# Generalizable Episodic Memory for Deep Reinforcement Learning

Hao Hu [1]  Jianing Ye [2]  Guangxiang Zhu [1]  Zhizhou Ren [3]  Chongjie Zhang [1]

## Abstract

Episodic memory-based methods can rapidly latch onto past successful strategies by a non-parametric memory and improve sample efficiency of traditional reinforcement learning. However, little effort is put into the continuous domain, where a state is never visited twice, and previous episodic methods fail to efficiently aggregate experience across trajectories. To address this problem, we propose **G**eneralizable **E**pisodic **M**emory (**GEM**), which effectively organizes the state-action values of episodic memory in a generalizable manner and supports implicit planning on memorized trajectories. GEM utilizes a double estimator to reduce the overestimation bias induced by value propagation in the planning process. Empirical evaluation shows that our method significantly outperforms existing trajectory-based methods on various MuJoCo continuous control tasks. To further show the general applicability, we evaluate our method on Atari games with discrete action space, which also shows a significant improvement over baseline algorithms.

## 1. Introduction

Deep reinforcement learning (RL) has been tremendously successful in various domains, like classic games (Silver et al., 2016), video games (Mnih et al., 2015), and robotics (Lillicrap et al., 2016). However, it still suffers from high sample complexity, especially compared to human learning (Tsividis et al., 2017). One significant deficit comes from gradient-based bootstrapping, which is incremental and usually very slow (Botvinick et al., 2019).

[1]The Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China [2]Peking University, Beijing, China [3]University of Illinois at Urbana-Champaign, IL, USA. Correspondence to: Chongjie Zhang <chongjiezhang@mail.tsinghua.edu.cn>.
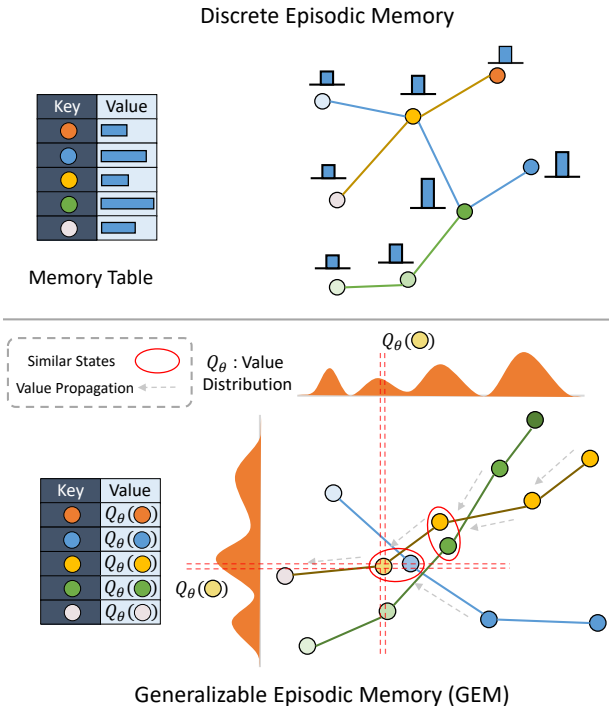
*Figure 1.* Illustration of our basic idea. Conventional episodic memory usually uses a non-parametric schema to store every state-action pairs and updates their values when re-encountering the same events. Instead, generalizable episodic memory stores the $Q$-values for each event by a parametric network $\mathcal{M}_\theta$ so that every single event can generalize to their neighborhoods. We further perform value propagation along adjacent states to encourage information exchange between related events.

Inspired by psychobiological studies of human episodic memory (Sutherland & Rudy, 1989; Marr et al., 1991; Lengyel & Dayan, 2007) and instance-based decision theory (Gilboa & Schmeidler, 1995), episodic reinforcement learning (Blundell et al., 2016; Pritzel et al., 2017; Lin et al., 2018; Hansen et al., 2018; Zhu et al., 2020) presents a non-parametric or semi-parametric framework that fast retrieves past successful strategies to improve sample efficiency. Episodic memory stores past best returns, and the agents can act accordingly to repeat the best outcomes without gradient-based learning.

However, most existing methods update episodic memory only by re-encountered events, leaving the generalization ability aside. As Heraclitus, the Greek philosopher, once said, "No man ever steps in the same river twice." (Kahn et al., 1981) Similarly, for an RL agent acting in continuous action and state spaces, the same state-action pair can hardly be observed twice. Nevertheless, humans can connect and retrospect from similar experiences of different times, with no need to re-encounter the same event (Shohamy & Wagner, 2008). Inspired by this human's ability to learn from generalization, we propose **G**eneralizable **E**pisodic **M**emory (**GEM**), a novel framework that integrates the generalization ability of neural networks and the fast retrieval manner of episodic memory.

We use Figure 1 to illustrate our basic idea. Traditional discrete episodic control methods usually build a non-parametric slot-based memory table to store state-value pairs of historical experiences. In the discrete domain, states can be re-encountered many times. This re-encountering enables traditional methods to aggregate trajectories by directly taking maximum among all historical returns. However, this is not feasible in environments with high dimensional states and action space, especially in the continuous domain, where an agent will never visit the same state twice. On the contrary, GEM learns a virtual memory table memorized by deep neural networks to aggregate similar state-action pairs that essentially have the same nature. This memory organization enables planning across different trajectories, and GEM uses this capability to do implicit planning by performing value propagation along trajectories saved in the memory and calculating the best sequence over all possible real and counterfactual trajectories.

We further propose to use twin networks to reduce the overestimation of these aggregated returns. Merely taking maximum among all possible plans leads to severe overestimation since overestimated values are preserved along the trajectory. Thus, we use two networks to separately calculate which trajectory has the maximal value and how large this maximum value is, which shares a similar idea with double Q-learning (van Hasselt, 2010).

The main contribution of our proposed method is threefold: 1. We present a novel framework inspired by psychology, GEM, to build generalizable episodic memory and improve sample efficiency. GEM consists of a novel value planning scheme for the continuous domain and a twin back-propagation process to reduce overestimation along the trajectories; 2. We formally analyze the value estimation and convergence properties of our proposed algorithm; 3. We empirically show the significant improvement of GEM over other baseline algorithms and its general applicability across continuous and discrete domains. Besides, we analyze the effectiveness of each part of GEM by additional comparisons and ablation studies.

## 2. Preliminaries

We consider a Markov Decision Process (MDP), defined by the tuple $\langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$, where $\mathcal{S}$ is the state space and $\mathcal{A}$ the action space. $P(\cdot|s,a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ denotes the transition distribution function and $r(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function. $\gamma \in [0,1)$ is the discount factor.

The goal of an RL agent is to learn a policy $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ which maximizes the expectation of the discounted cumulative reward, i.e.,

$$J(\pi) = \mathbb{E}_{\substack{s_0 \sim \rho_0, a_t \sim \pi(\cdot|s_t), \\ s_{t+1} \sim P(\cdot|s_t, a_t)}} \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right],$$

where $\rho_0$ denotes the distribution of initial states.

### 2.1. Deterministic Policy Gradient

In the context of continuous control, actor-critic architecture is widely used (Sutton et al., 1999; Peters & Bagnell, 2010) to handle the continuous action space. In actor-critic framework, a critic $Q_\theta$ is used to estimate the action-value function $Q_\theta(s,a) \approx Q^\pi(s,a)$, while the parametric actor $\pi_\phi$ optimizes its policy using the policy gradient from parametric $Q_\theta$. When $\pi_\phi$ is a deterministic function, the gradient can be expressed as follows:

$$\nabla_\phi J(\pi_\phi) = \mathbb{E}_{s \sim p_\pi}[\nabla_a Q_\theta(s,a)|_{a=\pi(s)} \nabla_\phi \pi_\phi(s)],$$

which is known as deterministic policy gradient theorem (Silver et al., 2014).
$Q_\theta$ can be learned from the TD-target derived from the Bellman Equation (Bellman, 1957):

$$y = r(s,a) + \gamma Q_{\theta'}(s',a'), a' \sim \pi_\phi(s')$$

### 2.2. Double Q-Learning

Q-learning update state-action values by the TD-target (Sutton, 1988a) :

$$r(s,a) + \gamma \max_{a'} Q(s',a').$$

However, the max operator in the target can easily lead to overestimation, as discussed in van Hasselt (2010). Thus, they propose to estimate the maximum $Q$-values over all actions $\max_a Q(s,a)$ using the double estimator, i.e.,

$$Q_{\text{double}}(s) = Q^A(s, a_B^*), \quad a_B^* = \arg\max_a Q^B(s,a),$$

where $Q^A, Q^B$ are two independent function approximators for $Q$-values.

## 2.3. Model-Free Episodic Control

Traditional deep RL approaches suffer from sample inefficiency because of slow gradient-based updates. Episodic control uses a non-parametric episodic memory (EM) to avoid gradient-based updates and speed up the learning process. The key idea is to store good past experiences in a tabular-based non-parametric memory and rapidly latch onto past successful policies when encountering similar states instead of waiting for many steps of optimization. When different experiences meet up at the same state-action pair $(s, a)$, model-free episodic control (MFEC; Blundell et al., 2016) aggregates the values of different trajectories by taking the maximum return $R$ among all these rollouts. Specifically, $Q^{\text{EM}}$ is updated by the following equation:

$$Q^{\text{EM}}(s, a) = \begin{cases} R, & \text{if } (s, a) \notin \text{EM}, \\ \max\{R, Q^{\text{EM}}(s, a)\}, & \text{otherwise.} \end{cases} \tag{1}$$

At execution time, MFEC selects the action that maximizes $Q^{\text{EM}}$ of the current state. If there is no exact match of the state, MFEC performs a $k$-nearest-neighbors lookup to estimate the state-action values, i.e.,

$$\widehat{Q}^{EM}(s, a) = \begin{cases} \dfrac{1}{k} \sum_{i=1}^{k} Q(s_i, a), & \text{if } (s, a) \notin \text{EM}, \\ Q^{\text{EM}}(s, a), & \text{otherwise,} \end{cases} \tag{2}$$

where $s_i$ $(i = 1, \cdots, k)$ are the $k$ nearest states from $s$.

## 3. Generalizable Episodic Memory

### 3.1. Overview

In generalizable episodic memory (GEM), we use a parametric network $\mathcal{M}_\theta$ to represent the virtual memory table with parameter $\theta$, which is learned from tabular memory $\mathcal{M}$. To leverage the generalization ability of $\mathcal{M}_\theta$, we enhance the returns stored in the tabular memory by propagating value estimates from $\mathcal{M}_\theta$ and real returns from $\mathcal{M}$ along trajectories in the memory and take the best value over all possible rollouts, as depicted in Equation (4). Generalizable memory $\mathcal{M}_\theta$ is trained by performing regression toward this enhanced target and is then used to guide policy learning as well as to build the new target for GEM's learning.

A significant issue for the procedure above is the overestimation induced from taking the best value along the trajectories. Overestimated values are preserved when doing back-propagation along the trajectory and hinder learning efficiency. To mitigate this issue, we propose to use twin networks for the back-propagation of value estimates. This twin back-propagation process uses the double estimator for estimating the maximum along trajectories, which resembles the idea of Double Q-learning (van Hasselt, 2010)

---

**Algorithm 1** Generalizable Episodic Memory

Initialize episodic memory(critic) networks $Q_\theta^{(1)}, Q_\theta^{(2)}$, and actor network $\pi_\phi$ with parameters $\theta_{(1)}, \theta_{(2)}, \phi$
Initialize targets $\theta'_{(1)} \leftarrow \theta_{(1)}, \theta'_{(2)} \leftarrow \theta_{(2)}, \phi' \leftarrow \phi$
Initialize episodic memory $\mathcal{M}$
**for** $t = 1, \ldots, T$ **do**
    Select action with noise $a \sim \pi(s) + \mathcal{N}(0, \sigma)$
    Observe reward $r$ and new state $s'$
    Store transition tuple $(s, a, r, s')$ in $\mathcal{M}$
    **for** $i \in \{1, 2\}$ **do**
        Sample $N$ transitions $(s_t, a_t, r_t, s'_t, R_t^{(i)})$ from $\mathcal{M}$
        Update $\theta_{(i)} \leftarrow \min_{\theta_{(i)}} \sum (R_t^{(i)} - Q_\theta^{(i)}(s_t, a_t))^2$
    **end for**
    **if** $t \bmod u = 0$ **then**
        $\phi' \leftarrow \tau\phi + (1 - \tau)\phi'$
        $\theta'_{(i)} \leftarrow \tau\theta_{(i)} + (1 - \tau)\theta'_{(i)}$
        UPDATE MEMORY()
    **end if**
    **if** $t \bmod p = 0$ **then**
        Update $\phi$ by the deterministic policy gradient
        $\nabla_\phi J(\phi) = \nabla_a Q_{\theta_1}(s, a)|_{a = \pi_\phi(s)} \nabla_\phi \pi_\phi(s)$
    **end if**
**end for**

---

**Algorithm 2** Update Memory

**for** trajectory $\tau$ in tabular memory $\mathcal{M}$ **do**
    **for** $s_t, a_t, r_t, s_{t+1}$ in REVERSED$(\tau)$ **do**
        $\tilde{a}_{t+1} \sim \pi_{\phi'}(s_{t+1}) + \text{clip}(\mathcal{N}(0, \tilde{\sigma}), -c, c)$
        Compute $Q_{\theta'_{(1,2)}}(s_{t+1}, \tilde{a}_{t+1})$
        Compute $V_{t,h}^{(1,2)}$ with Equation (7) for $h$ in $0 : T - t$
        Compute $R_t^{(1,2)}$ with Equation (8) and save into buffer $\mathcal{M}$
    **end for**
**end for**

---

and is illustrated in detail in Section 3.4. It is well known that vanilla reinforcement learning algorithms with function approximation already has a tendency to overestimate (van Hasselt, 2010; Fujimoto et al., 2018), making the overestimation reduction even more critical. To solve this challenge, we propose to use additional techniques to make the value estimation from $\mathcal{M}_\theta$ more conservative, which is illustrated in Section 3.5.

A formal description for the GEM algorithm is shown in Algorithm 1. In the following sections, we use $Q_\theta$ to represent $Q_{\mathcal{M}_\theta}$ for simplicity. Our implementation of GEM is available at https://github.com/MouseHu/GEM.

## 3.2. Generalizable Episodic Memory

Traditional discrete episodic memory is stored in a lookup table, learned as in Equation (1) and used as in Equation (2). This kind of method does not consider generalization when learning values and enjoys little generalization ability from the non-parametric nearest-neighbor search during execution. To enable the generalizability of such episodic memory, we use a parametric network $\mathcal{M}_\theta$ to learn and store values. As stated in Section 3.1, this parametric memory is learned by doing regression toward the best returns $R_t$ starting from the state-action pair $(s_t, a_t)$:

$$\mathcal{L}(Q_\theta) = \mathbb{E}_{(s_t,a_t,R_t)\sim\mathcal{M}}(Q_\theta(s_t, a_t) - R_t)^2. \quad (3)$$

where $R_t$ is computed from implicit planning over both real returns in tabular memory $\mathcal{M}$ and estimates in target generalizable memory table $\mathcal{M}'_\theta$, as depicted in the next section. This learned virtual memory table is then used for learning policy and building a new target to estimate $R_t$.

## 3.3. Implicit Memory-Based Planning

With the generalization ability of our parametric memory, GEM conducts implicit memory-based planning to estimate the value for the best possible rollout for each state-action pair. At each step, GEM compares the best return so far along the trajectory with the value estimates from $Q_\theta$ and takes the maximum between them. $Q_\theta$ is generalized from similar experiences and can be regarded as the value estimates for counterfactual trajectories. This procedure is conducted recursively from the last step to the first step along the trajectory, forming an implicit planning scheme within episodic memory to aggregate experiences along and across trajectories. The overall back-propagation process can be written in the following form:

$$R_t = \begin{cases} r_t + \gamma \max(R_{t+1}, Q_\theta(s_{t+1}, a_{t+1})), & \text{if } t < T, \\ r_t, & \text{if } t = T, \end{cases} \quad (4)$$

where $t$ denotes steps along the trajectory and $T$ the episode length. Further, the back-propagation process in Equation (4) can be unrolled and rewritten as follows:

$$V_{t,h} = \begin{cases} r_t + \gamma V_{t+1,h-1}, & \text{if } h > 0, \\ Q_\theta(s_t, a_t), & \text{if } h = 0, \end{cases}$$
$$R_t = V_{t,h^*}, h^* = \arg\max_{h>0} V_{t,h}, \quad (5)$$

where $h$ denotes different length of rollout steps, and we define $Q_\theta(s_t, a_t) = 0$ and $V_{t,h} = 0$ for $t > T$.

## 3.4. Twin Back-Propagation Process

In this section, we describe our novel twin back-propagation process to reduce planning-induced overestimation.

As mentioned in Section 3.1, using Equation (5) directly for planning can lead to severe overestimation, even if $Q_\theta$ is unbiased. Formally, for any unbiased set of estimators $\tilde{Q}_h(s,a) = Q_h(s,a) + U_h(s,a)$, where $Q_h(s,a)$ is the true value and $U_h(s,a)$ is a set of independent, zero-mean random noise, we have

$$\mathbb{E}_U\left[\max_h \tilde{Q}_h(s,a)\right] \geq \max_h Q_h(s,a), \quad (6)$$

which can be derived directly from Jensen's inequality.

Thus trajectory-augmented values have a tendency to overestimate and this overestimation bias can hurt performance greatly. We propose to use twin networks, $Q^{(1)}, Q^{(2)}$, to form a double estimator for estimating the best value $R_t$ in the value back-propagation in Equation (5). One network is used to estimate the length of rollouts with the maximum returns along a trajectory ($h^*$ in Equation (5)), while the other estimates how large the return is following this estimated best rollout ($V_{t,h^*}$ in Equation (5)). Formally, the proposed twin back-propagation process (TBP) is given by [1]:

$$V_{t,h}^{(1,2)} = \begin{cases} r_t + \gamma V_{t+1,h-1}^{(1,2)}, & \text{if } h > 0, \\ Q_\theta^{(1,2)}(s_t, a_t), & \text{if } h = 0, \end{cases} \quad (7)$$
$$R_t^{(1,2)} = V_{t,h^*_{(1,2)}}^{(2,1)}, h^*_{(1,2)} = \arg\max_{h>0} V_{t,h}^{(1,2)}. \quad (8)$$

And each $Q$-network is updated by

$$\mathcal{L}(\theta_{(1,2)}) = \mathbb{E}_{s_t,a_t,R_t\sim\mathcal{M}}\left(Q_\theta^{(1,2)}(s_t, a_t) - R_t^{(1,2)}\right)^2.$$

Note that this twin mechanism is different from double Q-learning, where we take the maximum over timesteps while double Q-learning takes the maximum over actions.

Formal analysis in Section 4.1 shows that this twin back-propagation process does not overestimate the true maximum expectation value, given unbiased $Q_\theta$. As stated in van Hasselt (2010) and Fujimoto et al. (2018), while this update rule may introduce an underestimation bias, it is much better than overestimation, which propagates along trajectories.

## 3.5. Conservative Estimation on Single Step

The objective in Equation (8) may still be prone to overestimation since $Q_\theta^{(1,2)}$ is not necessarily unbiased, and it is well known that deterministic policy gradient tends to overestimate true values (Fujimoto et al., 2018). To address this issue, we further augment each $Q_\theta^{(1,2)}$ functions with two independent networks $Q_A, Q_B$, and use the clipped

---

[1]In Equation (7) & (8), index $(1, 2)$ represents it can be either $(1)$ or $(2)$, while reversed index $(2, 1)$ refers to the other network's estimation, i.e., represents $(2)$ when $(1, 2)$ refers to $(1)$ and vice versa.

double Q-learning objective as in Fujimoto et al. (2018) for the estimation at each step:

$$Q^{(1,2)}(s,a) = \min_{A,B} \left( Q_A^{(1,2)}(s,a), Q_B^{(1,2)}(s,a) \right). \quad (9)$$

Finally, we propose the following asymmetric objective to penalize overestimated approximation error,

$$\mathcal{L}(\theta_{(1,2)}) = \mathbb{E}_{s_t,a_t,R_t \sim \mathcal{M}}[(\delta_t)_+^2 + \alpha(-\delta_t)_+^2], \quad (10)$$

where $\delta_t = Q^{(1,2)}(s_t, a_t) - R_t^{(1,2)}$ is the regression error and $(\cdot)_+ = \max(\cdot, 0)$. $\alpha$ is the hyperparameter to control the degree of asymmetry.

## 4. Theoretical Analysis

In this section, we aim to establish a theoretical characterization of our proposed approach through several aspects. We begin by showing an important property that the twin back-propagation process itself does not have an incentive to overestimate values, the same property as guaranteed by double Q-learning. In addition, we prove that our approach guarantees to converge to the optimal solution in deterministic environments. Moreover, in stochastic environments, the performance could be bounded by a dependency term regrading the environment stochasticity.

### 4.1. Non-Overestimation Property

We first investigate the algorithmic property of the twin back-propagation process in terms of *value estimation bias*, which is an essential concept for value-based methods (Thrun & Schwartz, 1993; van Hasselt, 2010; Fujimoto et al., 2018). Theorem 1 indicates that our method would not overestimate the real maximum value in expectation.

**Theorem 1.** Given unbiased and independent estimators $\tilde{Q}_\theta^{(1,2)}(s_{t+h}, a_{t+h}) = Q^\pi(s_{t+h}, a_{t+h}) + \epsilon_h^{(1,2)}$, Equation (8) will not overestimate the true objective, i.e.,

$$\mathbb{E}_{\tau,\epsilon}\left[ R_t^{(1,2)}(s_t) \right] \leq \mathbb{E}_\tau\left[ \max_{0 \leq h < T-t} Q_{t,h}^\pi(s_t, a_t) \right], \quad (11)$$

where $Q_{t,h}^\pi(s,a) =$

$$\sum_{i=0}^{h} \gamma^i r_{t+i} + \begin{cases} \gamma^{h+1} Q^\pi(s_{t+h+1}, a_{t+h+1}), & \text{if } h < T - t, \\ 0, & \text{if } h = T - t. \end{cases} \quad (12)$$

and $\tau = \{(s_t, a_t, r_t, s_{t+1})_{t=1,\cdots,T}\}$ is a trajectory.

The proof of Theorem 1 is deferred to Appendix B. As revealed by Theorem 1, the twin back-propagation process maintains the same non-overestimation nature of double Q-learning (van Hasselt, 2010), which ensures the reliability of our proposed value propagation mechanism.

### 4.2. Convergence Property

In addition to the statistical property of value estimation, we also analyze the convergence behavior of GEM. Following the same environment assumptions as related work (Blundell et al., 2016; Zhu et al., 2020), We first derive the convergence guarantee of GEM in deterministic scenarios as the following statement.

**Theorem 2.** In a finite MDP with a discount factor $\gamma < 1$, the tabular parameterization of Algorithm 1 would converge to $Q^*$ w.p.1 under the following conditions:

1. $\sum_t \alpha_t(s,a) = \infty, \sum_t \alpha_t^2(s,a) < \infty$

2. The transition function of the given environment is fully deterministic, i.e., $P(s'|s,a) = \delta(s' = f(s,a))$ for some deterministic transition function $f$

where $\alpha_t \in (0,1)$ denotes the scheduling of learning rates. A formal description for tabular parameterization of Algorithm 1 is included in Appendix A.

The proof of Theorem 2 is extended based on van Hasselt (2010), and details are deferred to Appendix B. Note that this theorem only applies to deterministic scenarios, which is a common assumption for memory-based algorithms (Blundell et al., 2016; Zhu et al., 2020). To establish a more precise characterization, we consider a more general class of MDPs, named near-deterministic MDPs, as stated in Definition 4.1.

**Definition 4.1.** We define $Q_{\max}(s_0, a_0)$ as the maximum value possible to receive starting from $(s_0, a_0)$, i.e.,

$$Q_{\max}(s_0, a_0) := \max_{\substack{(s_1,\cdots,s_T),(a_1,\cdots,a_T) \\ s_{i+1} \in supp(P(\cdot|s_i,a_i))}} \sum_{t=0}^{T} \gamma^t r(s_t, a_t).$$

An MDP is said to be nearly-deterministic with parameter $\mu$, if $\forall s \in \mathcal{S}, a \in \mathcal{A}$,

$$Q_{\max}(s,a) \leq Q^*(s,a) + \mu,$$

where $\mu$ is a dependency threshold to bound the stochasticity of environments.

Based on the definition of near-deterministic MDPs, we formalize the performance guarantee of our approach as the following statements:

**Lemma 1.** The value function $Q(s,a)$ learned by the tabular variant of Algorithm 1 satisfies the following inequality:

$$\forall s \in \mathcal{S}, a \in \mathcal{A}, Q^*(s,a) \leq Q(s,a) \leq Q_{\max}(s,a),$$

w.p.1, under condition 1 in Theorem 2.

*Proof Sketch.* We only need to show that $\|(Q - Q^*)_+\|_\infty$ is a $\gamma$-contraction and also $\|(Q_{\max} - Q)_+\|_\infty$. The rest of the proof is similar to Theorem 2. □

**Theorem 3.** For a nearly-deterministic environment with factor $\mu$, GEM's performance can be bounded w.p.1 by

$$V^{\tilde\pi}(s) \geq V^*(s) - \frac{2\mu}{1 - \gamma}, \forall s \in \mathcal{S}.$$

The complete proof of these statements are deferred to Appendix B. Theorem 3 ensures that, our approach is applicable to near-deterministic environments as most real-world scenarios.

# 5. Experiments

Our experimental evaluation aims to answer the following questions: (1) How well does GEM perform on the continuous state and action space? (2) How well does GEM perform on discrete domains? (3) How effective is each part of GEM?

## 5.1. Evaluation on Continuous Control Tasks

We compare our method with the most popular model-free RL algorithms, including DDPG (Lillicrap et al., 2016), TD3 (Fujimoto et al., 2018) and SAC (Haarnoja et al., 2018). Conventional episodic RL such as MFEC (Blundell et al., 2016), NEC (Pritzel et al., 2017), EMDQN (Lin et al., 2018), EVA (Hansen et al., 2018) and ERLAM (Zhu et al., 2020) adopts discrete episodic memory that is only designed for discrete action space, and we cannot directly compare with them. To compare with episodic memory-based approaches on the continuous domain, we instead choose self-imitation learning (SIL; Oh et al., 2018) in our experiments. It also aims to exploit past good experiences and share a similar idea with episodic control; thus, it can be regarded as a continuous version of episodic control. For a fair comparison, We use the same code base for all algorithms, and we combine the self-limitation learning objective with techniques in TD3 (Fujimoto et al., 2018), rather than use the origin implementation, which combines with PPO (Schulman et al., 2017).

We conduct experiments on the suite of MuJoCo tasks (Todorov et al., 2012), with OpenAI Gym interface (Brockman et al., 2016). We truncate the maximum steps available for planning to reduce the overestimation bias. The memory update frequency $u$ is set to 100 with a smoothing coefficient $\tau = 0.6$. The rest of the hyperparameters are mostly kept the same as in TD3 to ensure a fair comparison. The detailed hyperparameters used are listed in Appendix C.

The learning curve on different continuous tasks is shown in Figure 2. We report the performance of 1 million steps,

which is evaluated with 10 rollouts for every 10000 steps with deterministic policies. As the results suggest, our method significantly outperforms other baseline algorithms on most tasks. Only on Hopper, GEM is not the absolute best, but all the algorithms have similar performance.

## 5.2. Evaluation on Discrete Domains

Although GEM is proposed to facilitate continuous control tasks, it also has the general applicability of boosting learning on discrete domains. To demonstrate this, we compare GEM with several deep Q-learning algorithms for the discrete action space, including DQN (Mnih et al., 2015), DDQN (van Hasselt et al., 2016), and Dueling DQN (Wang et al., 2016). We also include the clipped double DQN, which is adopted from a state-of-the-art algorithm of the continuous domain, TD3 (Fujimoto et al., 2018). We evaluate all the above algorithms on 6 Atari games (Bellemare et al., 2013). We use hyper-parameters suggested in Rainbow (Hessel et al., 2018), and other hyper-parameters for our algorithm are kept the same as in the continuous domain. As shown in Figure 3, although not tailored for the discrete domain, GEM significantly outperforms baseline algorithms both in terms of sample efficiency and final performance.

## 5.3. Ablation Study

This section aims to understand each part's contribution to our proposed algorithm, including the generalizable episodic memory, implicit memory-based planning, and twin back-propagation. We also empirically demonstrate the effect of overestimation. To check whether our method, which uses four networks, benefits from the ensembling effect, we also compare our method with the random ensembling method as used in REDQ (Chen et al., 2021).

To verify the effectiveness of generalizable episodic memory, we compare our algorithm with SIL, which directly uses historical returns (which can be seen as a discrete episodic memory). As shown in both Figure 2 and Figure 4, although SIL improves over TD3, it only has marginal improvement. On the contrary, our algorithm improves significantly over TD3.

To understand the contribution of implicit planning, we compare with a variant of our method that uses $\lambda$-return rather than taking the maximum over different steps. We also compare our method with Optimality Tightening (He et al., 2017), which shares the similar idea of using trajectory-based information. As Figure 4 shows, Our method outperforms both methods, which verifies the effectiveness of implicit planning within memories.

To check the contribution of our twin networks and the conservative estimation, we compare the results without the twin networks and the results without asymmetric losses.
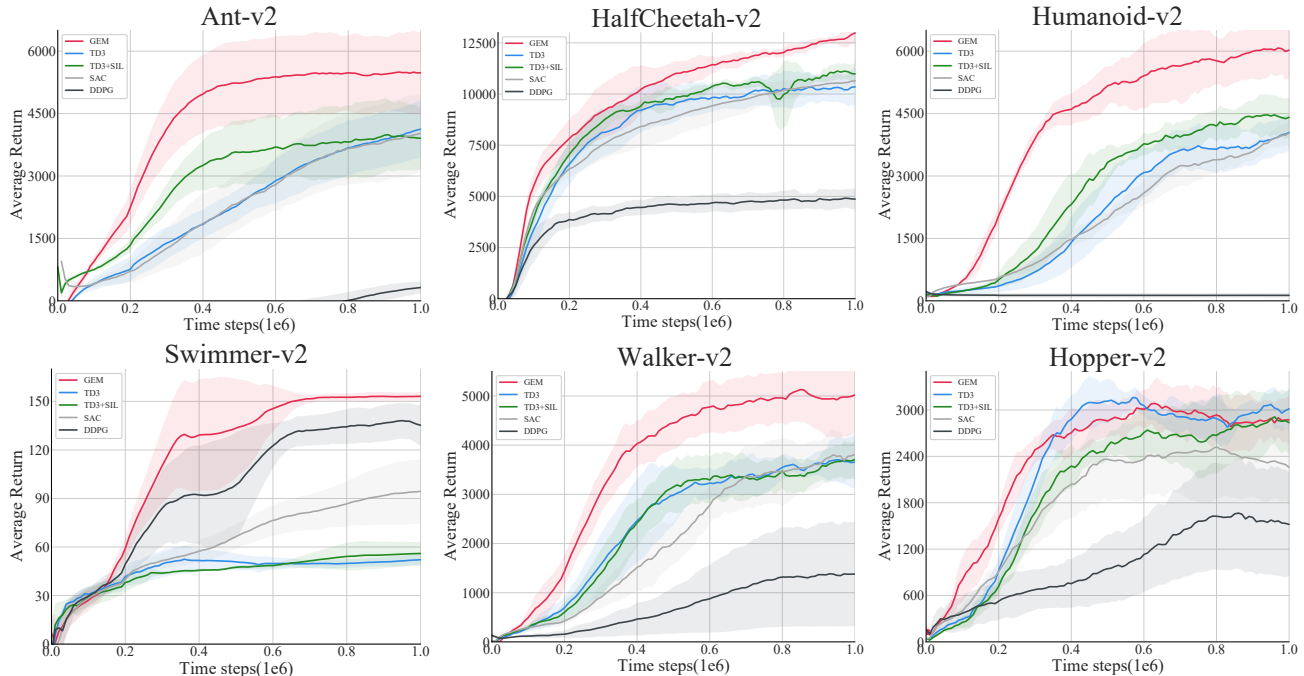
*Figure 2.* Learning curves on MuJoCo tasks, compared with baseline algorithms. The shaded region represents the standard deviation of the performance. Each curve is averaged over five seeds and is smoothed for visual clarity.

The results and the estimation errors are summarized in Figure 4. We can see that overestimation is severe without TBP or conservative estimation, and the performance is greatly affected, especially in games like Humanoid. In Humanoid, the living condition for the agent is strict; thus overestimation is more severely punished.

To understand the contribution of the ensembling effect in our method, we compare our result with random ensembling as used in Chen et al. (2021), since we are not able to directly remove the ensembling effect of TBP. We use the same update-to-data ratio as in GEM for a fair comparison. As Figure 4 shows, naive ensembling contributes little to performance, while overestimation reduction contributes much more than the ensembling effect.

From these comparisons, we can conclude that our generalizable memory with implicit planning aggregates return values much more effectively than directly using historical returns, and the twin networks and conservative estimation contribute to the performance of GEM significantly. In addition, we verify that the merits of the twin back-propagation process mainly come from overestimation reduction rather than ensembling.

## 6. Related Work

**Episodic Control.** Our method is closely related to episodic reinforcement learning, pioneered by Blundell et al. (2016), which introduces the concept of episodic memory into reinforcement learning and uses a non-parametric $k$-NN search to recall past successful experiences quickly. Pritzel et al. (2017) and Lin et al. (2018) consider to include a parametric counterpart for episodic memory, which enables better generalizability through function approximators. Although they provide some level of generalization ability, the aggregation of different trajectories still relies on the exact re-encountering, and their methods cannot handle continuous action space. Several advanced variants (Zhu et al., 2020; Lee et al., 2019) propose several ways to connect different experiences within the episodic memory to improve learning speed further. However, their methods are still not generally applicable in the continuous domain since they require taking maximum among all possible actions or require exact re-encountering. Hansen et al. (2018) adopt the similar idea of soft aggregation as ours, but they only consider decision time and leave the ability of learning by analogy aside. Self-imitation learning (Oh et al., 2018), another memory-based approach built upon policy gradient methods, uses a return-based lower bound to restrict the value function, which cannot aggregate associative trajectories effectively.
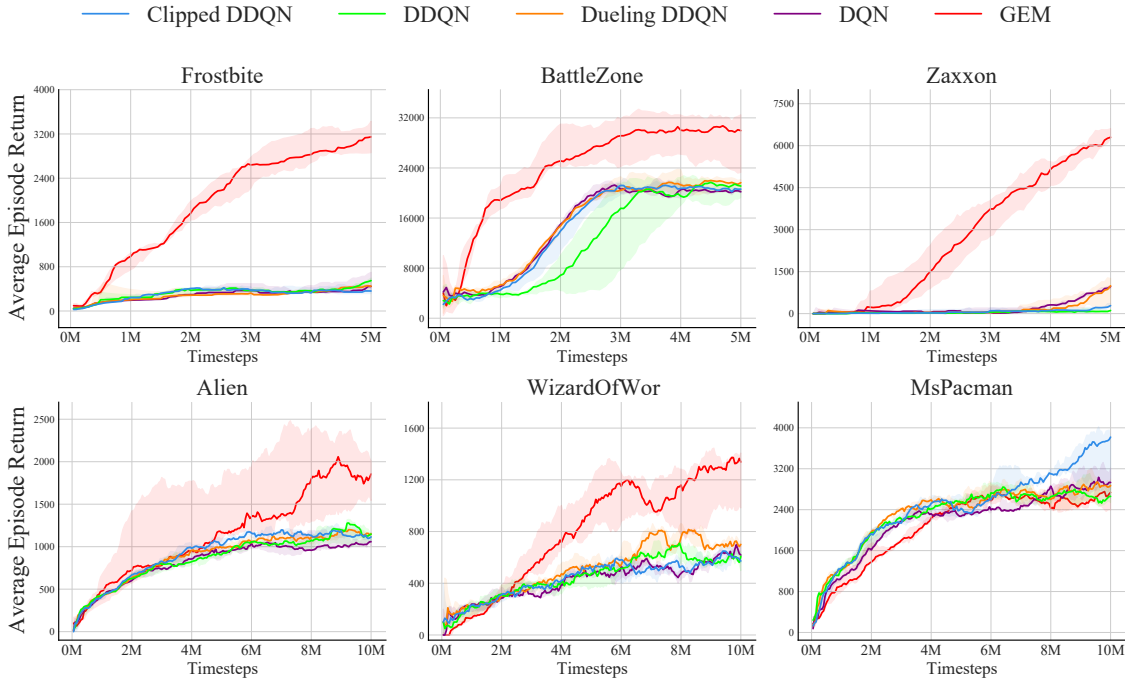
*Figure 3.* Performance comparison on six Atari games supported by OpenAI gym. Each curve is averaged over three seeds. The shaded region represents the standard deviation of the performance.

**From Discrete to Continuous.** Researchers have long been attracted to the challenge of making the RL algorithm applicable in the general continuous case. Kernel-based reinforcement learning (Ormoneit & Sen, 2002) proposes a way to overcomes the stability problems of temporal-difference learning in continuous state-spaces. Lillicrap et al. (2016) extends the success of deep Q-learning into the continuous domain, which successes on various MuJoCo tasks. Similarly, in episodic RL, GEM generalizes the idea of a discrete episodic memory by representing the episodic memory as a network-generated virtual table, making it generally applicable in the continuous case.

**Maximization Bias in Q-Learning.** The maximization bias of Q-learning, first highlighted by Thrun & Schwartz (1993), is a long-lasting issue that hinders the learning efficiency of value-based methods. van Hasselt (2010) proposed to use a second value estimator as cross-validation to address this bias, and this technique has been extended to the deep Q-learning paradigm (van Hasselt et al., 2016). In practice, since it is intractable to construct two fully independent value estimators, double Q-learning is observed to overestimate sometimes, especially in continuous domains. To address this concern, Fujimoto et al. (2018) proposed clipped double Q-learning to repress further the incentive of overestimation, which has become the default implementation of most advanced approaches (Haarnoja et al., 2018;

Kalashnikov et al., 2018). Recently, to control the estimation bias more precisely, people utilize an ensemble of value networks (Lan et al., 2020; Kuznetsov et al., 2020; Chen et al., 2021), which causes high computational costs.

**Multi-step Bootstrapping.** The implementation of our proposed algorithm is also related to a technique named multi-step bootstrapping. This branch originates from the idea of eligibility traces (Singh & Sutton, 1996; Sutton, 1988b). Recently, it is prevalent in policy gradient methods such as GAE (Schulman et al., 2016) and its variants (Touati et al., 2018). In the context of temporal-difference learning, multi-step bootstrapping is also beneficial to improving the stability of Q-learning algorithms (Van Hasselt et al., 2018). The main limitation of multi-step bootstrapping is that the performance of multi-step training is sensitive to the bootstrapping horizon choice, and the trajectory used is required to be at least nearly on-policy rather than an arbitrary one.

## 7. Conclusion

This work presents Generalizable Episodic Memory, an effective memory-based method that learns state-action values from both similar states and future consequences. GEM makes episodic memory generalizable by using a neural network to memorize discrete memory tables. We perform implicit planning with this generalizable memory by taking
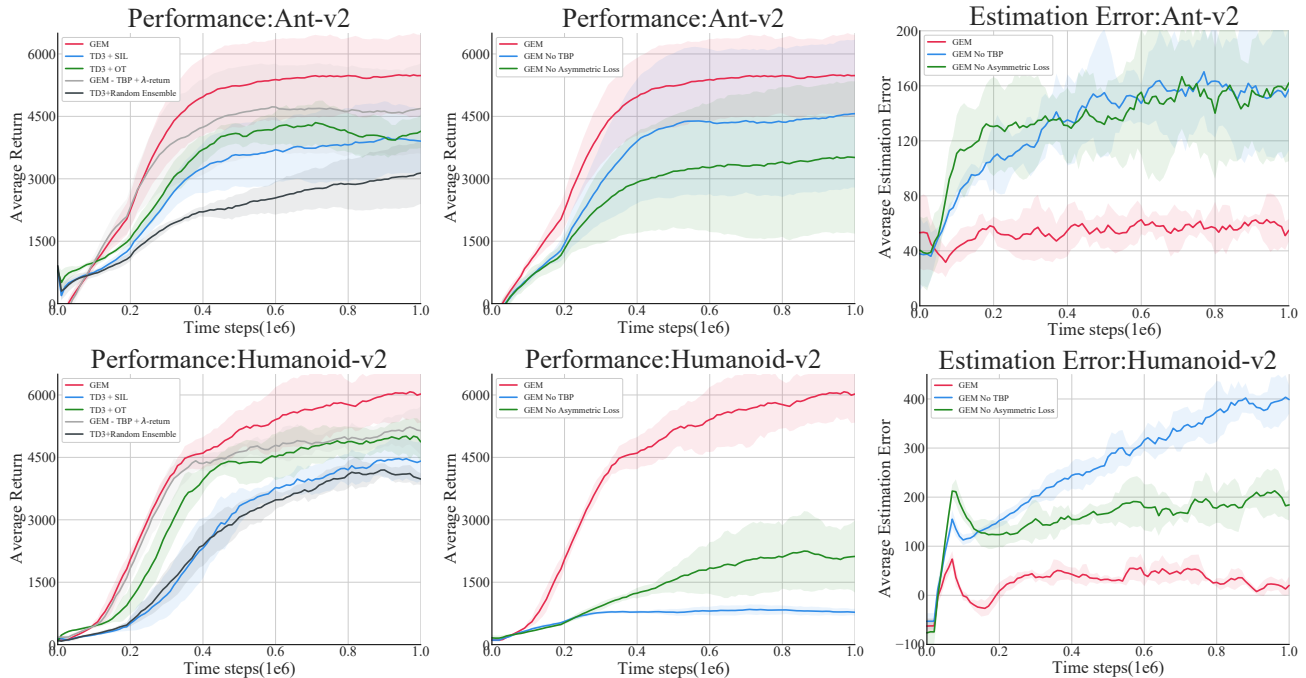
*Figure 4.* Addtional comparison, ablation study of GEM and empirical evaluation for overestimation. The shaded region represents the standard deviation of the performance. Each curve is averaged over five seeds and is smoothed for visual clarity. Estimation error refers to the average estimated Q-values minus the average returns.

the maximum value over all possible real and counterfactual trajectories. Twin networks are used to reduce the overestimation bias induced by implicit planning.

We provide theoretical analysis to show that our objective does not overestimate in general, and our algorithm converges to $Q^*$ under mild conditions. Experimental results on continuous control tasks show that our method outperforms popular model-free RL methods, as well as other episodic memory-based algorithms. Our method also demonstrates general applicability in the discrete domain, such as Atari games. Our method is not primarily designed for discrete domains but can still significantly improve the learning efficiency of RL agents under this setting.

We make the first step to endow episodic memory with generalization ability. This generalization ability also relies highly on the representation of states and actions. We leave the study of representation learning in episodic memory as interesting future work.

## 8. Acknowledgement

## References

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bellman, R. Dynamic programming princeton university press princeton. *New Jersey Google Scholar*, 1957.

Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J. Z., Rae, J., Wierstra, D., and Hassabis, D. Model-free episodic control. *arXiv preprint arXiv:1606.04460*, 2016.

Botvinick, M., Ritter, S., Wang, J. X., Kurth-Nelson, Z., Blundell, C., and Hassabis, D. Reinforcement learning, fast and slow. *Trends in cognitive sciences*, 23(5):408–422, 2019.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Chen, X., Wang, C., Zhou, Z., and Ross, K. Randomized ensembled double q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In

Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1582–1591. PMLR, 2018.

Gilboa, I. and Schmeidler, D. Case-based decision theory. *The Quarterly Journal of Economics*, 110(3):605–639, 1995.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1856–1865. PMLR, 2018.

Hansen, S., Pritzel, A., Sprechmann, P., Barreto, A., and Blundell, C. Fast deep reinforcement learning using on-line adjustments from the past. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018*, pp. 10590–10600, 2018.

He, F. S., Liu, Y., Schwing, A. G., and Peng, J. Learning to play in a day: Faster deep reinforcement learning by optimality tightening. In *5th International Conference on Learning Representations, ICLR 2017*. OpenReview.net, 2017.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M. G., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In McIlraith, S. A. and Weinberger, K. Q. (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pp. 3215–3222. AAAI Press, 2018.

Kahn, C. H. et al. *The art and thought of Heraclitus: a new arrangement and translation of the Fragments with literary and philosophical commentary*. Cambridge University Press, 1981.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*. PMLR, 2018.

Kuznetsov, A., Shvechikov, P., Grishin, A., and Vetrov, D. P. Controlling overestimation bias with truncated mixture of continuous distributional quantile critics. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5556–5566. PMLR, 2020.

Lan, Q., Pan, Y., Fyshe, A., and White, M. Maxmin q-learning: Controlling the estimation bias of q-learning. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

Lee, S. Y., Choi, S., and Chung, S. Sample-efficient deep reinforcement learning via episodic backward update. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pp. 2110–2119, 2019.

Lengyel, M. and Dayan, P. Hippocampal contributions to control: The third way. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, pp. 889–896. Curran Associates, Inc., 2007.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016,*, 2016.

Lin, Z., Zhao, T., Yang, G., and Zhang, L. Episodic memory deep q-networks. In Lang, J. (ed.), *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pp. 2433–2439. ijcai.org, 2018. doi: 10.24963/ijcai.2018/337.

Marr, D., Willshaw, D., and McNaughton, B. Simple memory: a theory for archicortex. In *From the Retina to the Neocortex*, pp. 59–128. Springer, 1991.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Oh, J., Guo, Y., Singh, S., and Lee, H. Self-imitation learning. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3875–3884. PMLR, 2018.

Ormoneit, D. and Sen, Ś. Kernel-based reinforcement learning. *Machine learning*, 49(2):161–178, 2002.

Peters, J. and Bagnell, J. A. Policy gradient methods. *Scholarpedia*, 5(11):3698, 2010.

Pritzel, A., Uria, B., Srinivasan, S., Badia, A. P., Vinyals, O., Hassabis, D., Wierstra, D., and Blundell, C. Neural episodic control. In Precup, D. and Teh, Y. W. (eds.),

*Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2827–2836. PMLR, 2017.

Schulman, J., Moritz, P., Levine, S., Jordan, M. I., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016*, 2016.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shohamy, D. and Wagner, A. D. Integrating memories in the human brain: hippocampal-midbrain encoding of overlapping events. *Neuron*, 60(2):378–389, 2008.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. A. Deterministic policy gradient algorithms. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pp. 387–395. JMLR.org, 2014.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Singh, S. P. and Sutton, R. S. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1):123–158, 1996.

Sutherland, R. J. and Rudy, J. W. Configural association theory: The role of the hippocampal formation in learning, memory, and amnesia. *Psychobiology*, 17(2):129–144, 1989.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988a.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988b.

Sutton, R. S., McAllester, D. A., Singh, S. P., Mansour, Y., et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, volume 99, pp. 1057–1063. Citeseer, 1999.

Thrun, S. and Schwartz, A. Issues in using function approximation for reinforcement learning. In *Proceedings of the Fourth Connectionist Models Summer School*, pp. 255–263. Hillsdale, NJ, 1993.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Touati, A., Bacon, P., Precup, D., and Vincent, P. Convergent TREE BACKUP and RETRACE with function approximation. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4962–4971. PMLR, 2018.

Tsividis, P., Pouncy, T., Xu, J. L., Tenenbaum, J. B., and Gershman, S. J. Human learning in atari. In *2017 AAAI*. AAAI Press, 2017.

van Hasselt, H. Double q-learning. In Lafferty, J. D., Williams, C. K. I., Shawe-Taylor, J., Zemel, R. S., and Culotta, A. (eds.), *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010*, pp. 2613–2621. Curran Associates, Inc., 2010.

van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In Schuurmans, D. and Wellman, M. P. (eds.), *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 2094–2100. AAAI Press, 2016.

Van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M., and de Freitas, N. Dueling network architectures for deep reinforcement learning. In Balcan, M. and Weinberger, K. Q. (eds.), *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1995–2003. JMLR.org, 2016.

Zhu, G., Lin, Z., Yang, G., and Zhang, C. Episodic reinforcement learning with associative memory. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.