.

Implementation Details We developed all variants of agents based on Convlab [4]. We used a batch size of 16 for both training and adaptation. We set the size of the training replay buffer and the evaluation replay buffer to 50,000. We initialized the replay buffers with Replay Buffer Spiking (RBS) [19] during the first 1000 episodes of meta-training, first 10 episodes of single-domain adaptation and first 50 episodes of composite-domain adaptation (see Appendix D for details).

5.2 Evaluation Results

Table 1 shows that our models (META-DTQN and DTQN) considerably outperform baseline systems on single-domain tasks in `hotel` and `train`. Dialogue tasks in `police` are relatively easy to accomplish, on which DQN-1K trained from scratch with only 1,000 frames can completely succeed. On the contrary, DQN-1K fails on all the tasks in `Hotel`. Also, note that DTQN-1K significantly outperforms DQN-1K across all domains. This demonstrates the effectiveness of modeling the dependency between the state and action space. Besides, the performance gain from meta-training is more significant in the `train` domain (i.e., 4.8% in success rate), which can be attributed to the similarity of state and action spaces between `train` and the source domain `taxi`.

Table 2 shows the adaptation results on the composite-domain setting, which is a much harder dialogue task. Here, META-DTQN has a clear advantage over other agents on both `hotel` and `train`,

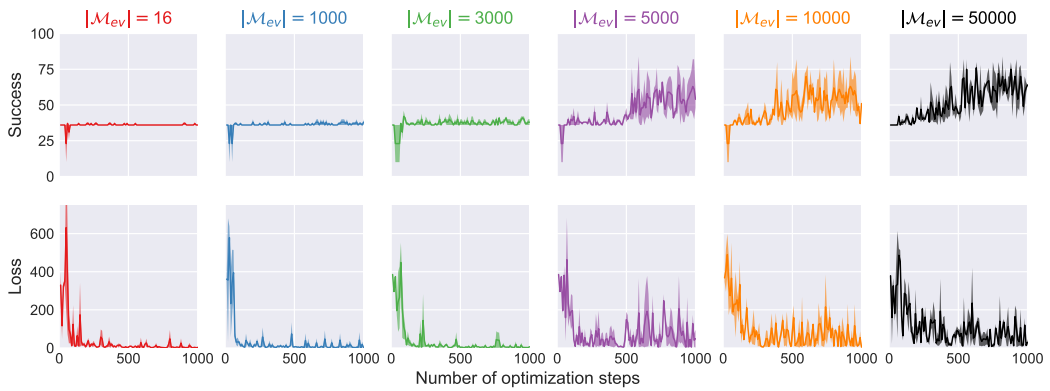

Figure 2: Development success rate (above) and training loss (below) of METADTQN with evaluation replay buffer of different sizes on composite-domain tasks. Shadow denotes variance.

showing that meta-learning can boost the quality of agents adapted with a small amount of data in complex dialogue tasks (see Appendix E for dialogue examples). Table 3 lists the performance of various models when evaluating on source domains. Here, meta-learning can also help to achieve better results, and the gain is larger in the more complex composite-domain settings.

It is worth noting that on all tasks, META-DTQN shows superior results than its single-replay counterpart, META-DTQN-SR, and the performance gap is particularly large on composite-domain dialogue tasks where the agent is more prone to suffer from initial reward sparsity.

Effects of dual replay We further investigate the effects of the proposed dual-replay method. In Figure 2, we show the performance of our model with a task evaluation memory of varied sizes. We start with pure on-policy evaluation $|\mathcal{M}_{ev}| = 16$, i.e., batch size $|\mathcal{B}_{ev}^{t_k}|$, and experiment with different buffer sizes: 16, 1000, 3000, 5000, 10000, and 50000. As shown, when the replay buffer is relatively small (< 5000), the success rate fails to improve. We argue that this optimization difficulty is due to the overfitting to on-policy data with sparse rewards at the beginning of the learning phase. This can be verified by the loss curve: the training loss abruptly drops from high values (100-500) to extremely low values (less than 10) soon after the RBS warm-up phase. When the evaluation memory size increases, our model is able to escape from the local minimum and get optimized continuously.

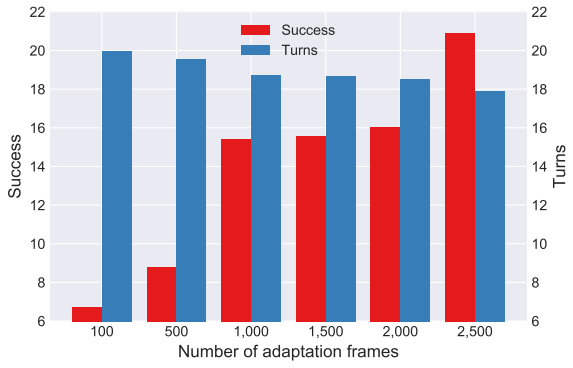

Figure 3: Performance of META-DTQN with adaptation data of varied sizes on composite tasks.

Effects of adaptation data size In addition, we show how the size of adaptation data affects the agent's performance on target domains in Figure 3. We test META-DTQN with adaptation data ranging from 100 frames (1% of the training data) to 2,500 frames (25% of the training data). As shown, the agent performance positively correlates with the amount of data available in the target domain. Note that as one episode has on average 10 frames, and we adopt RBS for the first 50 episodes, the agent is adapted only with off-policy rule-based experiences when the number of frames is less than 500. Therefore, the large performance gap between 500 and 1,000 frames indicates that our model can considerably benefit from a very small amount of on-policy data.

6 Conclusion

Dialogue policy is the central controller of a dialogue system and is usually optimized via reinforcement learning. However, it often suffers from insufficient training data, especially in multi-domain scenarios. In this paper, we propose the Deep Transferable Q-Network (DTQN) to share multi-level information between domains such as slots and acts. We also modify the meta-learning framework MAML and introduce a dual-replay mechanism. Empirical results show that our method outperforms traditional deep reinforcement learning models without domain knowledge sharing, in terms of both success rate and length of dialogue. As future work, we plan to generalize our method to more meta-RL applications in multi-domain and few-shot learning scenarios.

Broader Impact

Our work can contribute to dialogue research and applications, especially in new domains with scant training data. Our framework helps models quickly adapt to unseen domains to bootstrap applications. The outcome is a more effective and efficient dialogue agent system to facilitate activities in human society.

However, one needs to be cautious when collecting dialogue data, which may cause privacy issues. Deanonymization methods must be used to protect personal privacy.

References

[1] Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, Kam-Fai Wong, and Shang-Yu Su. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. *arXiv preprint arXiv:1801.06176*, 2018.

[2] Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2231–2240, 2017.

[3] Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and Li Deng. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[4] Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Zheng Zhang, Yaoqin Zhang, Xiang Li, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, et al. Convlab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 64–69, 2019.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[7] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International Conference on Machine Learning*, pages 5331–5340, 2019.

[8] Kun Qian and Zhou Yu. Domain adaptive dialog generation via meta learning. *arXiv preprint arXiv:1906.03520*, 2019.

[9] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

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

[11] Zhao Yan, Nan Duan, Peng Chen, Ming Zhou, Jianshe Zhou, and Zhoujun Li. Building task-oriented dialogue systems for online shopping. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[12] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[13] Heriberto Cuayáhuitl, Simon Keizer, and Oliver Lemon. Strategic dialogue management via deep reinforcement learning. *arXiv preprint arXiv:1511.08099*, 2015.

[14] Olivier Pietquin, Matthieu Geist, and Senthilkumar Chandramohan. Sample efficient on-line learning of optimal dialogue policies with kalman temporal differences. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[15] Da Tang, Xiujun Li, Jianfeng Gao, Chong Wang, Lihong Li, and Tony Jebara. Subgoal discovery for hierarchical dialogue policy learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2298–2309, 2018.

[16] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[17] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.

[18] 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. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[19] Zachary C Lipton, Jianfeng Gao, Lihong Li, Xiujun Li, Faisal Ahmed, and Li Deng. Efficient exploration for dialog policy learning with deep bbq networks & replay buffer spiking. *CoRR abs/1608.05081*, 2016.