The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Goal-Oriented Dialogue Policy Learning from Failures

Keting Lu,[1] **Shiqi Zhang,**[2] **Xiaoping Chen**[1]
[1]School of Computer Science, University of Science and Technology of China
[2]Department of Computer Science, SUNY Binghamton
ktlu@mail.ustc.edu.cn, szhang@cs.binghamton.edu, xpchen@ustc.edu.cn

Abstract

Reinforcement learning methods have been used for learning dialogue policies. However, learning an effective dialogue policy frequently requires prohibitively many conversations. This is partly because of the sparse rewards in dialogues, and the very few successful dialogues in early learning phase. Hindsight experience replay (HER) enables learning from failures, but the vanilla HER is inapplicable to dialogue learning due to the implicit goals. In this work, we develop two complex HER methods providing different trade-offs between complexity and performance, and, for the first time, enabled HER-based dialogue policy learning. Experiments using a realistic user simulator show that our HER methods perform better than existing experience replay methods (as applied to deep Q-networks) in learning rate.

Introduction

Goal-oriented dialogue systems aim at assisting users to accomplish specific goals using natural language, and have been used in a variety of applications (Zhang and Stone 2015; Su et al. 2016b; Li et al. 2017). Goal-oriented dialogue systems usually aim at *concise* conversations.[1] Such dialogue systems typically include a language understanding component for recognizing and parsing the language inputs into inner representations, a belief state tracking component for predicting user intent and updating the dialogue history, and a dialogue management component that generates dialogue actions. The dialogue actions can be converted into spoken or text-based language using a language generator. Goal-oriented dialogue managers are frequently modeled as a sequential decision-making problem (Young et al. 2013), where reinforcement learning (RL) (Sutton and Barto 1998) can be used for learning an optimal dialogue policy from user experiences. While a variety of RL methods have been developed for learning dialogue policies (Williams and Zweig 2016; Cuayáhuitl 2017), the methods typically require a large amount of dialogue experience until one can learn a good-quality dialogue policy. In particular, successful dialogues are rare in early learning phase, making it a challenge for many dialogue systems to learn much at the beginning and producing poor user experiences. This paper focuses on runtime dialogue data augmentation (DDA) for speeding up the process of learning goal-oriented dialogue policies.

The idea of data augmentation for RL tasks is not new. Existing research on hindsight experience replay (HER) has shown that, in robotic manipulation tasks, an RL agent's learning rate can be improved by changing goals to generate "successful" instances (Andrychowicz et al. 2017). However, their HER approach is not directly applicable to dialogue domains, because a dialogue agent has to interact with people to identify the desired goal state (whereas the goal state is explicit in manipulation tasks). We develop two complex HER methods for runtime DDA problems. The first method is called Trimming-based HER (**T-HER**), where failed dialogues are trimmed to generate successful dialogue instances. While T-HER enables the agent to significantly augment the training data, the generated instances are relatively short. The other method is Stitching-based HER (**S-HER**) that enables an agent to analyze the similarity between dialogue belief states, and stitch together dialogue segments to form relatively long, successful dialogues.

Figure is an overview of our DDA framework, illustrating the role of our HER methods. Like standard experience replay, we add user dialogues to the experience pool. Unlike existing methods, when incoming dialogues are unsuccessful, we use our HER methods to generate new, successful dialogues, enabling our agent to learn from failures.

We have applied our DDA framework, including T-HER and S-HER, to Deep Q-Networks (DQNs) (Mnih et al. 2015) for learning dialogue policies. Results collected using a realistic user simulator (Li et al. 2016) suggest that our methods perform better than competitive experience replay baselines, including prioritized experience replay (Schaul et al. 2015). Finally, our two HER methods can be combined to further improve the performance. To the best of our knowledge, this is the first work on HER-based dialogue policy learning.

Related Work

This work is closely related to research areas that aim at efficient (deep) RL methods, including, experience replay, reward shaping, exploration in RL, and supervised pre-training for RL.

[1]In comparison, the Chatbots that want to maximize social engagement, such as Microsoft XiaoIce, frequently result in extended conversations.

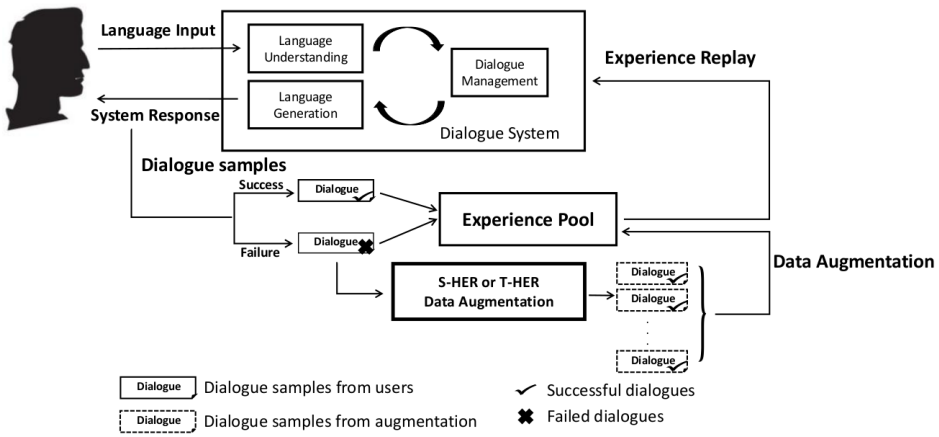

Figure 1: Overview of our dialogue data augmentation (DDA) framework, and our two complex hindsight experience replay (HER) methods.

Deep Q-Network (DQN) has enabled an agent to achieve human-level performance on playing Atari games (Mnih et al. 2015). RL algorithms, such as DQN, are generally data intensive and frequently require huge numbers of interactions with the environments. To better use the interaction experience, experience replay (ER), that suggests storing and reusing samples at training time, has been widely used for speeding up the RL agent's training process (Nair et al. 2015; Wang et al. 2016). Prioritized ER (PER) further accelerates training by assigning a weight based on temporal difference error (TD-error) to each sample (Schaul et al. 2015), so as to increase the likelihood of selecting samples with a high TD-error. While PER enables more effective sample selections (Han and Sung 2017), the applicability of PER is limited when very few successful samples are available. We develop HER methods that generate artificial "successful" samples to improve the learning rate of dialogue agents.

Reward shaping has been used for speeding up the learning of dialogue policies by adding a new dense reward function (Ferreira and Lefevre 2015). It has been proved that a well designed (dense) reward function does not alter the optimal policy (Ng, Harada, and Russell 1999). Recently, Su et al. applied Recurrent Neural Networks (RNNs) to predict the intermediate rewards for dialogues to reduce training time, and developed an on-line learning framework where dialogue policy and reward function were jointly trained via active learning (Su et al. 2016b). However, generating such complex reward functions require either considerable human effort or large datasets.

Another line of research focuses on developing effective exploration strategies for RL algorithms, enabling more sample-efficient dialogue learning. For instance, Pietquin et al. integrated Least-Squares Policy Iteration (Lagoudakis and Parr 2003) and Fitted-Q (Chandramohan, Geist, and Pietquin 2010) for dialogue policy optimization (Pietquin et al. 2011). Other examples include Gaussian Process (GP)-based sample-efficient dialogue learning (Gašić and Young 2014), the Bayes-by-Backprop Q-network (BBQN) (Lipton et al. 2017), and trust-region and gradient-based algorithms (Su et al. 2017). Our HER methods have the potential to be combined with the above sample-efficient RL methods to produce a more efficient learning process.

Pre-training has been used in dialogue learning for computing an initial policy from a corpus using supervised learning (SL) (Su et al. 2016a; Peng et al. 2017). After that, a dialogue agent can further improve the policy via learning from interactions with users. In line with past research on dialogue systems (as listed above), we use pre-training in this work (unless stated otherwise) to give our dialogue agent a "warm start".

Work closest to this research is the original HER method (Andrychowicz et al. 2017) that manipulates goals based on resulting states. But that method is only applicable to domains where goals are explicit to the agents, e.g., target positions in manipulation tasks. In dialogue domains, goals are not fully observable and must be identified via language actions, which motivates the development of complex HER methods in this work.

Background

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

MDP-based Dialogue Management

Dialogue control is modeled using MDPs in this work. An MDP-based dialogue manager (Lipton et al. 2017) can be described as a tuple $\langle \mathcal{S}, \mathcal{A}, T, s_0, \mathcal{R} \rangle$. $\mathcal{S}$ is the state set, where $s \in \mathcal{S}$ represents the agent's current dialogue state including the agent's last action, the user's current action, the distribution of each slot, and other domain variables as needed. $\mathcal{A}$ is the action set, where $a \in \mathcal{A}$ represents the agent's response. T is the stationary, probabilistic transition function with conditional density $p(s_{t+1}|s_t, a_t)$ that satisfies the (first-order) Markov property. $s_0 \in \mathcal{S}$ is the initial state. $\mathcal{R}$: $\mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is the reward function, where the agent receives a big bonus in successful dialogues, and has a small cost in each turn.

Solving an MDP-based dialogue management problem produces π, a dialogue policy. A dialogue policy maps a dialogue state to an action, π: $\mathcal{S} \rightarrow \mathcal{A}$, toward maximizing the discounted, accumulative reward in dialogues, 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.

Deep Q-Network

Deep Q-Network (DQN) (Mnih et al. 2015) is a model-free RL algorithm for discrete action space. DQN uses a neural network as an approximation of the optimal Q-function, $Q^* = Q(s, a; \theta)$, 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: $\pi_Q(s) = \arg\max_{a \in \mathcal{A}} Q(s, a; \theta)$ or being $\epsilon - greedy$, 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:

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

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

Naive DQNs often suffer from overestimation and instability, and two techniques are widely used to alleviate the issues. One is called *target network* (Mnih et al. 2015) whose parameters are updated by θ once every many iterations in the training phase. The other technique is *experience replay* (Lin 1993; Mnih et al. 2015), where an experience pool $\mathcal{E}$ stores samples, each in the form of (s_t, a_t, r_t, s_{t+1}). In training the DQN, a mini-batch is uniformly sampled from $\mathcal{E}$. Experience replay desensitizes DQN to the correlation among the samples, and increases the data efficiency via reusing the (potentially expensive) samples. Both techniques improve the performance of DQN and are used in this paper.

Dialogue Segmentation

In this section, we introduce the concept of *dialogue subgoal*, and present a segmentation algorithm that efficiently outputs *valid* dialogue segments, which are later used in our complex HER methods.

Definition of Dialogue Subgoal

Goal-oriented dialogue systems help users accomplish their goals via dialogue, where a goal G includes a set of constraints C and a set of requests R (Schatzmann and Young 2009): $G = (C, R)$.

Consider a movie booking domain. A user may ask about the *name* and *time* of a movie starring *Jackie Chan*, of genre *action*, and running *today*, where the goal is in the form of:

$$\textbf{Goal} = \left(C = \begin{bmatrix} actor = Jackie\ Chan \\ genre = action \\ date = today \end{bmatrix}, \; R = \begin{bmatrix} movie\ name = \\ start\ time = \end{bmatrix} \right)$$

Definition 1 ***Subgoal***[2] *Given $G = (C, R)$ and $G' = (C', R')$, we say G' is a subgoal of G, or $G' \sqsubset G$, if $C' \subset C$ and $R' \subset R$, where $G' \neq \emptyset$.*

[2]Our definition of *subgoal* is different from that in the dialogue learning literature on hierarchical reinforcement learning (Tang et al. 2018) and state space factorization (Thomson 2013).

$$\textbf{Subgoal 1} = \left(C = \begin{bmatrix} actor = Jackie\ Chan \\ genre = action \end{bmatrix}, \; R = \emptyset \right).$$

$$\textbf{Subgoal 2} = \left(C = \emptyset, \; R = \begin{bmatrix} movie\ name = \\ start\ time = \end{bmatrix} \right).$$

$$\textbf{Subgoal 3} = \left(C = \begin{bmatrix} actor = Jackie\ Chan \\ date = today \end{bmatrix}, \; R = \begin{bmatrix} movie\ name = \\ start\ time = \end{bmatrix} \right).$$

Figure 2: Example subgoals in the movie booking domain. For instance, **Subgoal 3** corresponds to the request of "*Please tell me the name and start time of the movie that is playing today and stars Jackie Chan.* "

In successful dialogues, the agent must correctly identify all constraints and requests in G, so as to correctly provide the requested information via querying a database. Figure 2 shows three example subgoals (out of many) in the movie booking example.

It should be noted that some subgoals do not make sense to humans, but can be useful for dialogue learning. For instance, Subgoal 2 corresponds to a query about *movie name* and *start time* without any constraints. Real users do not have such goals, but an agent can still learn from the experience of achieving such subgoals.

Continuing the "Jackie Chan" example, if the agent misidentifies *genre*, *start time*, or both, the dialogues will be deemed unsuccessful, meaning that the agent cannot learn much from it, even though the agent has correctly identified the other entries of *movie name*, *actor*, etc. In this work, we make use of such unsuccessful dialogues in the training process, leveraging the fact that the agent has achieved subgoals (not all) in these dialogues.

Dialogue Segment Validation

Given dialogues $\mathcal{D}'$ and $\mathcal{D}$, we say $\mathcal{D}'$ is a segment of $\mathcal{D}$, if $\mathcal{D}'$ includes a consecutive sequence of turns of $\mathcal{D}$. We introduce an assessment function[3], $success(G, \mathcal{D})$, that outputs *true* or *false* representing whether dialogue $\mathcal{D}$ accomplishes goal (or subgoal) G or not. Using the assessment function, we define the validity of dialogue segments.

Definition 2 ***Validity*** *of dialogue segments: Given dialogue $\mathcal{D}$, goal G, and dialogue segment $\mathcal{D}'$ (of $\mathcal{D}$), we say $\mathcal{D}'$ is a valid dialogue segment of $\mathcal{D}$, iff there exists a subgoal $G' \sqsubset G$, and $success(G', \mathcal{D}')$ is* true.

Using Definitions 1 and 2, one can assess the validity of dialogue segment D', using the entire dialogue D, the goal of this dialogue G, and the provided subgoal G'. However, there exist many subgoals of the ultimate goal (combinatorial explosion), and it soon becomes infeasible to assess the validity of a dialogue segment. Formally, given

[3]Such assessment functions are provided by dialogue simulators, e.g., TC-Bot (https://github.com/MiuLab/TC-Bot).

Algorithm 1 Dialogue Segmentation

Input:
Goal G that includes constraint set C and request set R;
Assessment function $success(\cdot, \cdot)$;
Output:
A collection of pairs of a valid (head) dialogue segment and a corresponding subgoal, Ω;

Initialize $\Omega = \emptyset$
Initialize $\mathbb{P} = \emptyset$ and $\mathbb{Q} = C \cup R$
while Dialogue $\mathcal{D}$ is not ended **do**
  Outcome flag $segment\_outcome = False$
  for $q \in \mathbb{Q}$ **do**
    Construct a subgoal $G' = \mathbb{P} \cup q$
    if $success(G', \mathcal{D})$ **then**
      $segment\_outcome = True$
      $\mathbb{P} \leftarrow \mathbb{P} \cup q$ and $\mathbb{Q} \leftarrow \mathbb{Q} \setminus q$
    end if
  end for
  if $segment\_outcome == True$ **then**
    $\Omega \leftarrow \Omega \cup <D, \mathbb{P}>$
  end if
end while

goal $G = (C, R)$, the number of subgoals is $\sum_{i=0}^{|C|} \binom{i}{|C|} \cdot \sum_{j=0}^{|R|} \binom{j}{|R|} - 2$, where we subtract the two extreme cases of the subgoal being $\emptyset$ and G. If $|C| = 5$ and $|R| = 5$, the number of subgoals is 1598.

Dialogue Segmentation Algorithm

Instead of assessing the validity of a dialogue segment using *all* subgoals, we aim at using only the ones with the so-far-highest "cardinality". The intuition comes from the following two observations. In line with previous research on dialogue policy learning, e.g., (Schatzmann et al. 2007) , we assume users are cooperative and consistent in dialogues.

1. During a dialogue, the number of constraints and requests that have been identified is monotonically increasing; and
2. When a subgoal is accomplished, all its "subsubgoals" are automatically accomplished.

Leveraging the above two observations, we develop a *dialogue segmentation* algorithm to efficiently identify valid dialogue segments. First, we initialize two collections $\mathbb{P}$ and $\mathbb{Q}$. $\mathbb{P}$ stores the constraints and requests that have been identified during the dialogues, and $\mathbb{Q}$ stores the ones that have not been identified. Therefore, at the very beginning of dialogues, $\mathbb{P}$ is an empty set, and $\mathbb{Q}$ stores all constraints and requests. After each dialogue turn, we generate a subgoal set $\mathbb{G}$ where all subgoals share the same (so far highest) cardinality. Formally, $\mathbb{G} = \{G' | G' = \mathbb{P} \cup q, \forall q \in \mathbb{Q}\}$. If $G' \in \mathbb{G}$ is accomplished by the current dialogue segment, the corresponding q is removed from $\mathbb{Q}$ and added into $\mathbb{P}$, which is used to generate the subgoal set for the next dialogue segment. So at each dialogue turn, only a small set of subgoals (i.e., $\mathbb{G}$) is used to assess a dialogue segment. More detailed procedure is shown in Algorithm 1. Compared to exhaustive subgoal identification that suffers from combinatorial explosion, our dialogue segmentation algorithm has $O(|C| + |R|)$ time complexity.

Building on our dialogue segmentation algorithm, we next introduce our runtime data augmentation methods for efficient dialogue policy learning.

Our Complex HER Methods for Dialogue Data Augmentation

In this section, we introduce two complex hindsight experience replay (HER) methods for dialogue data augmentation (DDA). The original HER method (Andrychowicz et al. 2017) is not applicable to dialogue domains, because goals in dialogues are not explicitly given (c.f., path planning for robot arms) and must be identified in dialogue turns. We say our HER methods are "complex", because our methods manipulate dialogue experiences based on the resulting dialogue states (which is more difficult than goal manipulation). The HER methods generate successful, artificial dialogues using dialogues from users (successful or not), and are particularly useful in early learning phase, where successful dialogues are rare.

Trimming-based HER (T-HER)

The idea of T-HER is simple: we pair valid dialogue segments (from Algorithm 1) and their corresponding subgoals to generate successful dialogue instances for training.

It requires care in rewarding the success of the generated dialogues (c.f., the ones from users). We want to encourage the agent to accomplish subgoals (particularly the challenging ones) using positive rewards, while avoiding the agent sticking to accomplishing only the subgoals. Given R_{max} and R_{min} being the reward and penalty to successful and unsuccessful dialogues with users, we design the following (simple) reward function for the successful, artificial dialogues from T-HER.

$$R(\mathcal{D}) = \alpha \cdot |G'|, \text{ ensuring } \alpha \cdot |G'| < R_{max} \qquad (2)$$

where α is a weight, G' is a subgoal, $\mathcal{D}$ is a dialogue segment, and $|G'|$ is the number of entries of G' that must be identified. The agent receives a small penalty (-1 in our case) at each turn to encourage short dialogues.

T-HER enables a dialogue agent to learn from the experience of completing imaginary, relatively easy tasks, and is useful while accomplishing ultimate goals is challenging (e.g., in early learning phase). However, as a relatively simple strategy for DDA, T-HER cannot generate dialogues that are more complex that those from users.

Stitching-based HER (S-HER)

S-HER is relatively more complex than T-HER, while both need valid dialogue segments from Algorithm 1. T-HER uses the segments as successful dialogues that accomplish subgoals, whereas S-HER uses them as parts to construct new, successful dialogues. The key questions to S-HER are *what dialogue segments are suitable for stitching,* and *how they are stitched together*. Next, we define *stitchability* of dialogue segments using *KL Divergence*:

$$D_{KL}(s_0 || s_1) = \sum_{i=0}^{n} s_0(i) \cdot \log \frac{s_0(i)}{s_1(i)}$$

Algorithm 2 Stitching-based HER

Input:
Dialogue $\mathcal{D}$, and its goal G
A collection of pairs of a valid (tail) dialogue segment and a corresponding subgoal, Γ
KL-divergence threshold, ε
Assessment function, $success(\cdot,\cdot)$
Output:
A set of newly generated dialogues, $\mathbb{M}$

1: Initialize $\mathbb{M} \leftarrow \emptyset$
2: Assess input dialogue: *success_flag* $= success(G, \mathcal{D})$
3: Call Algorithm 1 to compute Ω, using $\mathcal{D}$, G, and $success(\cdot,\cdot)$, where Ω is a collection of pairs of a valid (head) dialogue segment and a corresponding subgoal
4: **for** $<\mathcal{D}', G'> \in \Omega$ **do**
5: **if** *success_flag* **is** *True* **then**
6: $\Gamma \leftarrow \Gamma \cup <\mathcal{D} \ominus \mathcal{D}', G'>$, where $\ominus$ is a dialogue subtraction operator
7: **else**
8: **for** $<\mathcal{D}'', G''> \in \Gamma$ **do**
9: **if** G' **is** G'' *and* $D_{KL}(\mathcal{D}'||\mathcal{D}'') \leq \varepsilon$ **then**
10: $\mathcal{D}^{stitched} = concatenate(\mathcal{D}', \mathcal{D}'')$
11: $\mathbb{M} \leftarrow \mathbb{M} \cup \mathcal{D}^{stitched}$
12: Break
13: **end if**
14: **end for**
15: **end if**
16: **end for**
17: Return $\mathbb{M}$

where s_0 and s_1 are two probability distributions.

Definition 3 ***Stitchability:*** *Consider two dialogue segments*

$$\mathcal{D} = (s_0, a_0, s_1), (s_1, a_1, s_2), \cdots, (s_{M-1}, a_{M-1}, s_M),$$

$$\mathcal{D}' = (s'_0, a'_0, s'_1), (s'_1, a'_1, s'_2), \cdots, (s'_{N-1}, a'_{N-1}, s'_N),$$

where M and N are turn numbers (dialogue segment lengths) of $\mathcal{D}$ and $\mathcal{D}'$, and each turn includes initial state, dialogue action, and resulting state.

If the dialogues' corresponding (sub)goals are G and G', we say $\mathcal{D}$ and $\mathcal{D}'$ are stitchable, if and only if $G = G'$, and

$$D_{KL}(s_M \,||\, s'_0) \leq \varepsilon,$$

where $\varepsilon \in \mathbb{R}$ is a stitchability threshold.

We use *head (tail) dialogue segment* to refer to the segment that includes the early (late) turns in the resulting dialogue. In Definition 3, $\mathcal{D}$ and $\mathcal{D}$ are head and tail segments respectively.

The key idea of S-HER is to use unsuccessful user dialogues to generate valid *head* dialogue segments (stored in set Ω), use successful user dialogues to generate valid *tail* dialogue segments (stored in set Γ), and stitch dialogue segments from the two sets (one from each set) to form new successful artificial dialogues, only if their connecting dialogue states are similar enough.

Algorithm 2 presents S-HER. Given a new user dialogue $\mathcal{D}$, S-HER first calls Algorithm 1 to generate all valid dialogue segments that start from the beginning of $\mathcal{D}$, and stores them in set Ω. After that, S-HER assesses $\mathcal{D}$'s successfulness using its associated goal, and stores the result in *success_flag*. If successful, S-HER subtracts segments in Ω from $\mathcal{D}$ to generate the tail segments that eventually lead to successful dialogues. The tail segments are used to augment Γ. $\ominus$ in Line 6 is a dialogue subtraction operator, i.e., $\mathcal{D} \ominus \mathcal{D}'$ produces a dialogue segment by removing $\mathcal{D}'$ from $\mathcal{D}$, and this operation is valid, only if $\mathcal{D}'$ is a segment of $\mathcal{D}$. If the new user dialogue is unsuccessful, S-HER concatenates one head segment from Ω to one tail segment from Γ to form a new, successful dialogue $\mathcal{D}^{stitched}$, and add it into set $\mathbb{M}$, which is used for storing the algorithm output.

Remarks: S-HER uses KL divergence (Line 9 in Algorithm 2) to measure the similarity between dialogue states. If their divergence is lower than a threshold, S-HER stitches the corresponding two dialogue segments together to generates a new dialogue. S-HER is more effective than T-HER, when there are very few successful dialogues in the experience replay pool. This is because S-HER is able to generate many successful dialogues, even if there is only one successful dialogue from user, which significantly augments the dialogue data for accelerating the training process.

Experiment

Experiments have been conducted to evaluate the key hypothesis of T-HER and S-HER being able to improve the learning rate of DQN-based dialogue policies. We have combined the two methods with prioritized experience replay (Schaul et al. 2015) to produce better results, and compared T-HER and S-HER under different conditions.

Dialogue Environment

Our complex HER methods were evaluated using a dialogue simulation environment, where a dialogue agent communicates with simulated users on *movie-booking* tasks (Li et al. 2016; 2017). This movie-booking domain 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).

A dialogue state consists of five components: 1) one-hot representations of the current user action act and mentioned slots; 2) one-hot representations of last system action act and mentioned slots; 3) the belief distribution of possible values for each slot; 4) both a scalar and one-hot representation of the current turn number; and 5) a scalar representation indicating the number of results which can be found in a database according to current search constraints.

The system (dialogue agent) has 11 dialogue actions representing the system intent. These actions can be mapped into a natural language response according to the dialogue belief states and heuristic rules. Once a dialogue reaches the end, the system receives a big bonus 80, if it successfully helps users book tickets. Otherwise, it receives -40. In each turn, the system receives a fixed reward -1 to encourage shorter dialogues. The maximum number of turns allowed is 40. Our previous work has studied how such rewards affect dialogue behaviors, e.g., higher success bonus and/or lower failure penalty encourage more risk-seeking behaviors

	Failure	Success	
T0-U	Can I get tickets for zootopia?	Can I get tickets for zootopia?	T0-U
T1-S	night is available.	What movie are you interested in?	T1-S
T2-U	Could you help me to book The tickets?	I want to watch zootopia.	T2-U
T3-S	What time would you like to see it?	What time would you like to see it?	T3-S
T4-U	I want to watch at 9:10 pm.	I want to watch at 9:10 pm.	T4-U
T5-S	Which city would you like?	Which city would you like?	T5-S
T6-U	I want to watch at seattle.	I want to watch at seattle.	T6-U
T7-S	What movie are you interested in?	What date would you like to watch it?	T7-S
T8-U	I want to watch zootopia.	I want to set it up tomorrow	T8-U
T9-S	What movie are you interested in?	Which theater would you like?	T9-S
T10-U	I want to watch zootopia.	I want to watch at regal meridian 16.	T10-U
T11-S	What movie are you interested in?	How many tickets do you need?	T11-S
T12-U	I want to watch zootopia.	I want 2 tickets please!	T12-U
T13-S	What movie are you interested in?	I was able to purchase 2 tickets for you to see Zootopia tomorrow at regal Meridian 16 theater in seattle At 9:10 pm.	T13-S
T14-U	I want to watch zootopia.		
T15-S	What movie are you interested in?		
T16-U	I want to watch zootopia.	Thank you	T14-U
T17-S	What movie are you interested in?	Thank you	T15-S
T18-U	I want to watch zootopia.		
T19-S	What movie are you interested in?		
	...		

Figure 3: Examples of an unsuccessful dialogue and a successful dialogue, where we use "U" and "S" to indicate *user* and *system* turns respectively.

and higher QA costs result in less accurate, shorter conversations (Zhang and Stone 2015).

The DQN architecture used in this paper is a single-layer neural network of size 80. Its output layer has 11 units corresponding to the 11 dialogue actions. The techniques of target network and experience replay are applied (see Section). The size of experience pool is $100k$, and experience replay strategy is uniform sampling. The value of α in Equation 2 is 1.0, and $\epsilon-greedy$ policy is used, where ϵ is initialized with 0.3, and decayed to 0.01 during training.

Each experiment includes 1000 epochs. Each epoch includes 100 dialogue episodes. By the end of each epoch, we update the weights of target network using the current behavior network, and this update operation executes once every epoch. Every data point in the figures is an average of 500 dialogues from 5 runs (100 dialogues in each run). For instance, a data point of 0.6 success rate at episode 70 means that 60% of the 70th dialogue episodes (500 in total) were successful. In line with previous research (Su et al. 2016a; Peng et al. 2017), we use supervised learning to give our dialogue agent a "warm start" in each set of experiments, unless specified otherwise.

Case Illustration

Before presenting statistical results, we use two example dialogues to illustrate how T-HER and S-HER augment dialogue data, as shown in Figure 3. The goal of the unsuccessful dialogue (Left) includes three constraints of *start time*, *city*, and *movie name*, and one request of *ticket*. The goal of successful dialogue (Right) includes six constraints of *start time*, *city*, *movie name*, *theater*, *date*, and *number of people*, and one request of *ticket*. Next, we demonstrate how T-HER and S-HER use the two dialogues to generate artificial, successful dialogues.

T-HER selects valid dialogue segments according to achieved subgoals. On the left, the dialogue segment in blue

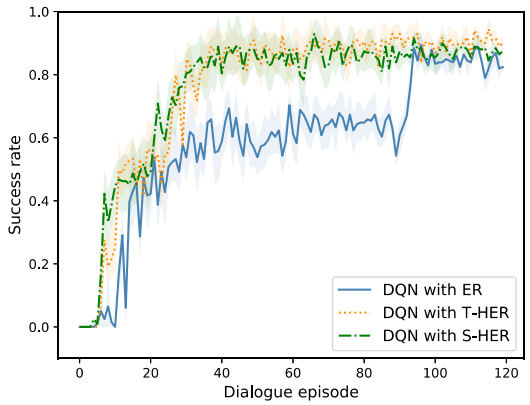

Figure 4: Learning to dialogue to accomplish tasks of booking movie tickets. Our two complex HER strategies improve the learning rate, c.f., naive DQN with standard experience reply (ER).

color achieves the subgoal that includes three constraints (*start time*, *city*, and *movie name*), and also its "subsubgoals". Accordingly, T-HER can generate three valid dialogue segments (new, successful dialogues), including the ones from T_0 to T_4, from T_0 to T_5, and from T_0 to T_8. Turns after T_8 are not considered, because our assessment function does not allow the agent asking useless questions.

S-HER generates new successful dialogues by stitching head and tail dialogue segments. In this example, the two dialogue segments in blue color achieved the same subgoal, although the two dialogue segments have very different flows. S-HER enables our agent to generate a new successful dialogue by stitching the dialogue segment in blue color on the left, and the dialogue segment in green color on the right. The newly generated, successful dialogue achieves the goal that is originally from the dialogue on the right.

Experimental Results

Experiments have been extensively conducted to evaluate the following hypotheses. I) Our HER methods perform better than standard experience reply; II) Our HER methods can be combined with prioritized ER for better performance; III) Our HER methods perform better than existing ER methods in cold start; and IV) T-HER and S-HER can be combined to produce the best performance.

DQN with our HER strategies Since the original HER method (Andrychowicz et al. 2017) is not applicable to dialogue systems, we compare our complex HER methods with a standard DQN-based reinforcement learner. Figure 4 shows that both T-HER and S-HER significantly accelerate the training process, which supports our key hypothesis. The KL divergence threshold of S-HER is 0.2 in this experiment. Next, we study the effect this parameter to the performance of S-HER.

KL divergence level of S-HER Stitching is allowed in S-HER, only if the KL divergence between two connecting states is below a "stitchability" threshold (details in Algorithm 2). Intuitively, if the threshold is too small, there will

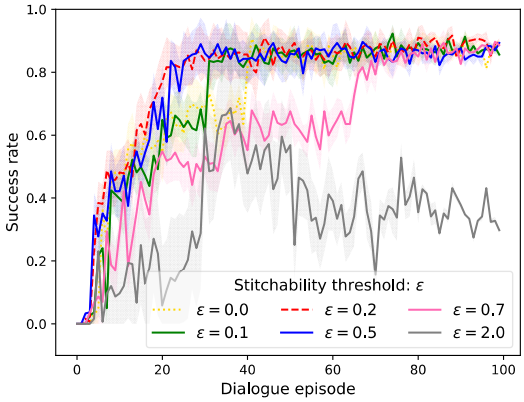

Figure 5: *Stitchability* in S-HER: KL divergence threshold ε in [0.2, 0.5] performs the best.

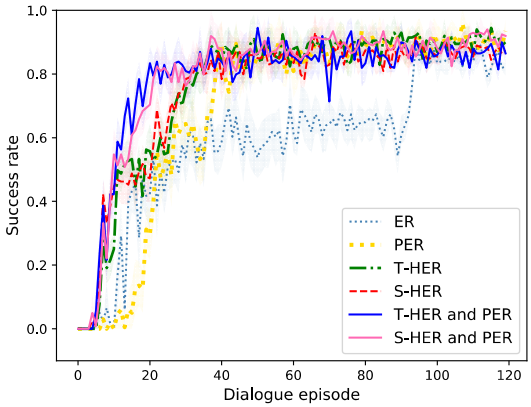

Figure 6: Incorporating prioritized experience replay to improve the performances of T-HER and S-HER.

be very few dialogue segments being stitched (though the newly generated dialogues are of very good quality). If the threshold is too big, many dialogue segments are stitched, but the quality of generated, artificial dialogues will be poor, producing negative effects to the learning process.

Figure 5 depicts the influences of different thresholds. As expected, when the threshold is too high (e.g., $\varepsilon = 2.0$), the RL agent failed to converge to a good policy; when the threshold is too small (e.g., $<= 0.1$), the improvement to the learning rate is not as good as using a threshold within the range of $[0.2, 0.5]$. Results here can serve as a reference to S-HER practitioners.

Integration with PER The first two sets of experiments used uniform sampling in experience replay. We further evaluate the performance of integrating our complex HER methods with prioritized experience replay (PER) that prioritizes selecting the potentially-valuable samples (in terms of TD-error) (Schaul et al. 2015). Figure 6 shows the results. We can see T-HER and S-HER perform better than PER-only, and that incorporating PER into T-HER and S-HER further improved the performances. It should be noted that all five learning strategies converge to policies that are of similar qualities.

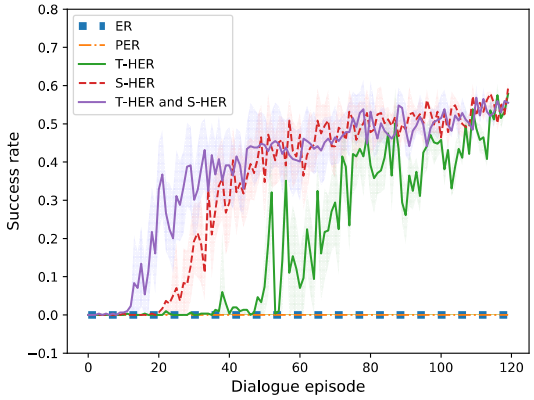

Figure 7: In extreme situations (cold start and small experience pool), S-HER performs better than T-HER, and their combination performs the best.

Learning with a cold start T-HER and S-HER produced similar results in previous experiments. In this set of experiments, we removed the warm-start policy (from supervised learning) that has been widely used in the literature (Li et al. 2017; Lipton et al. 2017), and reduced the experience replay pool from $100k$ to $1k$, resulting in an extremely challenging task to the dialogue learners.

Figure 7 shows the results. We can see that uniform experience replay (ER) (Mnih et al. 2015), and PER (Schaul et al. 2015) could not accomplish any task. Under such challenging settings, our complex HER methods achieved > 0.5 success rates. In particular, S-HER outperforms T-HER, by generating significantly more successful dialogues. Finally, combining T-HER and S-HER produced the best performance.

Conclusions and Future Work

In this work, we developed two complex hindsight experience replay (HER) methods, namely Trimming-based HER and Stitching-based HER, for dialogue data augmentation (DDA). Our two HER methods use human-agent dialogues to generate successful, artificial dialogues that are particularly useful for learning when successful dialogues are rare (e.g., in early learning phase). We used a realistic dialogue simulator for experiments. Results suggest that our methods significantly increased the learning rate of a DQN-based reinforcement learner, and incorporating other experience replay methods further improved their performance.

This is the first work that applies the HER idea to the problem of dialogue policy learning. In the future, we plan to evaluate our complex HER strategies using other dialogue simulation platform, e.g., PyDial (Ultes et al. 2017), and other testing environments. Another direction is to evaluate the robustness of T-HER and S-HER by replacing DQN with other RL algorithms, such as Actor Critic (Su et al. 2017). Finally, we will improve this line of research by further considering the noise from language understanding.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China under grant number U1613216 and Double-First Class Foundation of University of Science and Technology of China.

References

Andrychowicz, M.; Wolski, F.; Ray, A.; et al. 2017. Hindsight experience replay. In *Advances in Neural Information Processing Systems*.

Chandramohan, S.; Geist, M.; and Pietquin, O. 2010. Optimizing spoken dialogue management with fitted value iteration. In *Eleventh Annual Conference of the International Speech Communication Association*.

Cuayáhuitl, H. 2017. Simpleds: A simple deep reinforcement learning dialogue system. In *Dialogues with Social Robots*. Springer.

Ferreira, E., and Lefevre, F. 2015. Reinforcement-learning based dialogue system for human–robot interactions with socially-inspired rewards. *Computer Speech & Language*.

Gašić, M., and Young, S. 2014. Gaussian processes for pomdp-based dialogue manager optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Han, S., and Sung, Y. 2017. Multi-batch experience replay for fast convergence of continuous action control.

Lagoudakis, M. G., and Parr, R. 2003. Least-squares policy iteration. *Journal of machine learning research*.

Li, X.; Lipton, Z. C.; Dhingra, B.; et al. 2016. A user simulator for task-completion dialogues.

Li, X.; Chen, Y.-N.; Li, L.; et al. 2017. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Lin, L.-J. 1993. Reinforcement learning for robots using neural networks. Technical report, Carnegie-Mellon Univ Pittsburgh PA School of Computer Science.

Lipton, Z.; Li, X.; Gao, J.; Li, L.; et al. 2017. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; et al. 2015. Human-level control through deep reinforcement learning. *Nature*.

Nair, A.; Srinivasan, P.; Blackwell, S.; Alcicek, C.; et al. 2015. Massively parallel methods for deep reinforcement learning.

Ng, A. Y.; Harada, D.; and Russell, S. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*.

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

Pietquin, O.; Geist, M.; Chandramohan, S.; and Frezza-Buet, H. 2011. Sample-efficient batch reinforcement learning for dialogue management optimization. *ACM Transactions on Speech and Language Processing (TSLP)*.

Schatzmann, J., and Young, S. 2009. The hidden agenda user simulation model. *IEEE transactions on audio, speech, and language processing*.

Schatzmann, J.; Thomson, B.; Weilhammer, K.; et al. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*. Association for Computational Linguistics.

Schaul, T.; Quan, J.; Antonoglou, I.; and Silver, D. 2015. Prioritized experience replay.

Su, P.-H.; Vandyke, D.; Gasic, M.; et al. 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*.

Su, P.-H.; Gasic, M.; Mrksic, N.; et al. 2016a. Continuously learning neural dialogue management.

Su, P.-H.; Gasic, M.; Mrkšić, N.; 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 (Volume 1: Long Papers)*.

Su, P.-H.; Budzianowski, P.; Ultes, S.; Gasic, M.; and Young, S. 2017. Sample-efficient actor-critic reinforcement learning with supervised data for dialogue management. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An introduction*. MIT press Cambridge.

Tang, D.; Li, X.; Gao, J.; et al. 2018. Subgoal discovery for hierarchical dialogue policy learning.

Thomson, B. 2013. *Statistical methods for spoken dialogue management*. Springer Science & Business Media.

Ultes, S.; Rojas Barahona, L. M.; Su, P.-H.; Vandyke, D.; Kim, D.; Casanueva, I. n.; Budzianowski, P.; Mrkšić, N.; Wen, T.-H.; Gasic, M.; and Young, S. 2017. PyDial: A Multi-domain Statistical Dialogue System Toolkit. In *Proceedings of ACL 2017, System Demonstrations*. Association for Computational Linguistics.

Wang, Z.; Bapst, V.; Heess, N.; Mnih, V.; et al. 2016. Sample efficient actor-critic with experience replay.

Williams, J. D., and Zweig, G. 2016. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning.

Young, S.; Gašić, M.; Thomson, B.; and Williams, J. D. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*.

Zhang, S., and Stone, P. 2015. Corpp: commonsense reasoning and probabilistic planning, as applied to dialog with a mobile robot. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*.