DCUR: Data Curriculum for Teaching via Samples with Reinforcement Learning

Daniel Seita[1], **Abhinav Gopal**[1], **Zhao Mandi**[1], **John Canny**[1]
[1]University of California, Berkeley
`seita@berkeley.edu`

Abstract

Deep reinforcement learning (RL) has shown great empirical successes, but suffers from brittleness and sample inefficiency. A potential remedy is to use a previously-trained policy as a source of supervision. In this work, we refer to these policies as *teachers* and study how to transfer their expertise to new *student* policies by focusing on data usage. We propose a framework, **D**ata **CU**rriculum for **R**einforcement learning (DCUR), which first trains teachers using online deep RL, and stores the logged environment interaction history. Then, students learn by running either offline RL or by using teacher data in combination with a small amount of self-generated data. DCUR's central idea involves defining a class of data curricula which, as a function of training time, limits the student to sampling from a fixed subset of the full teacher data. We test teachers and students using state-of-the-art deep RL algorithms across a variety of data curricula. Results suggest that the choice of data curricula significantly impacts student learning, and that it is beneficial to limit the data during early training stages while gradually letting the data availability grow over time. We identify when the student can learn offline and match teacher performance without relying on specialized offline RL algorithms. Furthermore, we show that collecting a small fraction of online data provides complementary benefits with the data curriculum. Supplementary material is available at `https://tinyurl.com/teach-dcur`.

1 Introduction

Humans often learn best when guided through a curricula. When providing expert demonstrations to human students, the demonstrations should fall within a particular range of difficulties. If they are too easy the student learns nothing, but if they are too difficult the student may have trouble learning [7, 53]. With this intuition, we consider the analogous context in Reinforcement Learning (RL) [47], and study how a teacher policy can best "teach" a student policy, where both are trained with reinforcement learning. We investigate when a teacher provides the student with a dataset of samples (*i.e.,* data tuples) $\mathcal{D}^{(T)}$. For the student, rather than propose a new algorithm to learn from data, we use *existing* algorithms but focus on *data usage*. In particular, given a student employing a standard off-the-shelf RL algorithm, how can it better sample from $\mathcal{D}^{(T)}$?

We propose a framework, **DCUR** (pronounced *dee-curr*): **D**ata **CU**rriculum for **R**einforcement learning, where we study how to filter a teacher's static dataset to accelerate a student's learning progress, measured in terms of environment episodic reward. The framework is compatible with pure offline reinforcement learning [10, 31, 32] and when the student can engage in a small amount of self-generated data, which we refer to as apprenticeship learning. Either case reduces the need for the student to engage in extensive and potentially risky exploratory behavior, and thus may hold tremendous promise for enabling robots to learn from existing, massive datasets. In experiments, we test the DCUR framework by training teachers in a standard fashion with online environment

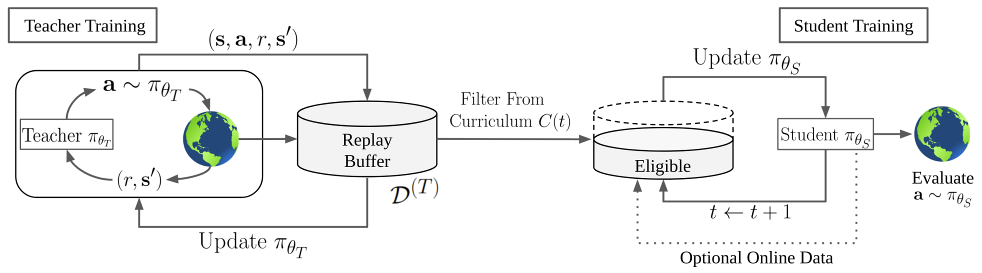


Figure 1: Visualization of DCUR. First, a *teacher* policy π_{θ_T} is trained using standard Deep RL with online environment interactions, and fills a replay buffer $\mathcal{D}^{(T)}$ with all the experienced data tuples. After training π_{θ_T}, we choose a fixed data curriculum $C(t)$ (see Figure 2) for training a *student* policy π_{θ_S} from $\mathcal{D}^{(T)}$. In general, we study when students train *offline*, but since this can be challenging, we optionally allow students to gather some online data to use for learning, in addition to $\mathcal{D}^{(T)}$. For each time t (*i.e.,* minibatch gradient update), the fixed curriculum strategy $C(t)$ restricts the samples the student can draw from $\mathcal{D}^{(T)}$, resulting in a set of eligible data tuples (shaded in light gray) that the student can use for gradient updates.

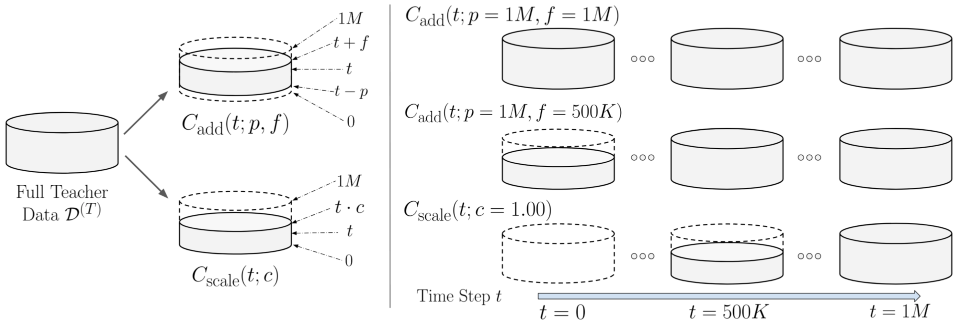
interaction. We store the logged history of experienced environment interactions as a large dataset $\mathcal{D}^{(T)}$ to be used by the student for learning. Results over a range of standard continuous control tasks [6, 49] suggest that students running off-the-shelf, off-policy RL algorithms can make use of curricula to more efficiently learn from static teacher data, even without the use of specialized offline RL algorithms. See Figure 1 for an overview of DCUR. The contributions of this paper include:

- We introduce the DCUR framework which considers data improvements, rather than algorithmic improvements, for accelerating learning from large, teacher-provided datasets.
- We show that the best curricula, combined with potentially longer training, can enable students training offline to match the teacher's top performance.
- We show that students can use some online data (2.5-5.0% of the offline data size) in combination with data curricula to aid learning in more complex environments.

2 Related Work

Curriculum Learning. The use of curriculum learning in machine learning dates to at least Elman et al. [9], who showed the benefit of initially training on "easier" samples while gradually increasing the difficulty. Subsequent work by Bengio et al. [5] confirmed these results by accelerating classification and language modeling by arranging samples in order of difficulty. Other work in curriculum learning has included training teacher agents to provide samples or loss functions to a student [11, 55], and larger-scale studies to investigate curricula for image classification [56]. In RL, curriculum learning has shown promise in multi-task [35, 23] and self-play [46] contexts, for selecting one of several teachers to provide samples [44], and for generating a curriculum of start [14] or goal [13, 61] states. In this work, we design a curriculum of samples (*i.e.,* data tuples) for (mostly) offline RL, and we do not focus on the multi-task setting nor do we require self-play or goal generation. When students execute online steps, this can be interpreted as a form of apprenticeship learning [1] where the student can "practice" in addition to using offline data.

Offline Reinforcement Learning. Offline RL [32], also referred to as Batch RL [10, 31], has seen an explosion of recent interest. Offline RL is the special case of reinforcement learning [47] without exploration, so the agent must learn from a static dataset. It differs from imitation learning [38] in that data is annotated with rewards, which can be utilized by reinforcement learning algorithms to learn better policies than the underlying data generating policy. Many widely utilized Deep RL algorithms, such as DQN [36, 50] for discrete control and DDPG [33], SAC [20], and TD3 [18] for continuous control are off-policy algorithms and, in principle, capable of learning offline. In practice, however, researchers have found that such off-policy algorithms are highly susceptible to bootstrapping errors and thus may diverge quickly and perform poorly [17, 16, 27, 26, 25]. One remedy is to incorporate conservatism such as by regularizing the value functions in model-free RL settings [17, 26, 28, 62, 45, 57, 54]. Other studies have found promising results in model-based



1M
t + f
t
t − p
0
$C_{\text{add}}(t; p, f)$
Full Teacher Data $\mathcal{D}^{(T)}$
1M
t · c
t
0
$C_{\text{scale}}(t; c)$
$C_{\text{add}}(t; p = 1M, f = 1M)$
$C_{\text{add}}(t; p = 1M, f = 500K)$
$C_{\text{scale}}(t; c = 1.00)$
Time Step t
t = 0
t = 500K
t = 1M

Figure 2: Visualization of the data curricula described in Section 4.2. For all buffer visuals, the bottom represents index 0, *i.e.,* the first data tuple from the teacher's training history, and the top is index 1M. At a student training step t, data tuples available for sampling are shaded gray. **Left**: we illustrate the $C_{\text{add}}(t)$ and $C_{\text{scale}}(t)$ curricula, which determine different ranges of data tuples in $\mathcal{D}^{(T)}$. **Right**: three examples of curricula (one per row) showing how the available data to the student changes over 1M training time steps. For $C_{\text{add}}(t; p = 1M, f = 1M)$, the full $\mathcal{D}^{(T)}$ buffer is available at all times, whereas $C_{\text{scale}}(t; c = 1.00)$ only enables indices 0 to t for time t, and hence the available samples grows linearly throughout training.

contexts [59, 24, 58]. The goals of this work are orthogonal to work that attempts to develop specialized offline RL algorithms, because the focus here is on knowledge transfer from a teacher to a student, with the offline setting as one possible learning scenario for the student.

Reinforcement Learning from Teachers. Combining reinforcement learning with data from teachers is a highly effective technique for training students, particularly for hard exploration environments. One line of research has explored distillation techniques [40, 42, 8, 30] for multi-task learning, which trains student networks to match output from teacher networks (*e.g.,* Q-values). Another active area of research focuses on demonstrations [4]; in off-policy RL, a replay buffer [34] can contain teacher demonstrations, which can be used along with self-generated samples from a student. Examples of such algorithms in discrete control settings include DQfD [22], Ape-X DQfD [41], and R2D3 [39]. Other work utilizes demonstrations in continuous control by adding transitions to a replay buffer [37, 51, 52], specifying an auxiliary loss [43] or estimating value functions for model-based RL [48]. These works enable additional exploration from the student, allowing for self-generated samples, whereas we aim to understand how well students can learn with minimal exploration. Furthermore, these works often use a very low *demo ratio*, or the fraction of expert (teacher) demonstrations in a given minibatch. For example, R2D3 [39] reported that a demo ratio as small as $1/256$ was ideal. That these algorithms perform best when utilizing so little teacher data motivates the need to understand how to use teacher data without requiring frequent student environment interaction.

3 Problem Statement and Preliminaries

We utilize the Markov Decision Process (MDP) framework for RL [47]. An MDP is specified as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ where at each time step t, the agent is at state $\mathbf{s}_t \in \mathcal{S}$ and executes action $\mathbf{a}_t \in \mathcal{A}$. The dynamics map the state-action pair into a successor state $\mathbf{s}_{t+1} \sim P(\cdot \mid \mathbf{s}_t, \mathbf{a}_t)$, and the agent receives a scalar reward $r_t = R(\mathbf{s}_t, \mathbf{a}_t)$. The objective is to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes the expected discounted return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$ with discount $\gamma \in (0, 1]$. In Deep RL, the policy π_θ is parameterized by a deep neural network with parameters θ.

The RL framework we study involves two agents: a *teacher* T and a *student* S, following respective policies π_{θ_T} and π_{θ_S} with parameters θ_T and θ_S. We assume the teacher is trained via standard online RL, and produces a dataset $\mathcal{D}^{(T)} = \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_{i+1})\}_{i=0}^{N}$ of N *data tuples* (also referred to as "samples"), where each contains a state $\mathbf{s}_i$, action $\mathbf{a}_i$, scalar reward r_i, and successor state $\mathbf{s}_{i+1}$. In general, we use the i subscript notation in $(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_{i+1})$ to specify a time indexing of the tuples across the full data, and use $(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')$ when precise time indexing is not needed. Data tuples from teacher data $\mathcal{D}^{(T)}$ are provided to the student S, which runs an off-policy RL algorithm, so that it can in principle learn from just the fixed data. We consider the problem of designing a curriculum to decide, for each time step t of the student's learning progress,[1] which data tuples from $\mathcal{D}^{(T)}$ should be "available" to the student when it samples minibatches for gradient updates.

4 Method

4.1 Teachers and Data Generation

The DCUR framework is agnostic to the precise algorithm to train students and teachers. Unless stated otherwise, we generate teacher data $\mathcal{D}^{(T)}$ using TD3 [18], a state-of-the-art off-policy Deep RL algorithm for continuous control. TD3 is an actor-critic algorithm where the actor π_{θ_T} is a policy, and the critic is a value function which consists of two Q-networks, Q_{ϕ_1} and Q_{ϕ_2}, with target networks $Q_{\phi_1,\text{targ}}, Q_{\phi_2,\text{targ}}$. During gradient updates, TD3 mitigates overestimation of Q-values by taking the minimum of the two Q-networks to compute targets y for the Bellman update:

$$y = r + \gamma \min_{i\in\{1,2\}} Q_{\phi_i,\text{targ}}(\mathbf{s}', \mathbf{a}'(\mathbf{s}')) \tag{1}$$

with discount factor γ, and where $\mathbf{a}'$ is the action considered at the successor state $\mathbf{s}'$:

$$\begin{aligned} \mathbf{a}'(\mathbf{s}') &= \text{clip}(\pi_{\theta_T}(\mathbf{s}') + \epsilon, \mathbf{a}_{\text{low}}, \mathbf{a}_{\text{high}}) \\ \epsilon &\sim \text{clip}(\mathcal{N}(0,\sigma), -\beta, \beta) \end{aligned} \tag{2}$$

which in practice involves adding zero-mean clipped Gaussian noise ϵ to $\pi_{\theta_T}(\mathbf{s}')$ for some β, then clipping (again) component-wise to an environment-dependent action range $[\mathbf{a}_{\text{low}}, \mathbf{a}_{\text{high}}]$. For more details on TD3, we refer the reader to Fujimoto et al. [18]. To generate $\mathcal{D}^{(T)}$, we use the *logged environment interaction history* of the TD3 teacher from online training, resulting in a set of ordered tuples:

$$\mathcal{D}^{(T)} = \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_{i+1})\}_{i=0}^{N=1M}, \tag{3}$$

in a *replay buffer*, where following standard MuJoCo training [18], the number of environment steps and the buffer capacity are both 1M, so no data tuples are overwritten. To our knowledge, the only prior work that has tested TD3 in offline RL with logged data is from Agarwal et al. [3], who report that TD3 outperformed Batch Constrained Q-learning [17] when learning from logged data generated from DDPG agents. We perform a deeper investigation of training on logged data by showing the utility of curricula (Section 4.2) and self-generated data (Section 4.3).

4.2 Data CUrriculum for Reinforcement Learning (DCUR)

We propose to accelerate student learning with a curriculum $C(t)$ which specifies the range of eligible data tuples in the static teacher data $\mathcal{D}^{(T)}$ which can be sampled for the minibatch gradient update at time t. We define two classes of curricula: *additive* and *scale*. An additive curricula C_{add} uses two parameters, $p \geq 0$ and $f \geq 0$, specifying the *previous* and *future* data tuples in $\mathcal{D}^{(T)}$ *relative to* t. For ease of notation, we omit the p parameter if the intent is to always allow data tuples from index 0, which represents the teacher's earliest environment interaction. We thus denote the additive curricula on $\mathcal{D}^{(T)}$ using one of two conventions:

$$\begin{aligned} C_{\text{add}}(t;p,f) &= \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_{i+1})\}_{i=\max(0,t-p)}^{i=\min(1M,t+f)} \\ C_{\text{add}}(t;f) &= \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_{i+1})\}_{i=0}^{i=\min(1M,t+f)} \end{aligned} \tag{4}$$

where the valid time indices are centered at the current student training time cursor t, and are always limited by the buffer data size of $|\mathcal{D}^{(T)}| = 1M$ studied in this work. The second class of curricula C_{scale}, is parameterized by a single *scale* parameter $c > 0$ and defined as:

$$C_{\text{scale}}(t;c) = \{(\mathbf{s}_i, \mathbf{a}_i, r_i, \mathbf{s}_{i+1})\}_{i=0}^{i=\min(1M,t\cdot c)} \tag{5}$$

which enables index 0 up to index $t \cdot c$. See Figure 2 for a visualization. If the student trains for 1M gradient updates following $C_{\text{add}}(t; f = 1M)$, it has the full buffer accessible for sampling data

[1] In this work, we consider RL contexts where it is standard to have a 1:1 ratio of environment steps and gradient (*i.e.,* training) updates, modulo any initial online data collection to partially fill in a replay buffer before training begins. We thus treat a "time step" t as referring to environment steps and gradient updates interchangeably. If students learn offline, then a "time step" is interpreted as a gradient update only.

Table 1: **Offline RL Results with DCUR**. Student performance as a function of 11 data curricula, with teacher performance (bottom row) as a reference. We report the "M1" and "M2" evaluation metrics (Section 5). We run 1 teacher per environment, then use 5 random seeds for training students from that same teacher data. Hence, all numbers reported below for student data curriculum experiments are averages over 5 independent offline RL runs; see the Appendix for standard error values. For each column, values are bolded for the best M1 and M2 results among *all* 11 curricula for students, *and* for other students with overlapping standard errors. The bolded values do not consider teacher performance, which is only present as a reference.

	Ant-v3		HalfCheetah-v3		Hopper-v3		Walker2d-v3	
Data Curriculum	M1	M2	M1	M2	M1	M2	M1	M2
$C_{\text{add}}(t; f = 50K)$	219.9	382.0	**8067.3**	**7147.0**	2986.2	1669.5	**2715.6**	**1712.1**
$C_{\text{add}}(t; f = 100K)$	-604.2	-409.6	7416.2	**7052.6**	2627.4	1980.1	**2746.7**	**1810.5**
$C_{\text{add}}(t; f = 200K)$	288.4	-620.4	**8028.4**	6521.5	2525.3	1887.9	**2691.8**	**1651.9**
$C_{\text{add}}(t; f = 500K)$	283.3	-801.1	7108.7	5041.2	2498.0	1546.2	1515.7	1183.5
$C_{\text{add}}(t; f = 1M)$	-1552.1	-1390.5	7447.3	3846.8	2323.0	1610.3	1741.4	1096.4
$C_{\text{add}}(t; p = 800K, f = 0)$	-889.7	1497.9	**7650.4**	6942.7	2132.2	1924.4	792.0	1521.8
$C_{\text{scale}}(t; c = 0.50)$	285.0	301.4	7467.8	6261.9	2332.3	1450.5	1866.0	799.9
$C_{\text{scale}}(t; c = 0.75)$	167.0	412.2	7392.6	6689.2	**3284.7**	1818.2	1864.9	1251.5
$C_{\text{scale}}(t; c = 1.00)$	825.6	1103.7	**8305.3**	6980.0	2984.3	2092.6	2178.5	1305.7
$C_{\text{scale}}(t; c = 1.10)$	**2952.5**	**2212.0**	**8306.0**	**7095.6**	**3185.2**	**2317.1**	2423.9	**1698.1**
$C_{\text{scale}}(t; c = 1.25)$	-1851.1	199.6	7843.8	**7175.4**	2755.5	1891.9	**2839.4**	**1747.4**
TD3 Teacher	4876.2	3975.8	8573.6	7285.5	3635.2	2791.9	3927.9	2579.8

tuples at all times. Using $C_{\text{scale}}(t; c = 1.00)$ means the student can only sample from indices 0 up to t at time t, so the available offline data grows over time. In addition, $C_{\text{add}}(t; f = 0)$ and $C_{\text{scale}}(t; c = 1.00)$ define the same curriculum; we default to the latter notation.

In prior work, Fujimoto et al. [17] tested two special cases of these curricula, named *final buffer* and *concurrent*. The final buffer setting enables the entire data at all times for learning and was later studied in Agarwal et al. [3]. This is equivalent to $C_{\text{add}}(t; f = 1M)$. The concurrent setting is represented as $C_{\text{add}}(t; c = 1.00)$. We remark that that limiting the size of the buffer is a known option to stabilize online RL [60], but we aim to study this in an largely offline context in more complex environments, and where the eligible data buffer can grow over time.

4.3 Apprenticeship Learning: Small Amount of Online Data

While we primarily study the student learning purely offline, we additionally explore learning with small amounts of on-policy student data, to see how much this stabilizes training and to also check that such effects are complementary with a data curriculum. This means the student forms a smaller buffer $\mathcal{D}^{(S)}$ of self-generated online data with exploration noise, and where $|\mathcal{D}^{(S)}| \ll |\mathcal{D}^{(T)}|$. We call this setting *X% Online* if the student, by the end of its training, has collected self-generated data tuples that amount to X% of the full teacher data (1M in this work). The student still performs 1M gradient updates, but takes one online step at equally spaced time intervals (*i.e.,* one step every 100/X updates). For example, with "5% Online," the student takes online environment steps 1 out of every 20 gradient updates, and $\mathcal{D}^{(S)}$ contains 50K data tuples at the end of training. This can equivalently be viewed as the student performing 50K consecutive online environment steps, but with 20 gradient updates between consecutive steps. The data tuples from $\mathcal{D}^{(S)}$ are never discarded. When the student samples a minibatch at time t, it first applies the pre-selected curriculum (see Section 4.2) to get the appropriate subset of $\mathcal{D}^{(T)}$, then combines the resulting eligible data tuples with all of $\mathcal{D}^{(S)}$, then uniformly samples from that.

5 Experiments

Environments. We test DCUR using standard MuJoCo environments [49] for Deep RL: Ant-v3, HalfCheetah-v3, Hopper-v3, and Walker2d-v3 from OpenAI [6]. Earlier versions of these environments (-v1 or -v2) have also been benchmarked in work focusing on offline RL [17, 26, 57].

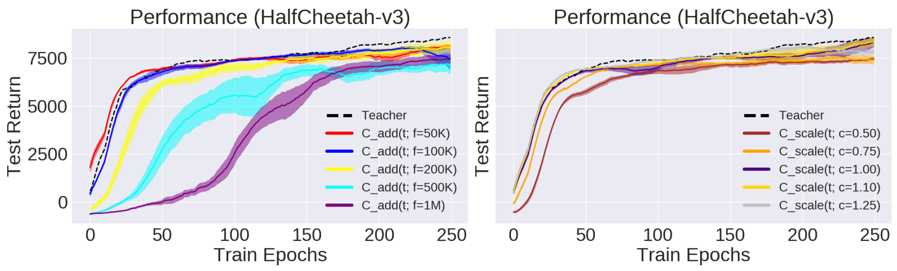


Figure 3: Offline student performance on HalfCheetah-v3 with additive curricula (left) and scale curricula (right); these experiments correspond to numerical values in Table 1. In both subplots, we show the teacher's performance for reference (at *test* time, without noise) with dashed black lines. Results from additive curricula suggest a clear pattern that access to more samples initially (*i.e.,* larger f) slows learning.

Teachers and Students. We train TD3 teachers using the standard 1M online steps [18, 20, 57], and store all encountered data tuples $(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')$ to make $\mathcal{D}^{(T)}$. Each training epoch consists of 4000 time steps, so there are 250 training epochs for teachers and students. In Section 6.2 we investigate training students for 10X more epochs. We apply standard noise levels for the teacher's exploration; the noise added to actions is $\mathcal{N}(0, 0.1)$ instead of $\mathcal{N}(0, 0.5)$ as done in some experiments in [3, 17] to increase data diversity. In the Appendix, we have results from SAC [20] teachers and students. The code we use is built on top of SpinningUp [2].

Data Curriculum Experiments. We test a variety of additive and scale curricula from Section 4.2. For additive curricula, we adjust the "forward" samples allowed: $f \in \{50K, 100K, 200K, 500K, 1M\}$, and we also check if ignoring older samples in $\mathcal{D}^{(T)}$ helps (with $p = 800K$). We report scale curricula with $c \in \{0.50, 0.75, 1.00, 1.10, 1.25\}$, where $c < 1$ tests whether the student needs more gradient updates on data tuples in $\mathcal{D}^{(T)}$ relative to the teacher, and $c > 1$ tests whether it helps to have additional "forward" samples, which is similar to $f > 0$ in additive curricula, but where f increases throughout student training because the maximum eligible index $c \cdot t$ is a function of t. Due to computational limitations, for most experiments after Section 6.1, we test two particular curricula (from [17]): all the data ($C_{\text{add}}(t; f = 1M)$) or concurrent data ($C_{\text{scale}}(t; c = 1.00)$). We test with $C_{\text{add}}(t; f = 1M)$ because it serves a baseline of using all data, which can intuitively be viewed as using no curriculum. As Deep RL evaluations are notoriously noisy [21], all experiments are reproducible from code and data available on the project website.

Evaluation. To evaluate student and teacher performance, we use two metrics, "M1" and "M2":

- **M1 (Final)**: average reward of the last 100 test episodes.
- **M2 (Average)**: average reward across all test episodes.

In all experiments, students and teachers do 10 test episodes after every epoch. We use 5 random seeds for students, with respect to *one* teacher (per environment), to reduce the source of variability that would result from different teachers. We compute M1 and M2 statistics for each of the 5 runs, then average those 5 to get final numbers for M1 and M2. In the tables, when comparing a relevant set of students, we bold the best M1 and M2 results, *and additionally* bold other results with overlapping standard errors for a fairer comparison; see the Appendix for the exact formula.

6 Results

We present experimental results of DCUR and report M1 and M2 metrics. We defer some results to the Appendix.

6.1 Effect of Data Curricula on Student Training Offline

We train students offline using 11 data curricula and list results in Table 1. The overall results suggest that a curriculum allowing for the available samples from $\mathcal{D}^{(T)}$ to grow over time, and to include a few samples "ahead" relative to the student's training time t produces stronger results. In particular,

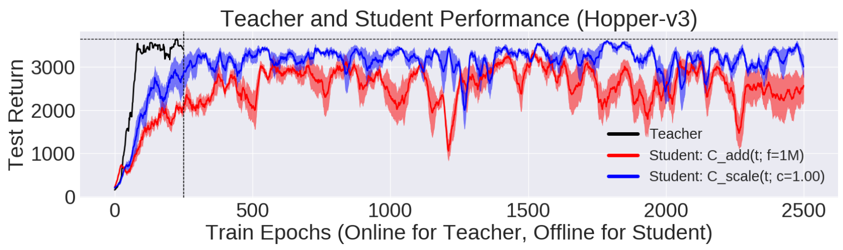


Figure 4: Hopper-v3 test-time episodic returns comparing the teacher performance (black curve) over its training period of 250 epochs (dashed vertical line), versus students trained offline over 2500 (10X more) epochs. For the two data curricula, we train 5 independent students from the fixed teacher data $\mathcal{D}^{(T)}$ with different seeds. Results suggest that the $C_{\rm scale}(t; c = 1.00)$ curriculum (blue curve) leads to better average performance over 2500 epochs, and matches the teacher's best performance (dashed horizontal line).

Table 2: **Offline RL Results with DCUR, 10X Training**. Student performance as a function of the data curriculum, where students train for 10X longer (2500 epochs) as compared to Table 1. Besides this change, the setup and table formatting is identical to Table 1. See Section 6.2 for details. The teacher metrics are repeated for reference, and are not considered when bolding numbers here.

	Ant-v3		HalfCheetah-v3		Hopper-v3		Walker2d-v3	
Curriculum	M1	M2	M1	M2	M1	M2	M1	M2
$C_{\rm add}(t;$ f=1M)	**-2770.0**	-2088.3	**6041.6**	6847.9	**2593.1**	2482.0	**3713.1**	2763.5
$C_{\rm scale}(t;$ c=1.00)	**-2780.6**	**-1511.2**	**6496.2**	**7555.5**	**3019.4**	**3052.3**	**3208.0**	**3121.4**
TD3 Teacher	4876.2	3975.8	8573.6	7285.5	3635.2	2791.9	3927.9	2579.8

$C_{\rm scale}(t; c = 1.10)$ has the best results, as it obtains the top scores or close to it (*i.e.,* with overlapping standard error) on 7 out of the 8 columns in Table 1. The next best curriculum, with 4 out of 8 top scores, is $C_{\rm add}(t; f = 50K)$, and it similarly allows the available range of samples to go slightly past t, in this case by a fixed $t + 50K$. In contrast, allowing too much data with $C_{\rm add}(t; f = 1M)$ or $C_{\rm add}(t; f = 500K)$ results in poor performance, with neither of these curricula ranking among the best in any of the environments with respect to either metric. Ignoring older samples from the teacher's early training history with $C_{\rm add}(t; p = 800K, f = 0)$ also exhibits weak performance. See Figure 3 for a representative set of learning curves for HalfCheetah-v3, showing some curricula that result in the student essentially matching teacher performance, despite known challenges associated with pure offline learning, even with relying on concurrent-style training [17].

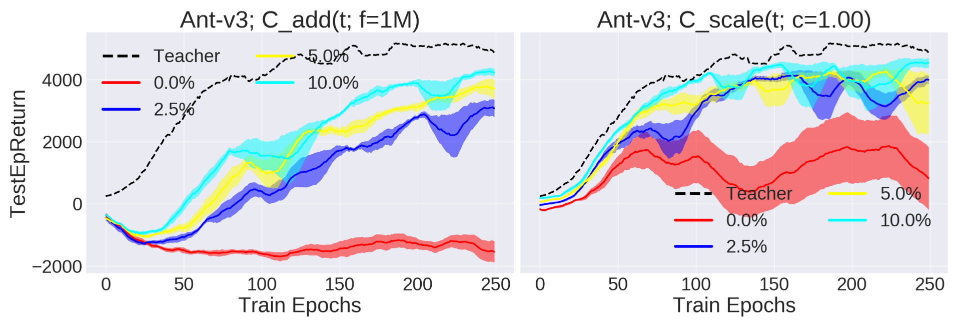


Figure 5: Ant-v3 student test performance with various amounts of online data, from 0% (*i.e.,* offline) to 10% online. We show the teacher curve (dashed black line) for reference. We plot results from the two curricula tested in Table 3, $C_{\rm add}(t; f = 1M)$ (left), and $C_{\rm scale}(t; c = 1.00)$ (right).

Table 3: **DCUR Results with Online Data.** Student performance based on one of two data curricula and the addition of a small fraction of online data. At the end of 250 training epochs, the student has collected self-generated, online data that constitutes 2.5%, 5.0%, or 10.0% of the original teacher data size $|\mathcal{D}^{(T)}| = 1M$. We use the same teacher data as in Table 1 and report the M1 and M2 metrics. We bold values by comparing only the two curricula with the same X% online experiments per column and bolding the maximum only (if standard errors do not overlap) or both (if otherwise).

	Ant-v3		HalfCheetah-v3		Hopper-v3		Walker2d-v3	
Curriculum; % Online	M1	M2	M1	M2	M1	M2	M1	M2
$C_{\text{add}}(t; f = 1M)$; 2.5%	3093.1	894.4	**8581.7**	5275.3	**3354.2**	1977.9	**3298.1**	1882.6
$C_{\text{scale}}(t; c = 1.00)$; 2.5%	**4004.7**	**2857.0**	**8417.9**	**7212.2**	2712.9	**2238.1**	**3144.9**	**2050.3**
$C_{\text{add}}(t; f = 1M)$; 5.0%	**3693.9**	1556.6	**8658.8**	5634.6	**3232.1**	2230.5	**3243.8**	**2097.3**
$C_{\text{scale}}(t; c = 1.00)$; 5.0%	**3251.1**	**3068.2**	**8601.4**	**7274.0**	**3356.2**	**2459.0**	**3155.8**	**2097.0**
$C_{\text{add}}(t; f = 1M)$; 10.0%	4229.0	2007.6	**8864.2**	6024.5	**2741.2**	1979.4	**3607.7**	**2448.2**
$C_{\text{scale}}(t; c = 1.00)$; 10.0%	**4510.9**	**3349.0**	8608.5	**7290.5**	**3182.9**	**2580.6**	3146.9	2204.2
TD3 Teacher	4876.2	3975.8	8573.6	7285.5	3635.2	2791.9	3927.9	2579.8

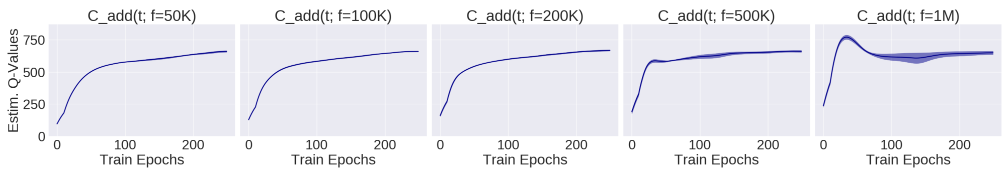


Figure 6: Students' estimated Q-values in HalfCheetah-v3. With more data available at the start (*i.e.,* increasing f, shown left to right) this creates initial over-estimation of Q-values. All 5 subplots are derived from students reported in Table 1, and all curves average over the 5 seeds and show standard errors.

6.2 Data Curriculum for Training 10X Longer

Here, as in Section 6.1, we run the $C_{\text{add}}(t; f = 1M)$ and $C_{\text{scale}}(t; c = 1.00)$ curricula again, but train 10X longer to understand how this affects learning. The choice of curriculum will *only* affect the first 1/10 of training (*i.e.,* the first 250 epochs), since after that, both curricula reduce to the student sampling data tuples from the entire teacher buffer $\mathcal{D}^{(T)}$. To make results comparable with those in Section 6.1, we use the same teacher buffers. Table 2 has results in a similar manner as in Table 1. We find that, somewhat surprisingly, the curricula affects the long-term performance of the student in the remaining 9/10 of training. Across all 8 columns (4 environments and the M1/M2 metrics), using $C_{\text{scale}}(t; c = 1.00)$ outperforms $C_{\text{add}}(t; f = 1M)$ or is statistically similar to it based on the standard error metric we use (see Section 5), suggesting that the initial curriculum assists TD3 in finding a stable set of policy and value functions so that it can continue training without significant deterioration. Figure 4 shows a representative set of learning curves for Hopper-v3, which shows the student with $C_{\text{scale}}(t; c = 1.00)$ matching the teacher's performance given sufficient training.

6.3 Results with Small Amounts of Online Data

We next study when students can use a small amount of online data to address challenges with pure offline learning. We train students for 250 epochs using 2.0%, 5.0% and 10.0% online data collection, *i.e.,* at the *end* of 1M training steps, students get 25K, 50K, and 100K self-generated data tuples, respectively, for $\mathcal{D}^{(S)}$, which they can sample from in addition to the (filtered) data tuples from $\mathcal{D}^{(T)}$. Table 3 has a complete overview of the M1 and M2 results across all 4 environments, and the Appendix contains further details. The results indicate that even with as little as 2.5% online data, students are able to significantly improve versus offline learning, with the improvement most notable in the complex Ant-v3 environment as shown in Figure 5. Furthermore, $C_{\text{scale}}(t; c = 1.00)$ continues to provide some benefit to learning speed compared to $C_{\text{add}}(t; f = 1M)$ with online data, particularly with respect to the M2 metric.

6.4 Investigation of Q-Values

To understand why the data curriculum matters, we investigate the student's estimated Q-values. With too much data available from $\mathcal{D}^{(T)}$ during early stages of training, this causes overestimation of Q-values, whereas a curriculum that restricts data tuples results in more stable, monotonically increasing Q-values. Figure 6 shows different additive curricula on HalfCheetah-v3 and plots the estimated Q-values, where the pattern of an initial "hump" at the start is prominent for certain curricula, particularly $C_{\text{add}}(t; f = 1M)$. As shown in the Appendix, a similar trend holds for other environments.

7 Conclusion and Future Work

This work introduces the DCUR framework, which studies how to filter a given dataset for existing RL algorithms. Results across a variety of curricula and training settings suggest that the choice significantly impacts the learning speed of students running RL. The greatest benefits come from curricula that gradually let the available data grow as a function of training time. We caution that the results presented are contingent on the way we generated the datasets. In future work, we will test other datasets and environments, such as D4RL [15] or RL Unplugged [19], possibly with multiple teachers [29]. While we kept relevant experience replay hyperparameters such as the *replay ratio* constant, we will use findings from recent research [12] to investigate the interplay between experience replay and data curricula. Finally, we plan to devise more sophisticated data curricula.

Acknowledgments and Disclosure of Funding

Daniel Seita was supported by the GFSD throughout this research. We thank members of the CannyLab for helpful advice and suggestions.

References

[1] P. Abbeel and A. Ng, “Apprenticeship Learning via Inverse Reinforcement Learning,” in *International Conference on Machine Learning (ICML)*, 2004.

[2] J. Achiam, “Spinning Up in Deep Reinforcement Learning,” 2018.

[3] R. Agarwal, D. Schuurmans, and M. Norouzi, “An Optimistic Perspective on Offline Reinforcement Learning,” in *International Conference on Machine Learning (ICML)*, 2020.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, “A Survey of Robot Learning from Demonstration,” *Robotics and Autonomous Systems*, vol. 57, no. 5, 2009.

[5] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum Learning,” in *International Conference on Machine Learning (ICML)*, 2009.

[6] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, “OpenAI Gym,” 2016.

[7] S. Chaiklin, “The Zone of Proximal Development in Vygotsky’s Analysis of Learning and Instruction,” *Vygotsky’s Educational Theory in Cultural Context*, 2003.

[8] W. M. Czarnecki, R. Pascanu, S. Osindero, S. M. Jayakumar, G. Swirszcz, and M. Jaderberg, “Distilling Policy Distillation,” in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[9] J. Elman, “Learning and Development in Neural Networks: The Importance of Starting Small,” *Cognition*, July 1993.

[10] D. Ernst, P. Geurts, and L. Wehenkel, “Tree-based Batch Mode Reinforcement Learning,” *Journal of Machine Learning Research*, 2005.

[11] Y. Fan, F. Tian, T. Qin, X.-Y. Li, and T.-Y. Liu, “Learning to Teach,” in *International Conference on Learning Representations (ICLR)*, 2018.

[12] W. Fedus, P. Ramachandran, R. Agarwal, Y. Bengio, H. Larochelle, M. Rowland, and W. Dabney, “Revisiting Fundamentals of Experience Replay,” in *International Conference on Machine Learning (ICML)*, 2020.

[13] C. Florensa, D. Held, X. Geng, and P. Abbeel, “Automatic Goal Generation for Reinforcement Learning Agents,” in *International Conference on Machine Learning (ICML)*, 2018.

[14] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, “Reverse Curriculum Generation for Reinforcement Learning,” in *Conference on Robot Learning (CoRL)*, 2017.

[15] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, “D4RL: Datasets for Deep Data-Driven Reinforcement Learning,” *arXiv preprint arXiv:2004.07219*, 2020.

[16] S. Fujimoto, E. Conti, M. Ghavamzadeh, and J. Pineau, “Benchmarking Batch Deep Reinforcement Learning Algorithms,” *arXiv preprint arXiv:1910.01708*, 2019.

[17] S. Fujimoto, D. Meger, and D. Precup, “Off-Policy Deep Reinforcement Learning without Exploration,” in *International Conference on Machine Learning (ICML)*, 2019.

[18] S. Fujimoto, H. van Hoof, and D. Meger, “Addressing Function Approximation Error in Actor-Critic Methods,” in *International Conference on Machine Learning (ICML)*, 2018.

[19] C. Gulcehre, Z. Wang, A. Novikov, T. L. Paine, S. G. Colmenarejo, K. Zolna, R. Agarwal, J. Merel, D. Mankowitz, C. Paduraru, G. Dulac-Arnold, J. Li, M. Norouzi, M. Hoffman, O. Nachum, G. Tucker, N. Heess, and N. de Freitas, “RL Unplugged: A Suite of Benchmarks for Offline Reinforcement Learning,” in *Neural Information Processing Systems (NeurIPS)*, 2020.

[20] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor,” in *International Conference on Machine Learning (ICML)*, 2018.

[21] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, “Deep Reinforcement Learning that Matters,” in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2018.

[22] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, G. Dulac-Arnold, I. Osband, J. Agapiou, J. Z. Leibo, and A. Gruslys, “Deep Q-Learning From Demonstrations,” in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2018.

[23] A. Jabri, K. Hsu, B. Eysenbach, A. Gupta, S. Levine, and C. Finn, “Unsupervised Curricula for Visual Meta-Reinforcement Learning,” in *Neural Information Processing Systems (NeurIPS)*, 2019.

[24] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, “MOReL: Model-Based Offline Reinforcement Learning,” in *Neural Information Processing Systems (NeurIPS)*, 2020.

[25] A. Kumar, R. Agarwal, D. Ghosh, and S. Levine, “Implicit Under-Parameterization Inhibits Data-Efficient Deep Reinforcement Learning,” in *International Conference on Learning Representations (ICLR)*, 2021.

[26] A. Kumar, J. Fu, G. Tucker, and S. Levine, “Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction,” in *Neural Information Processing Systems (NeurIPS)*, 2019.

[27] A. Kumar, A. Gupta, and S. Levine, “DisCor: Corrective Feedback in Reinforcement Learning via Distribution Correction,” in *Neural Information Processing Systems (NeurIPS)*, 2020.

[28] A. Kumar, A. Zhou, G. Tucker, and S. Levine, “Conservative Q-Learning for Offline Reinforcement Learning,” in *Neural Information Processing Systems (NeurIPS)*, 2020.

[29] A. Kurenkov, A. Mandlekar, R. Martin-Martin, S. Savarese, and A. Garg, “AC-Teach: A Bayesian Actor-Critic Method for Policy Learning with an Ensemble of Suboptimal Teachers,” in *Conference on Robot Learning (CoRL)*, 2019.

[30] K.-H. Lai, D. Zha, Y. Li, and X. Hu, “Dual Policy Distillation,” in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.

[31] S. Lange, T. Gabel, and M. Riedmiller, “Batch Reinforcement Learning,” *RL. Adaptation, Learning, and Optimization*, 2012.

[32] S. Levine, A. Kumar, G. Tucker, and J. Fu, “Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems,” *arXiv preprint arXiv:2005.01643*, 2020.

[33] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous Control With Deep Reinforcement Learning,” in *International Conference on Learning Representations (ICLR)*, 2016.

[34] L.-J. Lin, “Self-Improving Reactive Agents Based on Reinforcement Learning, Planning and Teaching,” *Machine Learning*, 1992.

[35] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, “Teacher-Student Curriculum Learning,” *arXiv preprint arXiv:1707.00183*, 2017.

[36] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-Level Control Through Deep Reinforcement Learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 02 2015.

[37] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, “Overcoming Exploration in Reinforcement Learning with Demonstrations,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[38] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, “An Algorithmic Perspective on Imitation Learning,” in *Foundations and Trends in Robotics*, 2018.

[39] T. L. Paine, C. Gulcehre, B. Shahriari, M. Denil, M. Hoffman, H. S. nd Richard Tanburn, S. Kapturowski, N. Rabinowitz, D. Williams, G. Barth-Maron, Z. Wang, N. de Freitas, and W. Team, "Making Efficient Use of Demonstrations to Solve Hard Exploration Problems," in *International Conference on Learning Representations (ICLR)*, 2020.

[40] E. Parisotto, J. L. Ba, and R. Salakhutdinov, "Actor-Mimic: Deep Multitask and Transfer Reinforcement Learning," in *International Conference on Learning Representations (ICLR)*, 2016.

[41] T. Pohlen, B. Piot, T. Hester, M. G. Azar, D. Horgan, D. Budden, G. Barth-Maron, H. van Hasselt, J. Quan, M. Večerík, M. Hessel, R. Munos, and O. Pietquin, "Observe and Look Further: Achieving Consistent Performance on Atari," *arXiv preprint arXiv:1805.11593*, 2018.

[42] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy Distillation," in *International Conference on Learning Representations (ICLR)*, 2016.

[43] S. Schmitt, J. J. Hudson, A. Zidek, S. Osindero, C. Doersch, W. M. Czarnecki, J. Z. Leibo, H. Kuttler, A. Zisserman, K. Simonyan, and S. M. A. Eslami, "Kickstarting Deep Reinforcement Learning," *arXiv preprint arXiv:1803.03835*, 2018.

[44] D. Seita, C. Tang, R. Rao, D. Chan, M. Zhao, and J. Canny, "ZPD Teaching Strategies for Deep Reinforcement Learning from Demonstrations," *arXiv preprint arXiv:1910.12154*, 2019.

[45] N. Y. Siegel, J. T. Springenberg, F. Berkenkamp, A. Abdolmaleki, M. Neunert, T. Lampe, R. Hafner, N. Heess, and M. Riedmiller, "Keep Doing What Worked: Behavioral Modelling Priors for Offline Reinforcement Learning," in *International Conference on Learning Representations (ICLR)*, 2020.

[46] S. Sukhbaatar, Z. Lin, I. Kostrikov, G. Synnaeve, A. Szlam, and R. Fergus, "Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play," in *International Conference on Learning Representations (ICLR)*, 2018.

[47] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[48] B. Thananjeyan, A. Balakrishna, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg, "Safety Augmented Value Estimation from Demonstrations (SAVED): Safe Deep Model-Based RL for Sparse Cost Robotic Tasks," in *IEEE Robotics and Automation Letters (RA-L)*, 2020.

[49] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A Physics Engine for Model-based Control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[50] H. van Hasselt, A. Quez, and D. Silver, "Deep Reinforcement Learning With Double Q-Learning," in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2016.

[51] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothorl, T. Lampe, and M. Riedmiller, "Leveraging Demonstrations for Deep Reinforcement Learning on Robotics Problems with Sparse Rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[52] M. Vecerik, O. Sushkov, D. Barker, T. Rothorl, T. Hester, and J. Scholz, "A Practical Approach to Insertion with Variable Socket Position Using Deep Reinforcement Learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[53] L. S. Vygotsky, *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press, 1978.

[54] Z. Wang, A. Novikov, K. Zolna, J. T. Springenberg, S. Reed, B. Shahriari, N. Siegel, J. Merel, C. Gulcehre, N. Heess, and N. de Freitas, "Critic Regularized Regression," in *Neural Information Processing Systems (NeurIPS)*, 2020.

[55] L. Wu, F. Tian, Y. Xia, Y. Fan, T. Qin, L. Jian-Huang, and T.-Y. Liu, "Learning to Teach with Dynamic Loss Functions," in *Neural Information Processing Systems (NeurIPS)*, 2018.

[56] X. Wu, E. Dyer, and B. Neyshabur, “When Do Curricula Work?” in *International Conference on Learning Representations (ICLR)*, 2021.

[57] Y. Wu, G. Tucker, and O. Nachum, “Behavior Regularized Offline Reinforcement Learning,” *arXiv preprint arXiv:1911.11361*, 2019.

[58] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, “COMBO: Conservative Offline Model-Based Policy Optimization,” *arXiv preprint arXiv:2102.08363*, 2021.

[59] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Zou, S. Levine, C. Finn, and T. Ma, “MOPO: Model-based Offline Policy Optimization,” in *Neural Information Processing Systems (NeurIPS)*, 2020.

[60] S. Zhang and R. S. Sutton, “A Deeper Look at Experience Replay,” in *Deep Reinforcement Learning Symposium, NeurIPS*, 2017.

[61] Y. Zhang, P. Abbeel, and L. Pinto, “Automatic Curriculum Learning through Value Disagreement,” in *Neural Information Processing Systems (NeurIPS)*, 2020.

[62] W. Zhou, S. Bajracharya, and D. Held, “PLAS: Latent Action Space for Offline Reinforcement Learning,” in *Conference on Robot Learning (CoRL)*, 2020.