Adaptive Dialog Policy Learning with Hindsight and User Modeling

Yan Cao[1] **Keting Lu**[2] **Xiaoping Chen**[1] **Shiqi Zhang**[3]
[1]School of Computer Science, University of Science and Technology of China
[2]Commercialization Recommending Researching Department, Baidu Inc.
[3]Department of Computer Science, SUNY Binghamton
caotian@mail.ustc.edu.cn; ktlu@mail.ustc.edu.cn;
xpchen@ustc.edu.cn; zhangs@binghamton.edu

Abstract

Reinforcement learning methods have been used to compute dialog policies from language-based interaction experiences. Efficiency is of particular importance in dialog policy learning, because of the considerable cost of interacting with people, and the very poor user experience from low-quality conversations. Aiming at improving the efficiency of dialog policy learning, we develop algorithm LHUA (Learning with Hindsight, User modeling, and Adaptation) that, for the first time, enables dialog agents to adaptively learn with hindsight from both simulated and real users. Simulation and hindsight provide the dialog agent with more experience and more (positive) reinforcements respectively. Experimental results suggest that, in success rate and policy quality, LHUA outperforms competitive baselines from the literature, as well as its no-simulation, no-adaptation, and no-hindsight counterparts.

1 Introduction

Dialog systems have enabled intelligent agents to communicate with people using natural language. For instance, virtual assistants, such as Siri, Alexa, and Cortana, have been increasingly popular in daily life. We are particularly interested in goal-oriented dialog systems, where the task is to efficiently and accurately exchange information with people, and the main challenge is on the ubiquitous ambiguity in natural language processing (spoken or text-based). Goal-oriented dialog systems typically include components for language understanding, dialog management, and language synthesis, while sometimes the components can be constructed altogether, resulting in end-to-end dialog systems (Bordes et al., 2016; Williams and Zweig, 2016; Wen et al., 2017; Young et al., 2018; Yang et al., 2017). In this paper, we focus on the problem of policy learning for dialog management.

Reinforcement learning (RL) algorithms aim at learning action policies from trial-and-error experiences (Sutton and Barto, 2018), and have been used for learning dialog policies (Young et al., 2013; Levin et al., 1997). Deep RL methods (e.g. (Mnih et al., 2013)) have been developed for dialog policy learning in dialog domains with large state spaces (Su et al., 2016a; Fatemi et al., 2016; Serban et al., 2017). While it is always desirable for RL agents to learn from the experiences of interacting with the real world, such interactions can be expensive, risky, or both in practice. Back to the context of dialog systems, despite all the advances in RL (deep or not), dialog policy learning remains a challenge. For instance, interacting with people using natural language is very costly, and low-quality dialog policies produce very poor user experience, which is particularly common in early learning phases. As a result, it is critical to develop sample-efficient RL methods for learning high-quality dialog policies with limited conversational experiences.

In this paper, we develop an algorithm called LHUA (Learning with Hindsight, User modeling, and Adaptation) for sample-efficient dialog policy learning. LHUA, for the first time, enables a dialog agent to simultaneously learn from real, simulated, and hindsight experiences, which identifies the key contribution of this research. Simulated experience is generated using learned user models, and hindsight experience (of successful dialog samples) is generated by manipulating dialog segments and goals of the (potentially many) unsuccessful samples. Dialog experience from simulation and hindsight respectively provide more dialog samples and more positive feedback for dialog policy learning. To further improve the sample efficiency, we develop a meta-agent for LHUA that adaptively

, pages 64 - 73

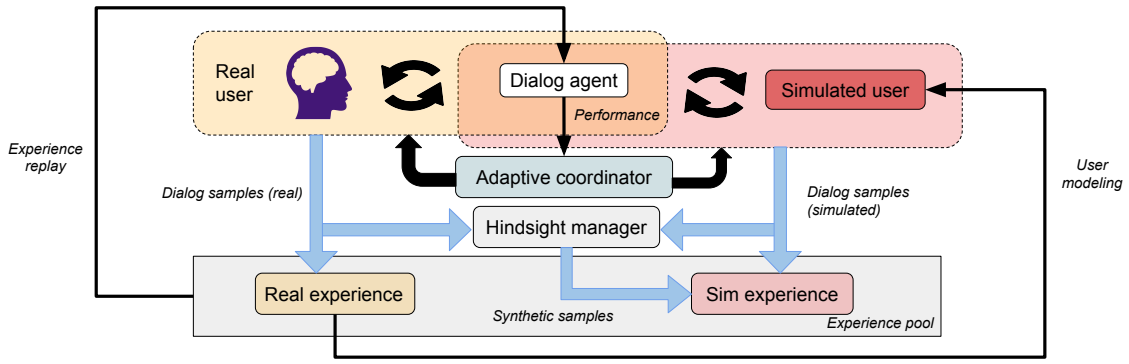

Figure 1: An overview of LHUA. A *dialog agent* interacts with both real and simulated users while learning a dialog policy from this interaction experience. A *simulated user* is modeled using real dialog samples, and interacting with this simulated user provides the dialog agent with simulated dialog samples. An *adaptive coordinator* learns from the dialog agent's recent performance to adaptively assign one user (real or simulated) for the dialog agent to interact with. A *hindsight manager* manipulates both real and simulated dialog samples (of mixed qualities) to "synthesize" successful dialog samples.

learns to switch between real and simulated users in the dialog-based interactions, which identifies the second contribution of this research. An overview of LHUA is shown in Figure 1.

Experiments were conducted using a realistic movie-ticket booking platform (Li et al., 2017). LHUA has been compared with state-of-the-art methods (Peng et al., 2018; Lu et al., 2019; Su et al., 2018) in dialog policy learning tasks. Results suggest that ablations of LHUA produce comparable (or better) performances in comparison to competitive baselines in success rate, and LHUA as a whole performed the best.

2 Related Work

In this section, we summarize three different ways of improving the efficiency of dialog policy learning (namely user modeling, hindsight experience replay, and reward shaping), and qualitatively compare them with our methods.

Researchers have developed "two-step" algorithms that first build user models through supervised learning with real conversational data, and then learn dialog policies by interacting with the simulated users (Schatzmann et al., 2007; Li et al., 2016b). In those methods, user modeling must be conducted offline before the start of dialog policy learning. As a result, the learned policies are potentially biased toward the historical conversational data. Toward online methods for dialog policy learning, researchers have developed algorithms for simultaneously constructing models of real users, and learning from the simulated interaction experience with user models (Asri et al., 2016; Su et al., 2016b; Lipton et al., 2016; Zhao and Eskenazi, 2016; Williams et al., 2017; Dhingra et al., 2017; Li et al., 2017; Liu and Lane, 2017; Peng et al., 2017; Wu et al., 2019; Li et al., 2016a). Those methods enable agents to simultaneously build and leverage user models in dialog policy learning. However, the problem of learning high-quality user models by itself can be challenging. Our algorithms support user modeling, while further enabling agents to adaptively learn from both hindsight and real conversations.

In comparison to many other RL applications, goal-oriented dialog systems have very sparse feedback from the "real world" (human users), where one frequently cannot tell dialogs being successful or not until reaching the very end. Positive feedback is even rarer, when dialog policies are of poor qualities. Hindsight experience replay (HER) (Andrychowicz et al., 2017) methods have been developed to convert unsuccessful trials into successful ones through goal manipulation. The "policy learning with hindsight" idea has been applied to various domains, including dialog (Lu et al., 2019). Our methods support the capability of learning from hindsight experience, while further enabling user modeling and learning from simulated users.

Within the dialog policy learning context, reward shaping is another way of providing the dialog agents with extra feedback, where a dense reward function can be manually designed (Su et al., 2015), or learned (Su et al., 2016b). Researchers also developed efficient exploration strategies to speed up the policy learning process of dialog agents, e.g., (Pietquin et al., 2011; Lagoudakis and Parr, 2003). Those methods are orthogonal to ours, and

can potentially be combined to further improve the dialog learning efficiency. In comparison to all methods mentioned in this section, LHUA is the first that enables dialog policy learning from real, simulated, and hindsight experiences simultaneously, and its performance is further enhanced through a meta-policy for switching between interactions with real and simulated users.

3 Background

In this section, we briefly introduce the two building blocks of this research, namely Markov decision process (MDP)-based dialog management, and Deep Q-Network (DQN).

3.1 MDP-based Dialog Management

Markov Decision Processes (MDPs) can be specified as a tuple $< \mathcal{S}, \mathcal{A}, T, \mathcal{R}, s_0 >$, where $\mathcal{S}$ is the state set, $\mathcal{A}$ is the action set, T is the transition function, $\mathcal{R}$ is the reward function, and s_0 is the initial state. In MDP-based dialog managers, dialog control can be modeled using MDPs for selecting language actions. $s \in \mathcal{S}$ represents the current dialog state including the agent's last action, the user's current action, the distribution of each slot, and other domain variables as needed. $a \in \mathcal{A}$ represents the agent's response. The reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbf{R}$ gives the agent a big bonus in successful dialogs, a big penalty in failures, and a small cost in each turn.

Solving an MDP-based dialog management problem produces π, a dialog policy. A dialog policy maps a dialog state to an action, $\pi : \mathcal{S} \rightarrow \mathcal{A}$, toward maximizing the discounted, accumulative reward in dialogs, i.e., $R_t = \sum_{i=t}^{\infty} \gamma^{i-t} r_i$, where $\gamma \in [0, 1]$ is a discount factor that specifies how much the agent favors future rewards.

3.2 Deep Q-Network

Deep Q-Network (DQN) (Mnih et al., 2015) is a model-free RL algorithm. The approximation of the optimal Q-function, $Q^* = Q(s, a; \theta)$, is used by a neural network, where a is an action executed at state s, and θ is a set of parameters. Its policy is defined either in a greedy way: $\pi_Q(s) = argmax_{a \in \mathcal{A}} Q(s, a; \theta)$ or being ϵ-greedy, i.e., the agent takes a random action in probability ϵ and action $\pi_Q(s)$ otherwise. The loss function for minimization in DQN is usually defined using TD-error:

$$\mathcal{L} = \mathbf{E}_{s,a,r,s'}[(Q(s, a; \theta) - y)^2], \qquad (1)$$

where $y = r + \gamma max_{a' \in \mathcal{A}} Q(s', a'; \theta)$.

To alleviate the problem of unstable or non-convergence of Q values, two techniques are widely used. One is called *target network* whose parameters are updated by θ once every many iterations in the training phase. The other technique is *experience replay*, where an experience pool ε stores samples, each in the form of (s_t, a_t, r_t, s_{t+1}). It randomly selects small batches of samples from ε each time during training. Experience replay can reduce the correlation between samples, and increases the data efficiency.

4 Algorithms

In this section, we first introduce Learning with Hindsight, and User modeling (**LHU**), and then present LHU with Adaptation (**LHUA**), where algorithms LHU and LHUA point to the main contribution of this research.

LHU, for the first time, enables a dialog agent to learn dialog policies from **three dialog sources**, namely real users, simulated users, and hindsight dialog experience. More specifically, a real user refers to the human who converses with the dialog agent, and a simulated user refers to a learned user model that captures real users' interactive behaviors with our dialog agent. In this way, a simulated user is used for generating "human-like" dialog experience for speeding up the process of dialog policy learning. The last dialog source of "hindsight dialog experience" is used for creating many *successful* dialog samples using both successful and unsuccessful dialog samples, where the source samples are from both real and simulated users. Different from "simulated users" that generate dialog samples of mixed qualities, hindsight experience produces only successful (though not real) dialog samples, which is particularly useful for dialog policy learning at the early phase due to the very few successful samples.

Among the three dialog sources, hindsight experience is "always on", and synthesizes dialog samples throughout the learning process. The "real" and "simulated" dialog sources bring in the selection problem: *At a particular time, from which source should the agent obtain dialog experience for policy learning?* The "adaptation" capability of LHUA aims at enabling the dialog agent to learn to, before starting a dialog, select which user (real or simulated) to interact with.

4.1 Learning with Hindsight, and User Modeling

In this subsection, we focus on two components of LHUA, including user modeling, and hindsight management, which together form LHU, an ablation algorithm of LHUA. The two components' shared goal is to generate additional dialog experience (simulated and hindsight experiences respectively) to speed up dialog policy learning.

Dialog (Sub)Goal and Segmentation Goal-oriented dialog agents help users accomplish their goals via language-based multi-turn communications. Goal G includes a set of constraints C and a set of requests R, where $G = (C, R)$. Consider a service request "*I'd like to purchase one ticket of Titanic for this evening. Which theater is available?*" In this example, the goal is of the form:

$$G = \big(C = [ticket = one, time = eve, movie = titanic], R = [theater = ?]\big)$$

We define G' as a subgoal of $G = (C, R)$: $G' = (C', R')$, where $C' \subseteq C$, $R' \subseteq R$, and G' cannot be empty. Continuing the "titanic" example, one of its subgoals is

$$G' = \big(C' = [ticket = one, movie = titanic], R' = \emptyset\big).$$

Given an intact dialog D, we say D_{seg} is a segment of D, if D_{seg} includes a consecutive sequence of turns of D. With the concepts of dialog segment and subgoal, we introduce two segment sets (head and tail), which are later used in *hindsight manager*. A head segment set Ω consists of dialog segments D_{head} that include the early turns in the intact dialog with the corresponding completed subgoal G'.

$$\Omega = \{(D_{head}, G')\} \tag{2}$$

We use function $HeadSegGen$ to collect a head segment set Ω during dialog interactions. $HeadSegGen$ receives a dialog segment D_{seg}, and a goal G, then checks all subgoals of G, and finally outputs pairs (D_{seg}, G') where D_{seg} accomplishes subgoal G' of G.

A tail segment set Γ consists of dialog segments D_{tail} that include the late turns in the intact dialog with the corresponding completed subgoal G'.

$$\Gamma = \{(D_{tail}, G')\} \tag{3}$$

Function $TailSegGen$ is implemented to generate tail segments after interactions terminate. It receives a dialog D, a goal G and a corresponding head segment Ω. If the dialog D accomplishes the goal G, for each pair (D_{head}, G') from the head segment set Ω, $TailSegGen$ outputs a corresponding pair $(D \ominus D_{head}, G')$, where $D_1 \ominus D_2$ produces a dialog segment by removing D_2 from D_1.

Hindsight Manager Given head and tail segment sets (Ω and Γ), the *hindsight manager* is used for stitching two tuples, (D_{head}, G'_{head}) and (D_{tail}, G'_{tail}), respectively to "synthesize" successful dialog samples. There are two conditions for synthesization:

1. The two subgoals from head and tail segments are identical, $G'_{head} == G'_{tail}$, and
2. The last state of D_{head}, s_{last}, and the first state of D_{tail}, s'_{first}, are of sufficient similarity.

We use *KL Divergence* to measure the similarity between two states:

$$D_{KL}(s_{last} || s'_{first}) \leq \delta \tag{4}$$

where $\delta \in R$ is a threshold parameter. We implement a function to synthesize successful dialog samples as hindsight experience for dialog policy learning, as follows:

$$D_{hind} \leftarrow HindMan(\delta, \Omega, \Gamma) \tag{5}$$

$HindMan$ takes a threshold δ, a head segment set Ω, and a tail segment set Γ. It generates successful dialog samples D_{hind} that satisfy the above two conditions of synthesization.

Dialog with Simulated Users In dialog policy learning, dialog agents can learn from interactions with real users, where the generated real experience is stored in reply buffer B^R. To provide more experience, we develop a simulated user for generating simulated dialog experience to further speed up the learning of dialog policies.

The simulated user is of the form:

$$s', r \leftarrow M(s, a; \theta_M)$$

where, $M(s, a; \theta_M)$ takes the current dialog state s and the last dialog agent action a as input, and generates the next dialog state s', and reward r. M is implemented by a Multi-Layer Perceptron (MLP) parameterized by θ_M, and refined via stochastic

Algorithm 1 Algorithm LHU

Input: K, the times of interactions with the simulated user; δ, KL-divergence threshold

Output: the success rate SR^{Dlg}, and average rewards R^{Dlg} of $agent^{Dlg}$; $Q(\cdot)$ for $agent^{Dlg}$

1: Initialize $Q(s, a; \theta_Q)$ of $agent^{Dlg}$ and $M(s, a; \theta_M)$ of the simulated user via pre-training on human conversational data
2: Initialize experience replay buffers B^R and B^S for the interaction of $agent^{Dlg}$ with real and simulated users
3: Initialize head and tail dialog segment sets:
$\Omega \leftarrow \emptyset$, and $\Gamma \leftarrow \emptyset$
4: Collect initial state, s, by interacting with a real user following goal G^{Real}
5: Initialize $D^{Real} \leftarrow \emptyset$ for storing dialog turns (real)
6: **while** $s \notin$ term **do** // Start a dialog with real user
7: Select $a \leftarrow argmax_{a'} Q(s, a'; \theta_Q)$, and execute a
8: Collect next state s', and reward r
9: Add dialog turn $d = (s, a, r, s')$ to B^R and D^{Real}
10: $\Omega \leftarrow \Omega \cup HeadSegGen(D^{Real}, G^{Real})$
11: $s \leftarrow s'$
12: **end while**
13: $\Gamma \leftarrow \Gamma \cup TailSegGen(D^{Real}, G^{Real}, \Omega)$
14: **for** $k = 1 : K$ **do** // K interactions with simulated user
15: Sample goal G^{Sim}, and initial state s
16: Initialize $D^{Sim} \leftarrow \emptyset$ for storing dialog turns (sim)
17: **while** $s \notin$ term **do** // The k^{th} dialog with sim user
18: $a \leftarrow argmax_{a'} Q(s, a'; \theta_Q)$, and execute a
19: Collect next state s', and reward r from $M(s, a; \theta_M)$
20: Add dialog turn $d = (s, a, r, s')$ to B^S and D^{Sim}
21: $\Omega \leftarrow \Omega \cup HeadSegGen(D^{Sim}, G^{Sim})$
22: $s \leftarrow s'$
23: **end while**
24: $\Gamma \leftarrow \Gamma \cup TailSegGen(D^{Sim}, G^{Sim}, \Omega)$
25: **end for**
26: Synthesize hindsight experience, and store it in B^S: $D_{hind} \leftarrow HindMan(\delta, \Gamma, \Omega)$ // Hindsight Manipulation
27: Calculate the success rate SR^{Dlg} and average rewards R^{Dlg} of total interactions
28: Randomly sample a minibatch from both B^R and B^S, and update $agent^{Dlg}$ via DQN // $agent^{Dlg}$ training
29: Randomly sample a minibatch from B^R, and update simulated user via SGD // User modeling
30: **return** SR^{Dlg}, R^{Dlg}, $Q(\cdot)$

gradient descent (SGD) using real experience in B^R to improve the quality of simulated experience.

Simulated experience generated from interactions between the dialog agent and the simulated user is stored in the simulated replay buffer B^S, which is also manipulated by the *hindsight manager* to synthesize hindsight experience.

The LHU Algorithm Algorithm 1 presents the learning process, where our dialog agent interacts with a real user for one dialog, and a simulated user for k dialogs. In addition to parameter k, there is a *KL-divergence* threshold δ as a part of the input. We refer to this algorithm using LHU(k).

Algorithm 1 starts with an initialization of the dialog agent's real and simulated experience replay buffers (B^R and B^S respectively), the model of the simulated user, $M(\theta_M)$, and two segment sets for *hindsight manager* (Ω and Γ respectively). In the first *while* loop (starting in Line 6), the dialog agent interacts with a real user and stores the real experience in B^R. Then, k dialogs with the simulated user are conducted in the *for* loop, where simulated experience is stored in B^S. During interactions with both real and simulated users, head and tail segment sets are simultaneously collected (Lines 21 and 24). After all dialog interactions end, the *hindsight manager* is used to synthesize successful dialog samples and store them in B^S. Finally, the dialog agent is trained on B^R and B^S, and the simulated user is trained on B^R.

The output of Algorithm 1 is used in the next section, where we introduce how to further enable the dialog agent to learn a meta-policy for adaptively determining which user (real or simulated) to interact with.

4.2 LHU with Adaptation (LHUA)

Adaptively determining which user (real or simulated) the LHU agent should interact with can further speed up the dialog policy learning process. The idea behind it is that, if a simulated user can generate high-quality, realistic dialog experience, interactions with the simulated user should be encouraged. To enable this adapative "switching" behaviors, we develop an *adaptive coordinator* that learns a meta-policy for selecting between real and simulated users for collecting interaction experience. We learn this adaptive coordinator using reinforcement learning, producing the LHUA algorithm, which is described next.

State In each turn of interaction with the LHU agent, *adaptive coordinator* updates the adaptation state s^A using the equation below:

$$s_i^A = \begin{cases} [0, 0, 0, 0] & i = 0 \\ [SR_i, R_i, SR_i - SR_{i-1}, R_i - R_{i-1}] & i > 0 \end{cases} \quad (6)$$

where SR_i and R_i are respectively average success rate and rewards from LHU agent's training performance at i^{th} episode. In practice, R is normalized to have values between 0 and 1, same as SR. This form of adaptation state provides accessible information on different training phrases to represent LHU agent 's current performance.

Action Based on the state s^A, *adaptive coordinator* chooses action k to determine, after each dialog

Algorithm 2 LHU with Adaptation (LHUA)

Input: H, the max length of adaptation episode; δ, KL-divergence threshold; N, training times

Output: Π, the dialog policy;

Initialize $A(s^A, k; \theta_A)$ of $agent^{Adp}$, and replay buffer B^A as empty
for $i = 1 : N$ **do**
  Initialize adaptation state s^A using Eqn. 6
  Initialize turn counter h: $h = 0$
  while $h \leq H$ **do**
    Select action k: $k \leftarrow argmax_{k'} A(s^A, k'; \theta_A)$
    Execute action k:

$$SR^{Dlg}, R^{Dlg}, Q(\cdot) \leftarrow LHU^1(k, \delta)$$

    Collect reward r^A via Eqn. 7, and next adaptation state $\hat{s}^A$ using Eqn. 6
    $B^A \leftarrow B^A \cup (s^A, k, r^A, \hat{s}^A)$, $s^A \leftarrow \hat{s}^A$, and $h \leftarrow h + 1$
  end while
  Sample a minibatch from B^A, and update θ_A via DQN
end for
for all $s \in \mathcal{S}$: $\Pi(s) \leftarrow argmax_{a'} Q(s, a'; \theta_Q)$
return $\Pi(\cdot)$

with the real user, how many dialogs should be conducted with the simulated user. The value of action k ranges from 1 to K.

Reward *Adaptive coordinator* receives immediate rewards after executing an action k (i.e. LHU(k)) each time. We use success rate increment of LHU agent to design the reward function, as shown below:

$$r_i^A = \frac{SR_i - SR_{i-1}}{SR_i} \cdot \frac{k_i}{L_i} \quad (0 < i \leq H) \quad (7)$$

where k_i is the i^{th} action chosen by *adaptive coordinator*, and L_i means the total number of times of interactions with both real and simulated users, i.e.$L_i = k_i + 1$. Reward is continuously harvested, until the H^{th} turn.

Due to the continuous state space, the approximated value function of *adaptive coordinator* is implemented using a two-layer fully connected neural network, $A(s^A, k; \theta_A)$, parameterized by θ_A. Interactions between the *adaptive coordinator* and the LHU agent start with an initial state. In each turn, the *adaptive coordinator* obtains the state s^A using Eqn. 6, and selects the action k via ϵ-greedy policy to execute. Then, the current training performance of LHU agent is used for acquiring the reward r^A using Eqn. 7, and updating the next state $\hat{s}^A$. Finally, the experience $(s^A, k, r^A, \hat{s}^A)$ is stored for meta-policy learning. We improve the value function by adjusting θ_A to minimize the mean-squared loss function.

The LHUA Algorithm Algorithm 2 presents the dialog policy learning process, where our dialog agent adaptively learns from both simulated and real users. In addition to parameter δ for KL-divergence threshold, there is parameter H representing the length of one episode for adaptive coordinator as a part of the input.

Algorithm 2 starts with an initialization of replay buffer B^A for adaptive coordinator, and the value function $A(s^A, k; \theta_A)$. Before the start of each episode, a turn counter h is initialized as zero for turn counting. Adaptive coordinator interacts with LHU agent for H turns while collecting and saving experience in B^A. At the end of each adaptation episode, we use DQN to update θ_A.

LHUA enables the dialog agent to simultaneously learn from the dialogs with both real and simulated users. At the same time, *hindsight manager* manipulates both real and simulated dialog samples to synthesize more successful dialog samples. The *adaptive coordinator* is learned at runtime for adaptively switching between real and simulated users in the dialog policy learning process to further improve the sample efficiency. So far, LHUA enables dialog agents to adaptively learn with hindsight from both simulated and real users.

5 Experiment

Experiments have been conducted in a dialog simulation platform, called TC-bot (Li et al., 2016b, 2017).[1] TC-bot provides a realistic simulation platform for goal-oriented dialog system research. We use its *movie-ticket booking* domain that consists of 29 slots of two types, where one type is on *search constraints* (e.g., number of people, and date), and the other is on *system-informable* properties that are needed for database queries (e.g., critic rating, and start time). The dialog agent has 11 dialog actions, representing the system intent (e.g., confirm question, confirm answer, and thanks).

A dialog is considered successful only if movie tickets are booked successfully, and the provided information satisfies all the user's constraints. By the end of a dialog, the agent receives a bonus (positive reward) of $2 * L$ if successful, or a penalty (negative reward) of $-L$ for failure, where L is the maximum number of turns allowed in each dialog. We set $L = 40$ in our experiments. The

[1] To avoid possible confusions, we use "real user" to refer to the user directly provided by TC-bot, and use "simulated user" to refer to the user model learned by our dialog agents.

agent receives a unit cost in each dialog turn to encourage shorter conversations.

Implementation Details In line with existing research (Peng et al., 2018), all dialog agents are implemented using Deep Q-Network (DQN). The DQN includes one hidden layer with 80 hidden nodes and ReLU activation, and its output layer of 11 units corresponding to 11 dialog actions. We set the discount factor $\gamma = 0.95$. The techniques of target network and experience replay are applied. Both B^R and B^S share the buffer size of 5000, and we use uniform sampling in experience replay. The target value function is updated at the end of each epoch. In each epoch, $Q(\cdot)$ and $M(\cdot)$ are refined using one-step 16-tuple-minibatch update. We then pre-filled the experience replay buffer with 100 dialogs before training. The simulated experience buffer B^S is initialized as empty. Neural network parameters are randomly initialized, and optimized using RMSProp (Hinton et al., 2012).

The simulated user model, $M(\cdot)$, is a multi-task neural network (Liu et al., 2015), and contains two shared hidden layers and three task-specific hidden layers, where each layer has 80 nodes. Stitching threshold of *hindsight manager* δ is set 0.2. The policy network of *adaptive coordinator* is a single-layer neural network of size 64. Parameters k and H are described in Algorithm 2, and have the value of $k = 20$ and $H = 8$.

LHUA and Three Baselines Our key hypothesis is that adaptively learning from real, simulated, and hindsight experiences at the same time performs better than baselines from the literature. To evaluate this hypothesis, we have selected three competitive baselines for goal-oriented dialog policy learning, including DDQ (Su et al., 2018), D3Q (Wu et al., 2019), and S-HER (Lu et al., 2019). In implementing the DDQ agent, the ratio of interaction experiences between simulated and real users is ten, which is consistent with the original implementation (Su et al., 2018). The differences between LHUA and the baseline methods are qualitatively discussed in Section 2.

It is necessary to explain how the curves are generated in the figures to be reported. For each of the four methods (LHUA and three baselines), we have conducted five "runs", where each run includes 250 episodes. In each run, after every single episode for learning, we let the dialog agent interact with the real user for 50 dialogs, only for evaluation. We then compute the success rate over the 50 dialogs. Each data point in the figure is an average over the five success rates collected from the five runs of each method.

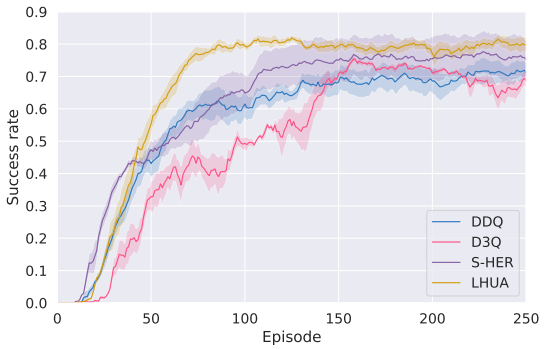

Figure 2: The performances of LHUA (ours), and three baseline methods, including DDQ (Su et al., 2018), D3Q (Wu et al., 2019), and S-HER (Lu et al., 2019). We see that, except for the very early phase (first 50 episodes), LHUA outperformed all baselines.

Figure 2 presents the key results of this research on the quantitative comparisons between LHUA and the three baselines. We can see that, except for the very early learning phase, LHUA performed consistently better than the three baseline methods. In particular, LHUA reached the success rate of 0.75 after about 70 episodes, whereas none of the baselines were able to achieve comparable performance within 150 episodes. The gap between LHUA and S-HER in early phase is due to the fact that LHUA needs to learn a user model, which requires extra interaction in early phase. Once the user model is of reasonable quality, LHUA is able to learn from the interaction experience with simulated users, and soon (after 45 episodes) LHUA outperformed S-HER.

LHUA and Its Ablations Results reported in Figure 2 have shown the advantage of LHUA over the three baseline methods. However, it is still unclear how much each component of LHUA contributes to its performance. We removed components from LHUA, and generated four different ablations of LHUA, including DQN, DDQ (LU, or Learning with User modeling), S-HER (LH, or Learning with Hindsight), LHU, and LHUA.

Figure 3 shows the ablation experiment's results. From the results, we see that LHUA performed much better than no-hindsight (LU), and no-user-modeling (S-HER, or LH) ablations. When both "hindsight" and "user modeling" are activated, there is LHUA's ablation of LHU, which performed bet-

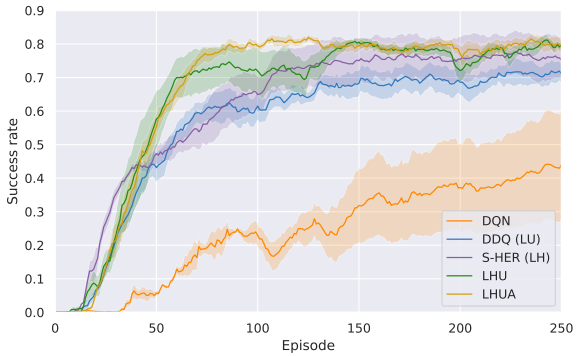

Figure 3: Comparisons between LHUA and its ablations: DQN (no hindsight manager, no user modeling, and no adaptive coordinator), DDQ (no hindsight manager, and no adaptive coordinator), S-HER (no user modeling, and no adaptive coordinator), and LHU (no adaptive coordinator). A complete LHUA includes all the components, including DQN (for naive dialog policy learning), hindsight manager, user modeling, and adaptive coordinator.

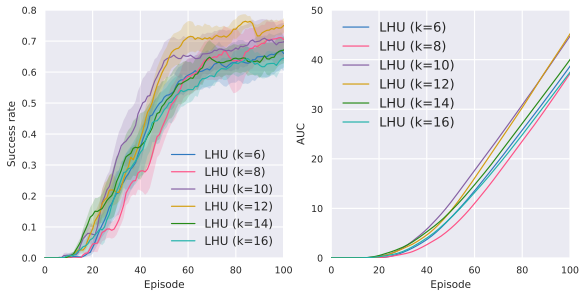

Figure 4: Success rate on the left, and Area under Curve (AUC) on the right, where we implemented six different versions of LHU with different k values, ranging from 6 to 16 at an interval of 2.

ter than all the other ablations. LHU still cannot generate comparable performance, c.f., LHUA, which justified the necessity of the adapative coordinator. It should be noted that performances of two of the ablations have been reported in Figure 2. We intentionally include their results in Figure 3 for the completeness of comparisons.

Adaptive Coordinator Learning Results reported in Figure 3 have shown the necessity of our adaptive coordinator in LHUA. In this experiment, we look into the learning process of the adaptive coordinator. More specifically, we are interested in how the value of k is selected (see Algorithm 2). We have implemented LHU with six different values of k, and their performances are reported in Figure 4, where the left subfigure is on success rate, and the right is on Area under Curve (AUC). The AUC metric has been used for the evaluation of learning speed (Taylor and Stone, 2009; Stadie et al., 2015). We see that, in early learning phase (within 100 episodes), the k value of 10 produced the best performance overall, though the performance is comparable to that with $k = 12$ to some level.

Figure 5 reports the selection of k values by our adaptive coordinator. Each bar corresponds to an average over the k values of 25 episodes. We see that the value of k was suggested to be around 10 within the first 100 episodes, which is consistent to our observation from the results of Figure 4. The consistency further justified our adaptive coordinator's capability of learning the interaction strategy in switching between real and simulated users.

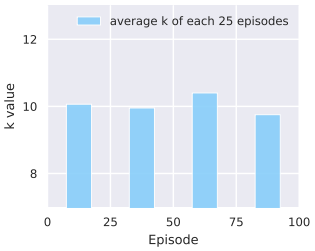

Figure 5: The k values selected by the *adaptive coordinator* of our LHUA agent

6 Conclusions and Future Work

In this work, we develop an algorithm called LHUA (Learning with Hindsight, User modeling, and Adaptation) for sample-efficient dialog policy learning. LHUA enables dialog agents to adaptively learn with hindsight from both simulated and real users. Simulation and hindsight provide the dialog agent with more experience and more (positive) reinforcements respectively. Experimental results suggest that LHUA outperforms competitive baselines (including success rate and learning speed) from the literature, including its no-simulation, no-adaptation, and no-hindsight counterparts. This is the first work that enables a dialog agent to adaptively learn from real, simulated, and hindsight experiences all at the same time.

In the future, we plan to evaluate our algorithm using other dialog simulation platform, e.g., PyDial (Ultes et al., 2017). Another direction is to combine other efficient exploration strategies, including learning directed exploration policies with different trade-offs between exploration and exploitation (Puigdomènech Badia et al., 2020). We will also focus on generating more synthetic dialog experience of different quality (Lu et al., 2020), to further improve the dialog learning efficiency.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China under grant number U1613216. Zhang is supported in part by grants from the National Science Foundation (IIS-1925044), Ford Motor Company (URP Award), OPPO (Faculty Research Award), and SUNY Research Foundation.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, et al. 2017. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pages 5048–5058.

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

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2016. Learning end-to-end goal-oriented dialog.

Bhuwan Dhingra, Lihong Li, Xiujun Li, et al. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. 2012. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*.

Michail G Lagoudakis and Ronald Parr. 2003. Least-squares policy iteration. *Journal of machine learning research*.

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

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016a. A persona-based neural conversation model. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Xiujun Li, Yun-Nung Chen, Lihong Li, et al. 2017. End-to-end task-completion neural dialogue systems. In *International Joint Conference on Natural Language Processing*.

Xiujun Li, Zachary C. Lipton, Bhuwan Dhingra, et al. 2016b. A user simulator for task-completion dialogues.

Zachary C. Lipton, Jianfeng Gao, Lihong Li, Xiujun Li, Faisal Ahmed, and Li Deng. 2016. Efficient exploration for dialogue policy learning with bbq networks & replay buffer spiking. Technical report.

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)*. IEEE.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, et al. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Keting Lu, Shiqi Zhang, and Xiaoping Chen. 2019. Goal-oriented dialogue policy learning from failures. *Proceedings of the AAAI Conference on Artificial Intelligence*.

Keting Lu, Shiqi Zhang, and Xiaoping Chen. 2020. Autoeg: Automated experience grafting for off-policy deep reinforcement learning. *arXiv preprint arXiv:2004.10698*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. 2015. Human-level control through deep reinforcement learning. *Nature*.

Baolin Peng, Xiujun Li, Jianfeng Gao, et al. 2018. Deep Dyna-Q: Integrating planning for task-completion dialogue policy learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Baolin Peng, Xiujun Li, Lihong Li, et al. 2017. Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Olivier Pietquin, Matthieu Geist, Senthilkumar Chandramohan, and Hervé Frezza-Buet. 2011. Sample-efficient batch reinforcement learning for dialogue management optimization. *ACM Transactions on Speech and Language Processing (TSLP)*.

Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, et al. 2020. Never give up: Learning directed exploration strategies. *arXiv*.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, et al. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *The Conference of the North American Chapter of the Association for Computational Linguistics*.

Iulian V. Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, et al. 2017. A deep reinforcement learning chatbot.

Bradly C Stadie, Sergey Levine, and Pieter Abbeel. 2015. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*.

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, et al. 2016a. Continuously learning neural dialogue management.

Pei-Hao Su, Milica Gašić, Nikola Mrkšić, et al. 2016b. On-line active reward learning for policy optimisation in spoken dialogue systems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Pei-Hao Su, David Vandyke, Milica Gasic, Nikola Mrksic, Tsung-Hsien Wen, and Steve Young. 2015. Reward shaping with recurrent neural networks for speeding up on-line policy learning in spoken dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

Shang-Yu Su, Xiujun Li, Jianfeng Gao, et al. 2018. Discriminative deep dyna-q: Robust planning for dialogue policy learning. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction*. MIT Press.

Matthew E Taylor and Peter Stone. 2009. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685.

Stefan Ultes, Lina M Rojas Barahona, Pei-Hao Su, et al. 2017. Pydial: A multi-domain statistical dialogue system toolkit. In *Proceedings of ACL 2017, System Demonstrations*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, et al. 2017. A network-based end-to-end trainable task-oriented dialogue system. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

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. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*.

Jason D Williams and Geoffrey Zweig. 2016. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning. *arXiv preprint arXiv:1606.01269*.

Yuexin Wu, Xiujun Li, Jingjing Liu, et al. 2019. Switch-based active deep dyna-q: Efficient adaptive planning for task-completion dialogue policy learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

X. Yang, Y. Chen, D. Hakkani-Tür, P. Crook, et al. 2017. End-to-end joint learning of natural language understanding and dialogue manager. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

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

Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. 2018. Augmenting end-to-end dialogue systems with commonsense knowledge. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.