Knowledge Transfer from Teachers to Learners in Growing-Batch Reinforcement Learning

Patrick Emedom-Nnamdi [1] Abram L. Friesen [2] Bobak Shahriari [2] Nando de Freitas [2] Matt W. Hoffman [2]

[1]Harvard University [2]DeepMind

Abstract

Standard approaches to sequential decision-making exploit an agent's ability to continually interact with its environment and improve its control policy. However, due to safety, ethical, and practicality constraints, this type of trial-and-error experimentation is often infeasible in many real-world domains such as healthcare and robotics. Instead, control policies in these domains are typically trained offline from previously logged data or in a *growing-batch* manner. In this setting a fixed policy is deployed to the environment and used to gather an entire batch of new data before being aggregated with past batches and used to update the policy. This improvement cycle can then be repeated multiple times. While a limited number of such cycles is feasible in real-world domains, the quantity and diversity of the resulting data are much lower than in the standard continually-interacting approach. However, data collection in these domains is often performed in conjunction with human experts, who are able to label or *annotate* the collected data. In this paper, we first explore the trade-offs present in this growing-batch setting, and then investigate how information provided by a teacher (i.e., demonstrations, expert actions, and gradient information—differentiated with respect to actions) can be leveraged at training time to mitigate the sample complexity and coverage requirements for actor-critic methods. We validate our contributions on tasks from the DeepMind Control Suite.

Background

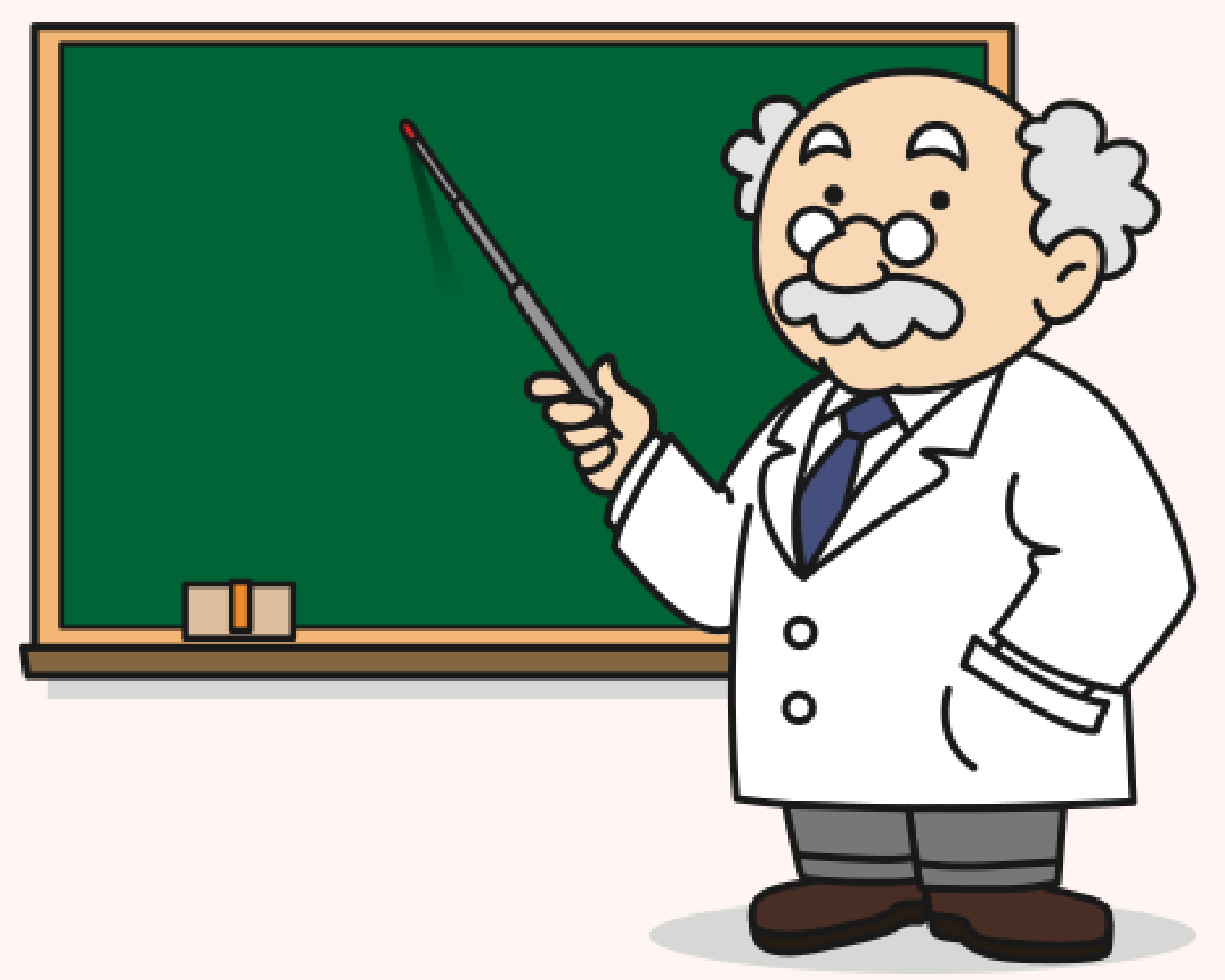
In most real-world applications of RL,

- Continual improvement of decision-making policies during environmental interaction is **infeasible** due to **resource/safety constraints**
- Policies are learned in an offline or a **growing-batch** manner
- Learning optimal policies is difficult due to limited opportunities for **online self-corrections**

Motivation

In many domains, **well-informed/task-specific knowledge** exists and can be queried

- An external teacher can help alleviate some of these issues, providing **external corrective feedback**
- Teachers can be represented as **human, RL agent, or program**
- Expert knowledge by the teacher can be provided
  1. Up-front via **demonstrations**, or
  2. Throughout the life-cycle of the student agent via **annotations**



How can we leverage teachers to improve the sample efficiency and performance of value-based RL agents learned in the growing-batch setting?

Growing-batch Reinforcement Learning with Expert Annotations

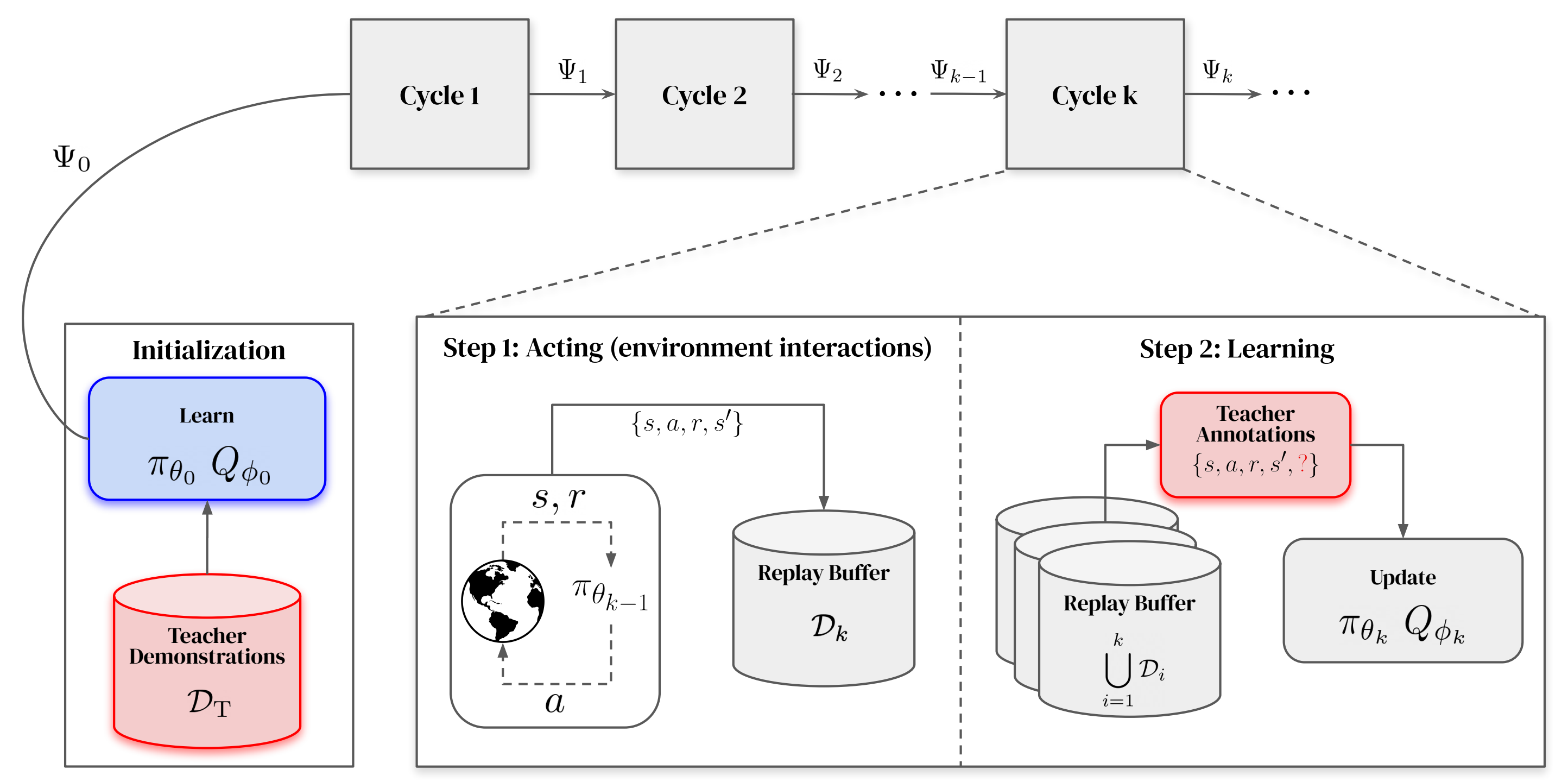


Figure 1. Growing-batch RL with teacher annotations. The policy and (optionally) critic networks are first initialized from offline data; in our work we use an offline dataset of teacher demonstrations. Within each cycle, a fixed policy $\pi_{\theta_{k-1}}$ is deployed within the environment and used to gather data in the replay buffer $\mathcal{D}_k$. Data from previous cycles are then aggregated (i.e., $\cup_{i=1}^{k}\mathcal{D}_i$) along with per-transition teacher annotations, and used to update the policy and critic networks, π_{θ_k} and Q_{ϕ_k}, respectively. The forms of annotations are teacher-suggested actions and teacher–provided critic gradients differentiated with respect to actions.

Actor Critic Agents

For our experiments, we primarily utilize **D4PG agents** to accommodate continuous actions, where we

1. Learn a critic $Q_\phi(s,a)$ via **distributional TD-Learning**, where $Q_\phi(s,a) = \mathbb{E}Z_\phi(s,a)$ and $Z_\phi(s,a)$ is estimated by minimizing the distributional TD error

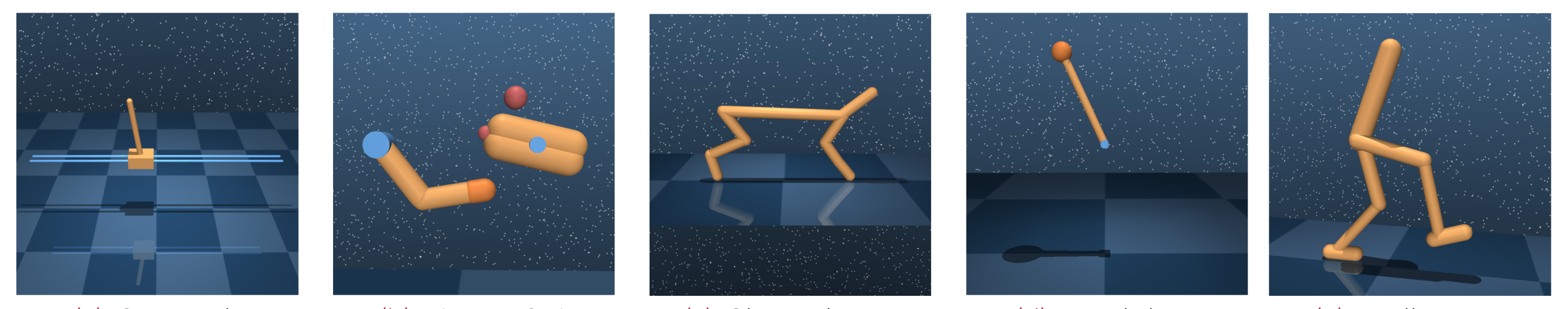
$$\mathcal{L}(\phi) = \mathbb{E}_{s\sim\rho^\pi}\left[d\left(\mathcal{T}_{\pi_{\theta'}} Z_{\phi'}(s,a), Z_\phi(s,a)\right)\right]$$

2. Learn a parametric policy π_θ using the **deterministic policy gradient (DPG)**

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_{s\sim\rho^\pi}\left[\nabla_\theta \pi_\theta(s) \nabla_a Q_\phi(s,a)\big|_{a=\pi_\theta(s)}\right]$$

where $\mathcal{J}(\theta) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[Q^{\pi_\theta}(s,a)\right]$ is the expected return.

DeepMind Control Suite



(a) Cart-pole: Balance & Swing-up (b) Finger-Spin (c) Cheetah-Run (d) Pendulum Swing-up (e) Walker-Run

Figure 2. The DeepMind Control Suite environments used in our experiments.

- Evaluated experiments on **six control suite environments**
- Fixed the **total number of actor steps** to 2M, which is the value needed to solve the task by an online agent
- Selected the **total number of SGD steps** to match the sample per insert (SPI) ratio of 32

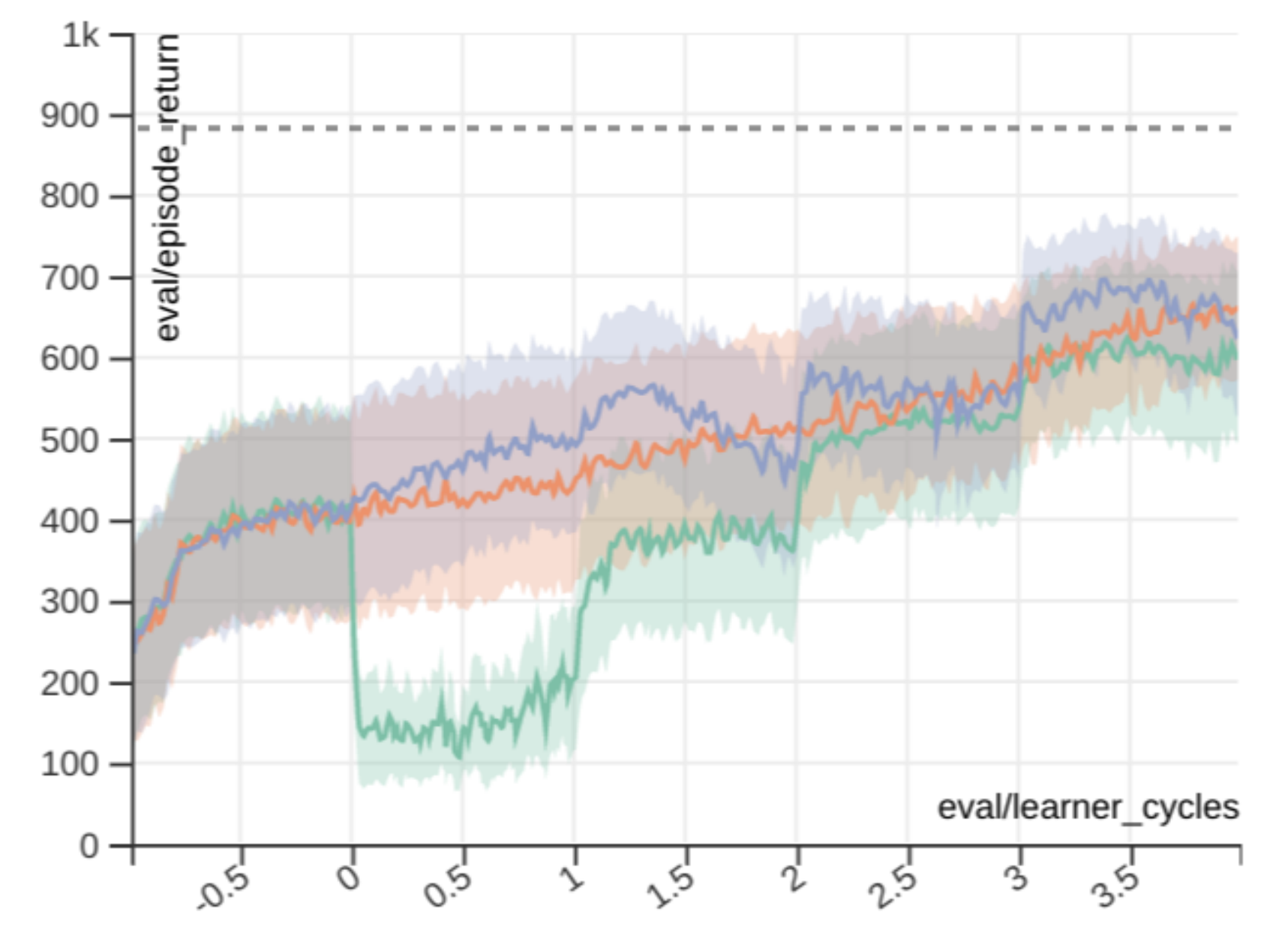
BC-Policy Regularization (Blissful Ignorance)

Assuming that the BC-initialized policy is **sub-optimal**, we would like the agent to **surpass it's performance**:

- Incorporated an **exponential decay weight** $\alpha \in (0,1)$ as a hyper-parameter

$$\mathcal{J}(\theta_k) = (1-\alpha)\,\mathcal{J}_{\text{D4PG}}(\theta_k) + \alpha\, \underbrace{\mathbb{E}_{s\sim\rho^\pi}\left[\|\pi_{\text{BC}}(s) - \pi_{\theta_k}(s)\|_2^2\right]}_{\text{BC regularizer}}$$

- By treating α as a **function of total learner steps**,
  1. We first **stay close** to the initialized policy
  2. Then, gradually **prioritize solely learning from the DPG** loss component



(a) All Environments, Number of cycles = 4

Figure 3. Baseline with BC initialization only vs. BC-policy regularizer with decay rate = 1 and decay rate = 5.

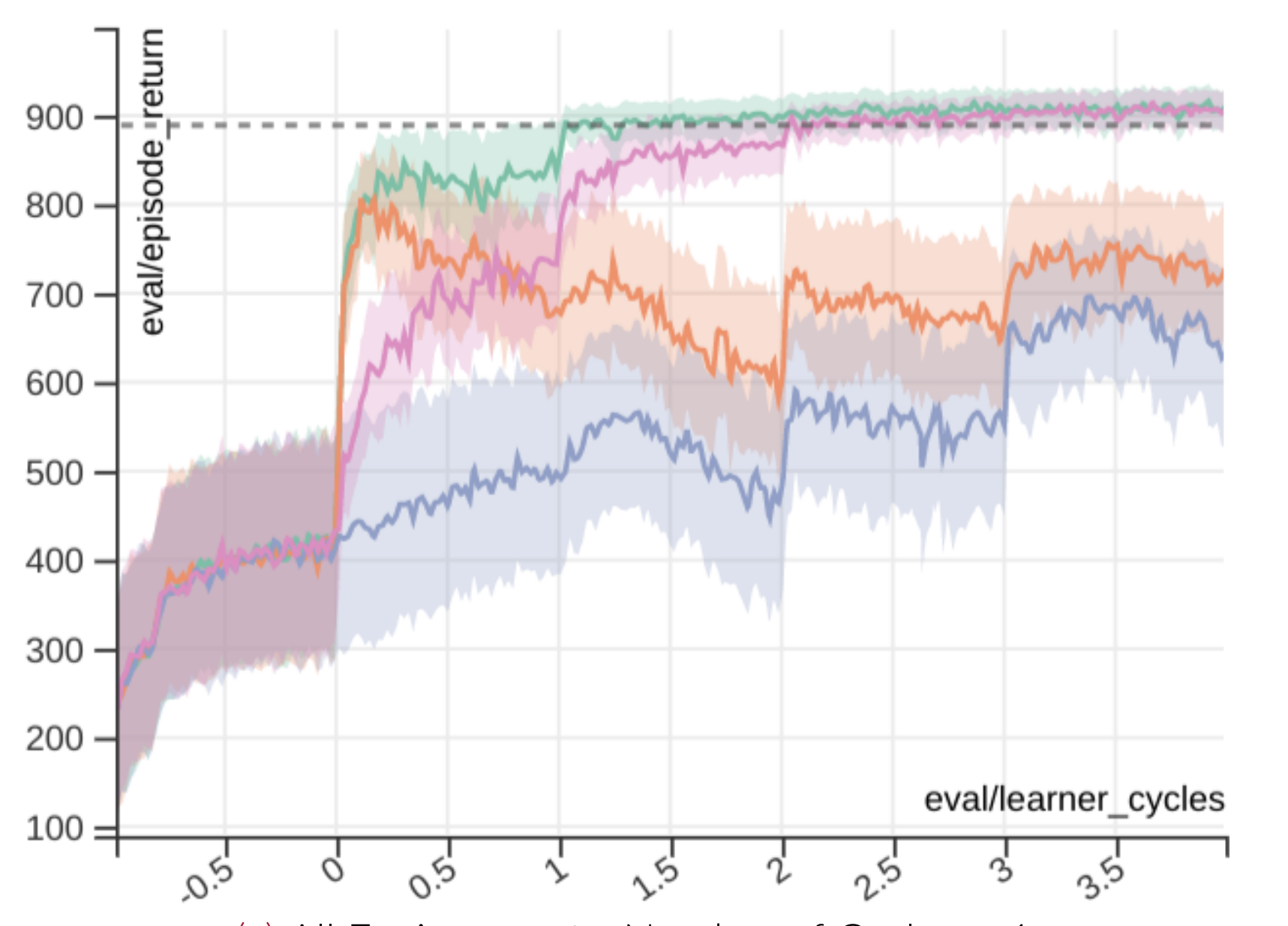
Filtered Teacher-Suggested Actions

We explore using a Q-filter to **determine which loss component (Teacher-action vs. D4PG) to learn from** for a given transition:

$$\nabla\mathcal{J}(\theta_k) = \mathbb{E}_{s\sim\rho^\pi}\Bigg[\underbrace{[1-\delta(s)]\nabla_{\theta_k}\pi_{\theta_k}\nabla_a Q_{\phi_k}(s,a)\big|_{a=\pi_{\theta_k}(s)}}_{\text{D4PG component}} + \underbrace{\delta(s)\nabla_{\theta_k}\|a^* - \pi_{\theta_k}(s)\|_2^2}_{\text{Teacher-action component}}\Bigg],$$

where $\delta(s) = 1\left[Q_{\phi_k}(s,a^*) \geq Q_{\phi_k}(s,\pi_\theta(s))\right]$ is a **Q-filter**.

- Ignore (or de-prioritize) learning from actions that produce lower values than using $\pi_{\theta_k}(s)$



(a) All Environments, Number of Cycles = 4

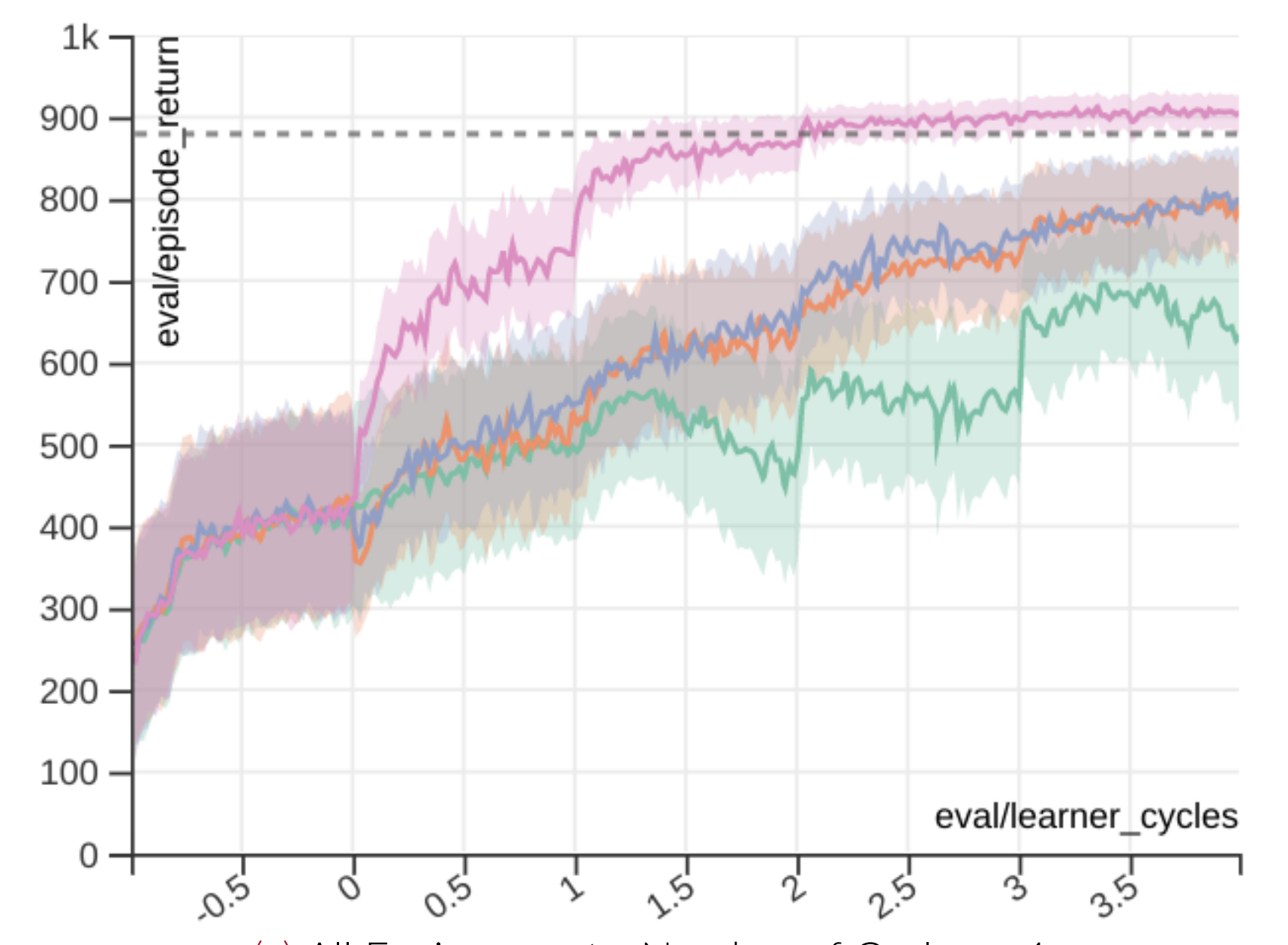
Figure 4. Baseline using BC-policy regularizer vs. Filtered Teacher-action vs. Teacher-action auxiliary loss with decay rate = 1, and decay rate = 5.

Teacher-gradient Annotations

We consider directly incorporating **gradient information** $G_a(s,a)$ **differentiated with respect to actions** a provided by the teacher.

$$\nabla\mathcal{J}(\theta_k) = \mathbb{E}_{s\sim\mathcal{D}}\Bigg[(1-\alpha)\nabla_{\theta_k}\pi_{\theta_k}\nabla_a Q_{\phi_k}(s,a)\big|_{a=\pi_{\theta_k}(s)} + \alpha\,\underbrace{G_a(s,\pi_{\theta_k}(s))\nabla_{\theta_k}\pi_{\theta_k}(s)}_{\text{Teacher-provided gradient signal}}\Bigg]$$

- Relying on teacher-provided gradients early can circumvent risks associated with learning from a **poorly initialized** or **potentially over-estimated** value function.
- These gradients can be interpreted as the directions in which the agent should adjust their actions to enhance their current policy.



(a) All Environments, Number of Cycles = 4

Figure 5. Baseline using BC-policy regularizer vs. Filtered DAgger vs. Expert Gradient auxiliary loss with decay rate = 1, and decay rate = 5.

patrickemedom@g.harvard.edu