DR3: VALUE-BASED DEEP REINFORCEMENT LEARNING REQUIRES EXPLICIT REGULARIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite overparameterization, deep networks trained via supervised learning are surprisingly easy to optimize and exhibit excellent generalization. One hypothesis to explain this is that overparameterized deep networks enjoy the benefits of implicit regularization induced by stochastic gradient descent, which favors parsimonious solutions that generalize well on test inputs. It is reasonable to surmise that deep reinforcement learning (RL) methods could also benefit from this effect. In this paper, we discuss how the implicit regularization effect of SGD seen in supervised learning could in fact be harmful in the offline deep RL setting, leading to poor generalization and degenerate feature representations. Our theoretical analysis shows that when existing models of implicit regularization are applied to temporal difference learning, the resulting derived regularizer favors degenerate solutions with excessive aliasing, in stark contrast to the supervised learning case. We back up these findings empirically, showing that feature representations learned by a deep network value function trained via bootstrapping can indeed become degenerate, aliasing the representations for state-action pairs that appear on either side of the Bellman backup. To address this issue, we derive the form of this implicit regularizer and, inspired by this derivation, propose a simple and effective *explicit* regularizer, called DR3, that counteracts the undesirable effects of this implicit regularizer. When combined with existing offline RL methods, DR3 substantially improves performance and stability, alleviating unlearning in Atari 2600 games, D4RL domains and robotic manipulation from images.

1 INTRODUCTION

Deep neural networks are overparameterized, which in principle should leave them vulnerable to overfitting. Despite this, supervised learning with deep networks still learn representations that generalize well. A widely held consensus is that deep nets find simple solutions that generalize due to various *implicit* regularization effects (Blanc et al., 2020; Woodworth et al., 2020; Arora et al., 2018; Gunasekar et al., 2017; Wei et al., 2019). We may surmise that using deep nets in reinforcement learning (RL) will work well for the same reason, learning effective representations that generalize due to such implicit regularization effects. But is this actually the case for value functions trained via bootstrapping?

In this paper, we argue that, while implicit regularization leads to effective representations in supervised deep learning, it may lead to poor learned representations when training overparameterized deep network value functions. There is already evidence that value functions trained via bootstrapping learn poor representations: value functions trained with offline deep RL eventually degrade in performance (Agarwal et al., 2020; Kumar et al., 2021) and this degradation is correlated with the emergence of low-rank features in the value network (Kumar et al., 2021).We focus specifically on the offline RL setting, in order to rule out any confounding effects from exploration and non-stationary data. In this paper, we aim to understand the underlying cause of the emergence of poor representations and develop a potential solution. Building on the theoretical framework developed by Blanc et al. (2020); Damian et al. (2021), we characterize the implicit regularizer that arises when training deep value functions with TD learning. This implicit regularizer implies that TD-learning would alias representations of state-action tuples that appear on either side of a Bellman backup.

We show that this theoretically predicted aliasing phenomenon manifests in practice as feature **co-adaptation**, where the features of consecutive state-action tuples learned by the Q-value network

become extremely similar in terms of their dot product (Section 3). This co-adaptation can lead to oscillatory learning dynamics, and potentially to poor performance. We find that even when Q-values are not overestimated, prolonged training in offline RL can result in performance degradation as feature co-adaptation increases. To mitigate the co-adaptation issue that arises as a result of implicit regularization, we propose an *explicit regularizer* that we call DR3 (Section 4). While exactly estimating and cancelling the effects of the theoretically derived implicit regularizer is computationally difficult, DR3 provides a simple and tractable theoretically-inspired approximation, that mitigates the issues discussed above. In practice, DR3 amounts to regularizing the features at consecutive state-action pairs to be dissimilar in terms of their dot-product similarity. Empirically, we find that DR3 allows neural net Q-functions to use their full representational capacity as measured by the rank of the learned features, giving rise to methods that train for longer, reach a better solution, and remain stable at this solution without severe degradation (Section 6).

Our first contribution is the derivation of the implicit regularizer that arises when training deep net value functions via TD learning, and empirically demonstrate it manifests as *feature co-adaptation* in the offline deep RL setting. Feature co-adaptation accounts at least in part for some of the challenges of offline deep RL, including degradation of performance with prolonged training. Second, we propose a simple and effective *explicit* regularizer for offline value-based RL, DR3, which minimizes the feature similarity between state-action pairs appearing in a bootstrapping update. DR3 is inspired by the theoretical derivation of the implicit regularizer, it alleviates co-adaptation and can be easily combined with modern offline RL methods, such as REM (Agarwal et al., 2020), CQL (Kumar et al., 2020b), and BRAC (Wu et al., 2019). Empirically, using DR3 in conjunction with existing offline RL methods provides about **60%** performance improvement on the harder D4RL (Fu et al., 2020) tasks, and **160%** and **25%** stability gains for REM and CQL respectively, on offline RL tasks in 17 Atari 2600 games. Additionally, we observe large improvements on image-based robotic manipulation tasks (Singh et al., 2020).

2 Preliminaries

The goal in RL is to maximize the long-term discounted reward in an MDP, defined as $(\mathcal{S}, \mathcal{A}, R, P, \gamma)$ (Puterman, 1994), with state space $\mathcal{S}$, action space $\mathcal{A}$, a reward function $R(\mathbf{s}, \mathbf{a})$, dynamics $P(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ and a discount factor $\gamma \in [0, 1)$. The Q-function $Q^\pi(\mathbf{s}, \mathbf{a})$ for a policy $\pi(\mathbf{a}|\mathbf{s})$ is the expected sum of discounted rewards obtained by executing action $\mathbf{a}$ at state $\mathbf{s}$ and following $\pi(\mathbf{a}|\mathbf{s})$ thereafter. $Q^\pi(\mathbf{s}, \mathbf{a})$ is the fixed point of $Q(\mathbf{s}, \mathbf{a}) := R(\mathbf{s}, \mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}' \sim P(\cdot|\mathbf{s},\mathbf{a}), \mathbf{a}' \sim \pi(\cdot|\mathbf{s}')} [Q(\mathbf{s}', \mathbf{a}')]$. We study the offline RL setting, where the algorithm must learn a policy only using a given dataset $\mathcal{D} = \{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$, generated from some behavior policy, $\pi_\beta(\mathbf{a}|\mathbf{s})$, without active data collection. The Q-function is parameterized with a neural net with parameters θ. We will denote the penultimate layer of the deep network (the learned *features*) $\phi_\theta(\mathbf{s}, \mathbf{a})$, such that $Q_\theta(\mathbf{s}, \mathbf{a}) = \mathbf{w}^T \phi(\mathbf{s}, \mathbf{a})$, where $\mathbf{w} \in \mathbb{R}^d$. Standard deep RL methods (Mnih et al., 2015; Haarnoja et al., 2018) convert the Bellman equation into a squared temporal difference (TD) error objective for Q_θ:

$$\mathcal{L}_{\text{TD}}(\theta) = \sum_{\mathbf{s},\mathbf{a},\mathbf{s}' \sim \mathcal{D}} \left(R(\mathbf{s}, \mathbf{a}) + \gamma \overline{Q}_\theta(\mathbf{s}', \mathbf{a}') - Q_\theta(\mathbf{s}, \mathbf{a}) \right)^2, \tag{1}$$

where $\bar{Q}_\theta$ is a delayed copy of same Q-network, referred to as the *target network* and $\mathbf{a}'$ is computed by maximizing the target Q-function at state $\mathbf{s}'$ for Q-learning (i.e., when computing Q^*) and by sampling $\mathbf{a}' \sim \pi(\cdot|\mathbf{s})$ when computing the Q-value Q^π of a policy π.

A major problem in offline RL is the issue of distributional shift between the learned policy and the behavior policy (Levine et al., 2020). Since our goal is to study the effect of implicit regularization in TD-learning and not distributional shift, we build on top of existing offline RL methods in our experiments: CQL (Kumar et al., 2020b), which penalizes erroneous Q-values during training, REM (Agarwal et al., 2020), which utilizes an ensemble of Q-functions, and BRAC (Wu et al., 2019), which applies a policy constraint. An overview of these methods is provided in Appendix E.

3 Implicit Regularization in Deep RL via TD-Learning

While the "deadly-triad" (Sutton & Barto, 2018) suggests that training value function approximators with bootstrapping can lead to divergence, modern deep RL algorithms have been able to successfully combine these properties (van Hasselt et al., 2018). However, making too many TD updates to the Q-function in offline deep RL is known to sometimes lead to performance degradation and unlearning, even for otherwise effective modern algorithms (Fu et al., 2019; Fedus et al., 2020;

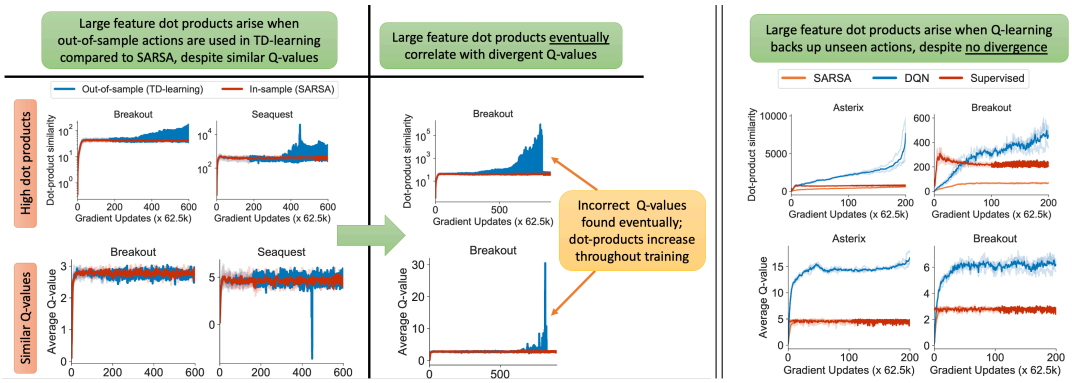


Figure 1: Feature dot-products $\phi(\mathbf{s}, \mathbf{a})^\top \phi(\mathbf{s}', \mathbf{a}')$ increase during training when backing up from *out-of-sample* but in-distribution actions (**TD-learning**: left, **Q-learning**: right), though the average Q-value converges and stays relatively constant. Using only seen state-action pairs for backups (**offline SARSA**) or not performing Bellman backups (i.e., **supervised regression**) avoids this issue, with stable and relatively low dot products. *Left*: TD-learning with high feature dot products eventually destabilizes and produces incorrect Q-values, *Right*: DQN attains extremely large feature dot products, despite a relatively stable trend in Q-values.

Agarwal et al., 2020; Kumar et al., 2021). Such unlearning is not typically observed when training overparameterized models via supervised learning, so what about TD learning is responsible for it? We show that one possible explanation behind this pathology is the implicit regularization induced by minimizing TD error on a deep Q-network. Our theoretical results suggest that this implicit regularization "aliases" the representations of state-action pairs that appear in a Bellman backup, and this aliasing is exacerbated when utilizing unseen state-action pairs for the Bellman backup. Empirically, this typically manifests as "co-adapted" (similar) features for consecutive state-action tuples, even with specialized TD-learning algorithms that account for distributional shift. Highly co-adapted features in turn lead to poor solutions as we will theoretically and empirically show. We first provide empirical evidence of the existence of this co-adaptation phenomenon in Section 3.1 (additional evidence in Appendix A.1) and then theoretically characterize the implicit regularization in TD learning, and discuss how it can explain the co-adaptation phenomenon in Section 3.2.

3.1 Feature Co-Adaptation And How It Relates To Implicit Regularization

In this section, we empirically identify a *feature co-adaptation* phenomenon that appears when training value functions via bootstrapping, where the feature representations of consecutive state-action pairs become excessively similar as measured by their dot product $\phi(\mathbf{s}, \mathbf{a})^\top \phi(\mathbf{s}', \mathbf{a}')$. Feature co-adaptation appears even when there is no explicit affinity to increase feature similarity.

Experimental setup. To empirically observe feature co-adaptation, we ran supervised regression and three variants of approximate dynamic programming (ADP) on an offline dataset consisting of 1% of uniformly-sampled data from the replay buffer of DQN on two Atari games, previously used in Agarwal et al. (2020). First, for comparison, we trained a Q-function via **supervised regression** to Monte-Carlo (MC) return estimates on the offline dataset to estimate the value of the behavior policy. Then, we trained variants of ADP which differ in the selection procedure for the action $\mathbf{a}'$ that appears in the target value in $\mathcal{L}_{\text{TD}}(\theta)$ (Equation 1). The **offline SARSA** variant aims to estimate the value of the behavior policy, Q^{π_β}, and sets $\mathbf{a}'$ to the actual action observed at the next time step in the dataset, such that $(\mathbf{s}', \mathbf{a}') \in \mathcal{D}$. The **TD-learning** variant also aims to estimate the value of the behavior policy, but utilizes an action $\mathbf{a}'$ sampled from the learned behavior policy π_β, $\mathbf{a}' \sim \pi_\beta(\cdot|\mathbf{s}')$. We also train **Q-learning**, which chooses the action $\mathbf{a}'$ to maximize the learned Q-function. While Q-learning learns a different Q-function, we can still compare the relative stability of these methods to gain intuition about the learning dynamics. In addition to feature dot products $\phi(\mathbf{s}, \mathbf{a})^\top \phi(\mathbf{s}', \mathbf{a}')$, we also track the average prediction of the Q-network over the dataset to measure whether the predictions diverge or are stable in expectation.

Observing feature co-adaptation empirically. As shown in Figure 1 (right), the average dot product (top row) between features at consecutive state-action tuples continuously increases for both Q-learning and TD-learning (after enough gradient steps), whereas it flatlines and converges to a small value for supervised regression. We might at first think that this is simply a case of Q-learning failing to converge. However, the bottom row shows that the average Q-values do in fact converge

to a stable value. Despite this, the optimizer drives the network towards higher feature dot products. There is no explicit term in the TD error objective that encourages this behavior, indicating the presence of some implicit regularization phenomenon. This *implicit* preference towards maximizing the dot products of features at consecutive state-action tuples is what we call "feature co-adaptation."

When does feature co-adaptation emerge? Observe in Figure 1 (right) that the feature dot products for offline SARSA converge quickly and are relatively flat, similarly to supervised regression. This indicates that utilizing a bootstrapped update alone is not responsible for the increasing dot-products and instability, because while offline SARSA uses backups, it behaves similarly to supervised MC regression. Unlike offline SARSA, feature co-adaptation emerges for TD-learning, which is surprising as TD-learning also aims to estimate the value of the behavior policy, and hence should match offline SARSA in expectation. The key difference is that while offline SARSA always utilizes actions $\mathbf{a}'$ observed in the training dataset for the backup, TD-learning may utilize potentially unseen actions $\mathbf{a}'$ in the backup, even though these actions $\mathbf{a}' \sim \pi_\beta(\cdot|\mathbf{s}')$ are *within* the distribution of the data-generating policy. This suggests that utilizing **out-of-sample** actions, even when they are not out-of-distribution, in the Bellman backup critically alters the learning dynamics. This finding is absent from literature in offline RL which primarily focuses on the impact of out-of-distribution actions (Levine et al., 2020), but not out-of-sample actions. The theoretical model developed in Section 3.2 will provide an explanation for this observation with a discussion about how feature co-adaption caused due to out-of-sample actions can be detrimental in offline RL.

3.2 Theoretically Characterizing Implicit Regularization in TD-Learning

Why does feature co-adaptation emerge in TD-learning and what do *out-of-sample* actions have to do with it? To answer this question, we theoretically characterize the implicit regularization effects in TD-learning. We analyze the learning dynamics of TD learning in the overparameterized regime, where there are many different parameter vectors θ that fully minimize the training set temporal difference error. When training an overparameterized $f_\theta(\mathbf{x})$ via supervised regression using the squared loss, many different values of θ will satisfy $L(\theta) = 0$ on the training set due to overparameterization, but Blanc et al. (2020) argue that the dynamics of noisy SGD will result in only those fixed points to be stable that additionally satisfy $\nabla_\theta R(\theta^*) = 0$, for a particular *implicit* regularizer $R(\theta)$. The noisy gradient updates analyzed in this model have the form:

$$\theta_{k+1} \leftarrow \theta_k - \eta \nabla_\theta L(\theta) + \eta \varepsilon_k, \;\; \varepsilon_k \sim \mathcal{N}(0, M), \tag{2}$$

and Blanc et al. (2020); Damian et al. (2021) show that when the regression targets are corrupted with $\mathcal{N}(0,1)$ label noise, then $M = \sum_{i=1}^{|\mathcal{D}|} \nabla_\theta f_\theta(\mathbf{x}_i) \nabla_\theta f_\theta(\mathbf{x}_i)^\top$ and the induced implicit regularizer is given by $R(\theta) = \eta \sum_i^{|\mathcal{D}|} ||\nabla_\theta f_\theta(\mathbf{x}_i)||_2^2$. Any solution θ^* found by Equation 2 must satisfy $\nabla_\theta R(\theta^*) = 0$. This model corroborates empirical findings (Mulayoff & Michaeli, 2020) about the solutions found by SGD with deep nets, which motivates our use of this framework. Following this framework, we analyze the fixed points of noisy TD-learning. We consider noisy pseudo-gradient TD updates with a general noise covariance M:

$$\theta_{k+1} = \theta_k - \eta \underbrace{\left(\sum_i \nabla_\theta Q(\mathbf{s}_i, \mathbf{a}_i) \left(Q_\theta(\mathbf{s}_i, \mathbf{a}_i) - (r_i + \gamma Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)) \right) \right)}_{:= g(\theta)} + \eta \varepsilon_k, \;\; \varepsilon_k \sim \mathcal{N}(0, M) \tag{3}$$

We use a deterministic policy $\mathbf{a}'_i = \pi(\mathbf{s}'_i)$ to simplify exposition. Following Blanc et al. (2020), we can set the noise model M as $M = \sum_i \nabla_\theta Q(\mathbf{s}_i, \mathbf{a}_i) \nabla_\theta Q(\mathbf{s}_i, \mathbf{a}_i)^\top$, or utilize a different choice of M, but we will derive the general form first. Let θ^* denote a stationary point of the training TD error, such that the pseudo-gradient $g(\theta^*) = 0$. Further, we denote to the derivative of $g(\theta)$ w.r.t. θ as matrix $G(\theta)$ and refer to as the *pseudo-Hessian*: although $G(\theta)$ is not actually the second derivative of any well-defined objective, since TD updates are not proper gradient updates, as we will see it will play a similar role to the Hessian in gradient descent. For brevity, define $G = G(\theta^*)$, $g = g(\theta^*)$ and $\nabla G = \nabla_\theta G(\theta^*)$. Also, $\lambda_i(P)$ denotes the i-th eigenvalue of matrix P.

Assumptions. To simplify analysis, we assume that matrices G and M (i.e., the noise covariance matrix) span the same n-dimensional basis in d-dimensional space, where d is the number of parameters and n is the number of datapoints, and $n \ll d$ due to overparameterization. We also require θ^* to satisfy a technical criterion that requires approximate alignment between the eigenspaces of G and the gradient of the Q-function, without which noisy TD may not be stable at θ^*. We summarize all the assumptions in Appendix C, and first present our regularizer.

Theorem 3.1 (Implicit regularizer at TD fixed points)**.** *Under the assumptions so far, a fixed point of TD-learning, θ^*, where $Q_{\theta^*}(\mathbf{s}_i, \mathbf{a}_i) = r_i + \gamma Q_{\theta^*}(\mathbf{s}'_i, \mathbf{a}'_i)$ for every $(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i) \in \mathcal{D}$ is stableif: **(1)** it satisfies $\mathrm{Re}(\lambda_i(G)) \geq 0, \forall i$ and $\mathrm{Re}(\lambda_i(G)) > 0$ if $|\mathrm{Imag}(\lambda_i(G))| > 0$, and **(2)** θ^* is the stationary point of the induced implicit regularizer,*

$$R_{\mathrm{TD}}(\theta) = \eta \sum_{i=1}^{|\mathcal{D}|} \nabla Q_\theta(\mathbf{s}_i, \mathbf{a}_i)^\top \Sigma^*_M \nabla Q_\theta(\mathbf{s}_i, \mathbf{a}_i) - \eta\gamma \sum_{i=1}^{|\mathcal{D}|} \mathrm{trace}\left(\Sigma^*_M \nabla Q_\theta(\mathbf{s}_i, \mathbf{a}_i) \left[\left[\nabla Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)^\top\right]\right]\right), \quad (4)$$

where $(\mathbf{s}_i, \mathbf{a}_i)$ and $(\mathbf{s}'_i, \mathbf{a}'_i)$ denote state-action pairs that appear together in a Bellman update, $[[\square]]$ denotes the stop-gradient function, which does not pass partial derivatives w.r.t. θ into $\square$. Σ^*_M is the fixed point of the discrete Lyapunov equation: $\Sigma^*_M := (I - \eta G)\Sigma^*_M (I - \eta G)^\top + \eta^2 M$.

A proof of Theorem 3.1 is provided in Appendix C. Next, we explain the intuition behind this result and provide a proof sketch. To derive the induced implicit regularizer for a stable fixed point θ^* of TD error, we study the learning dynamics of noisy TD learning (Equation 3) initialized at θ^*, and derive conditions under which this noisy update would stay close to θ^* with multiple updates. This gives rise to the two conditions shown in Theorem 3.1 which can be understood as controlling stability in mutually exclusive directions in the parameter space. If condition **(1)** is not satisfied, then even under-parameterized TD will diverge away from θ^*, since $I - \eta G$ would be a non-contraction as the spectral radius, $\rho(I - \eta G) \geq 1$ in that case. Thus, $\theta_k - \theta^*$ will grow or not decrease in some direction. When **(1)** is satisfied for all directions in the parameter space, there are still directions where both the real and imaginary parts of the eigenvalue are 0 due to overparameterization. In such directions, learning is governed by the projection of the noise under the third-order derivative ∇G, which appears in the Taylor expansion of $\theta_k - \theta^*$ around the point θ^*:

$$\theta_{k+1} = \theta_k - \eta\left(g + G(\theta_k - \theta^*) + \frac{1}{2}\nabla G[\theta_k - \theta^*, \theta_k - \theta^*]\right) + \varepsilon_k, \;\; \varepsilon_k \sim \mathcal{N}(0, M) \quad (5)$$

$$\implies \nu_{k+1} = (I - \eta G)\nu_k - \frac{\eta}{2}\nabla G[\nu_k, \nu_k] + \varepsilon_k, \quad (6)$$

where we define $\nu_k := \theta_k - \theta^*$. θ^* is stable if it is a stationary point of the implicit regularizer R_{TD} (condition **(2)**), which ensures that total noise (i.e., accumulated ε_k over iterations k) does not lead to a large deviation in ν_k.

Interpretation of Theorem 3.1. While the choice of the noise model M will change the form of the implicit regularizer, of course, in practice, the form of M is not known. We can consider other choices of M for interpretation, but Theorem 3.1 can be easy to qualitatively interpret for M such that $\Sigma^*_M = I$. In this case, we find that the implicit preference towards local minima of $R_{\mathrm{TD}}(\theta)$ can explain feature co-adaptation. In this case, the regularizer takes a simpler form: $R_{\mathrm{TD}}(\theta) := \sum_i ||\nabla Q_\theta(\mathbf{s}_i, \mathbf{a}_i)||_2^2 - \gamma \nabla Q_\theta(\mathbf{s}_i, \mathbf{a}_i) \nabla [[Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)]]$. The first term is equal to the squared per-datapoint gradient norm, which is same as the implicit regularizer in supervised learning obtained by Blanc et al. (2020); Damian et al. (2021) with label noise. However, $R_{\mathrm{TD}}(\theta)$ additionally includes a second term that is equal to the dot product of the gradient of the Q-function at the current and next states, $\nabla_\theta Q_\theta(\mathbf{s}_i, \mathbf{a}_i)^\top \nabla_\theta Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)$, and thus this term is effectively *maximized*. When restricted to the last-layer parameters of a neural network, this term is equal to the dot product of the features at consecutive state-action tuples: $\sum_i \nabla_\theta Q_\theta(\mathbf{s}_i, \mathbf{a}_i)^\top \nabla_\theta Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i) = \sum_i \phi(\mathbf{s}_i, \mathbf{a}_i)^\top \phi(\mathbf{s}'_i, \mathbf{a}'_i)$. The tendency to maximize this quantity to attain a local minimizer of the implicit regularizer corroborates the empirical findings of increased dot product in Section 3.1.

Explaining the difference between utilizing seen and unseen actions in the backup. If all state-action pairs $(\mathbf{s}'_i, \mathbf{a}'_i)$ appearing on the right-hand-side of the Bellman update also appear in the dataset $\mathcal{D}$, as in the case of offline SARSA (Figure 1), the preference to increase dot products will be balanced by the affinity to reduce gradient norm (first term of $R_{\mathrm{TD}}(\theta)$ when $\Sigma^*_M = I$): for example, when $(\mathbf{s}'_i, \mathbf{a}'_i)$ are permutations of $(\mathbf{s}_i, \mathbf{a}_i)$, R_{TD} is lower bounded by $(1 - \gamma)\sum_i ||\nabla_\theta Q_\theta(\mathbf{x}_i)||_2^2$ and hence minimizing $R_{\mathrm{TD}}(\theta)$ would minimize the feature norm instead of maximizing dot products. Note that this also corresponds to the implicit regularizer we would obtain when training Q-functions via supervised learning and hence, our analysis would predict that offline SARSA with in-sample actions (i.e., when $(\mathbf{s}', \mathbf{a}') \in \mathcal{D}$) would behave similarly to supervised regression.

However, the regularizer behaves very differently when unseen state-action pairs $(\mathbf{s}'_i, \mathbf{a}'_i)$ appear only on the right-hand-side of the backup. This happens with any algorithm where $\mathbf{a}'$ is not the dataset action, which is the case for all deep RL algorithms that compute target values by selecting

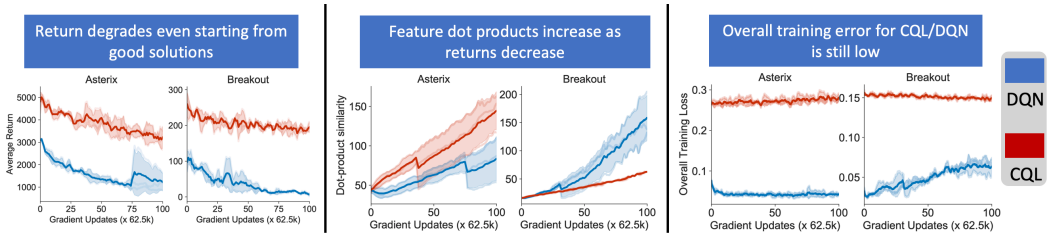


Figure 2: Even when current offline RL algorithms are initialized at a high-performing checkpoint that attains small feature dot products, feature dot products increase with further training and the performance degrades.

$\mathbf{a}'$ according to the current policy. In this case, we expect the dot product of gradients at $(\mathbf{s}, \mathbf{a})$ and $(\mathbf{s}', \mathbf{a}')$ to be large at any attractive fixed point, since this minimizes $R_{\text{TD}}(\theta)$. This is precisely a form of co-adaptation: *gradients at out-of-sample state-action tuples are highly similar to gradients at observed state-action pairs measured by the dot product*. This observation is also supported by the analysis in Section 3.1. Finally, note that the choice of M is a modelling assumption, and to derive our explicit regularizer, later in the paper, we will make a simplifying choice of M, though we empirically verify that a different M also performs well (Appendix A.4).

Why implicit regularization and hence feature co-adaptation can be detrimental to policy performance? To answer this question, we present theoretical and empirical evidence that depicts the adverse effects of this implicit regularizer in TD-learning. **Empirically,** we ran two algorithms, DQN and CQL initialized from a high-performing Q-function checkpoint obtained using DR3 (Section 4), which attains relatively small feature dot products (i.e., the second term of $R_{\text{TD}}(\theta)$ is small). Our goal is to see if TD updates starting from such a "good"initialization still stay around it or diverge to poorer solutions. Our theoretical analysis in Section 3.2 would predict that TD learning would destabilize from such a solution since it would not be a stable fixed point. Indeed, as shown in Figure 2, the policy starts to degrade right from the beginning, and the the dot-product similarities start to increase. This even happens with CQL, which explicitly corrects for any distributional shift confounds, implying that the performance drop cannot be directly explained by the typical out-of-distribution action explanations. To investigate the reasons behind this drop, we also measured the overall training error for these algorithms (i.e., TD error for DQN and TD error + CQL regularizer for CQL) and find in Figure 2 that the loss values generally exhibit a non-increasing trend for both CQL and DQN. In particular, the values attained by TD error are small and do not increase much with more training significantly, indicating that the preference to increase dot products is a consequence of an implicit phenomenon and is not explained by smaller values of the training loss.

To motivate why co-adapted features can lead to poor performance in TD-learning, we study the convergence of linear TD-learning on co-adapted features. Our theoretical result characterizes a lower bound on the feature dot products in terms of the feature norms for state-action pairs in the dataset $\mathcal{D}$, which if satisfied, will inhibit convergence:

Proposition 3.2 (TD-learning on co-adapted features)**.** *Assume that the features $\Phi = [\phi(\mathbf{s}, \mathbf{a})]_{\mathbf{s},\mathbf{a}}$ are used for linear TD-learning. Then, if $\sum_{\mathbf{s},\mathbf{a},\mathbf{s}' \in \mathcal{D}} \phi(\mathbf{s}, \mathbf{a})^\top \phi(\mathbf{s}', \mathbf{a}') \geq \frac{1}{\gamma} \sum_{\mathbf{s},\mathbf{a} \in \mathcal{D}} \phi(\mathbf{s}, \mathbf{a})^\top \phi(\mathbf{s}, \mathbf{a})$, linear TD-learning using features Φ will not converge.*

A proof of Proposition 3.2 is provided in Appendix D and it relies on a stability analysis of linear TD. While features change during training for TD-learning with neural networks, and arguably linear TD is a simple model to study consequences of co-adapted features, even in this simple linear setting, Proposition 3.2 indicates that TD-learning may be non-convergent as a result of co-adaptation.

4 DR3: Explicit Regularization for Deep TD-Learning

Since the implicit regularization effects in TD-learning can lead to feature co-adaptation, which in turn is correlated with poor performance, can we instead derive an *explicit* regularizer to alleviate this issue? Inspired by the analysis in the previous section, we will propose an *explicit* regularizer that attempts to counteract the second term in Equation 4, which would otherwise lead to co-adaptation and poor representations. The *explicit* regularizer that offsets the difference between the two implicit regularizers is given by: $\Delta(\theta) = \sum_i \text{trace}\left[\Sigma_M^{*\top} \nabla_\theta Q_\theta(\mathbf{s}_i, \mathbf{a}_i) \nabla_\theta Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)^\top\right]$, which represents the second term of $R_{\text{TD}}(\theta)$. Note that we drop the stop gradient on $Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)$ in $\Delta(\theta)$ as it performs slightly better in practice (Table A.1), although as shown in that Table, the version with

the stop gradient also significantly improves over the base method. The first term of $R_{\text{TD}}(\theta)$ corresponds to the regularizer from supervised learning. Our proposed method, DR3, simply combines approximations to $\Delta(\theta)$ with various offline RL algorithms. For any offline RL algorithm, ALG, with objective $\mathcal{L}_{\text{ALG}}(\theta)$, the training objective with DR3 is given by: $\mathcal{L}(\theta) := \mathcal{L}_{\text{ALG}}(\theta) + c_0\Delta(\theta)$, where c_0 is the DR3 coefficient. See Appendix E.3 for details on how we tune c_0 in this paper.

Practical version of DR3. In order to practically instantiate DR3, we need to choose a particular noise model M. In general, it is not possible to know beforehand the "correct" choice of M (Equation 3), even in supervised learning, as this is a complicated function of the data distribution, neural network architecture and initialization. Therefore, we instantiate DR3 with two heuristic choices of M: **(i)** M induced by label noise studied in prior work for supervised learning and for which we need to run computationally heavy fixed-point computation for M, and **(ii)** a simpler alternative that sets $\Sigma^*_M = I$. We find that both of these variants generally perform equally well empirically (Figure 6), and so we utilize **(ii)** in practice due to low computational costs. Additionally, because computing and backpropagating through per-example gradient dot products is slow, we instead approximate $\Delta(\theta)$ with the contribution only from the last layer parameters (*i.e.*, $\sum_i \nabla_{\mathbf{w}} Q_\theta(\mathbf{s}_i, \mathbf{a}_i)^\top \nabla_{\mathbf{w}} Q_\theta(\mathbf{s}'_i, \mathbf{a}'_i)$), similarly to tractable Bayesian neural nets. As shown in Appendix A.4, the practical version of DR3 performs similarly to the theoretically derived version.

$$\textbf{Explicit DR3 regularizer}: \qquad \overline{\mathcal{R}}_{\text{exp}}(\theta) = \sum_{i \in \mathcal{D}} \phi(\mathbf{s}_i, \mathbf{a}_i)^\top \phi(\mathbf{s}'_i, \mathbf{a}'_i). \tag{7}$$

5 Related Work

Prior analyses of learning dynamics in RL has focused primarily on analyzing error propagation in tabular or linear settings (*e.g.*, Chen & Jiang, 2019; Duan et al., 2020; Xie & Jiang, 2020; Wang et al., 2021a;b; Farahmand et al., 2010; De Farias, 2002), understanding instabilities in deep RL (Achiam et al., 2019; Bengio et al., 2020; Kumar et al., 2020a; Van Hasselt et al., 2018) and deriving weighted TD updates that enjoy convergence guarantees (Maei et al., 2009; Mahmood et al., 2015; Sutton et al., 2016), but these methods do not reason about implicit regularization or any form of representation learning. Ghosh & Bellemare (2020) focuses on understanding the stability of TD-learning in underparameterized linear settings, whereas our focus is on the overparameterized setting, when optimizing TD error and learning representations via SGD. Kumar et al. (2021) studies the learning dynamics of Q-learning and observes that the rank of the feature matrix, Φ, drops during training. While this observation is related, our analysis characterizes the implicit preference of learning towards feature co-adaptation (Theorem 3.1) on out-of-sample actions as the primary culprit for aliasing. Additionally, utilizing DR3 not only outperforms the srank(Φ) penalty in Kumar et al. (2021) by more than **100%**, but it also alleviates rank collapse, with no apparent term that explicitly increases rank. Somewhat related to DR3, Durugkar & Stone (2018); Pohlen et al. (2018) heuristically constrain gradients of TD to prevent changes in target Q-values to prevent divergence. Contrary to such heuristic approaches, DR3 is inspired from a theoretical model of implicit regularization, and does not prevent changes in target values, but rather reduces feature dot products.

6 Experimental Evaluation of DR3

Our experiments aim to evaluate the extent to which DR3 improves performance in offline RL in practice, and to study its effect on prior observations of rank collapse and how it compares to the more costly regularizer suggested by our analysis, which requires estimating $\Sigma^\star_M$. To this end, we investigate if DR3 improves offline RL performance and stability on three offline RL benchmarks: Atari 2600 games with discrete actions (Agarwal et al., 2020), continuous control tasks from D4RL (Fu et al., 2020), and image-based robotic manipulation tasks (Singh et al., 2020).

Following prior work (Fu et al., 2020; Gulcehre et al., 2020), we evaluate DR3 in terms of final offline RL performance after a given number of iterations. Additionally, we report *training stability*, which is important in practice as offline RL does not admit cheap validation of trained policies for model selection. To evaluate stability, we train for a large number of gradient steps (2-3x longer than prior work) and either report the **average performance** over the course of training or the final performance at the end of training. We expect that a stable method that does not unlearn with more gradient steps, should have better average performance, as compared to a method that attains good peak performance but degrades with more training. See Appendix E for further details.

Table 1: IQM normalized average performance (training stability) across 17 games, with 95% CIs in parenthesis, after 6.5M gradient steps for the 1% setting and 12.5M gradient steps for the 5%, 10% settings. Individual performances reported in Tables F.4-F.10. DR3 improves the stability over both CQL and REM.

Data	CQL	CQL + DR3	REM	REM + DR3
1%	43.7 (39.6, 48.6)	**56.9** (52.5, 61.2)	4.0 (3.3, 4.8)	**16.5** (14.5, 18.6)
5%	78.1 (74.5, 82.4)	**105.7** (101.9, 110.9)	25.9 (23.4, 28.8)	**60.2** (55.8, 65.1)
10%	59.3 (56.4, 61.9)	**65.8** (63.3, 68.3)	53.3 (51.4, 55.3)	**73.8** (69.3, 78)

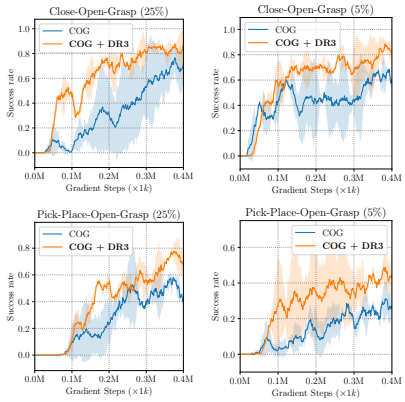


Figure 3: **Performance of DR3 + COG** on two manipulation tasks using only 5% and 25% of the data used by Singh et al. (2020) to make these more challenging. COG + DR3 outperforms COG in training and attains higher average and final performance.

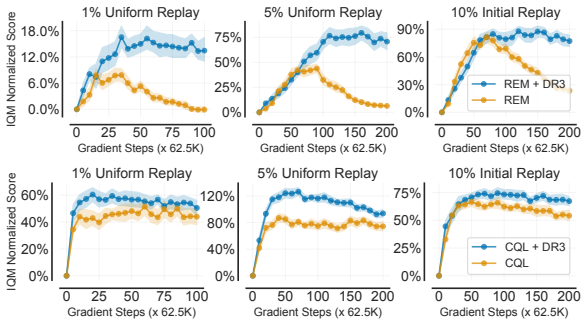
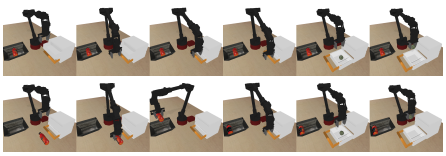


Figure 4: **Normalized performance across 17 Atari games for REM + DR3 (top), CQL + DR3 (bottom)**. x-axis represents *gradient steps*; no new data is collected. While naïve REM suffers from a degradation in performance with more training, REM + DR3 not only remains generally stable with more training, but also attains higher final performance. CQL + DR3 attains higher performance than CQL. We report IQM with 95% stratified bootstrap CIs (Agarwal et al., 2021).

Offline RL on Atari 2600 games. We compare DR3 to prior offline RL methods on a set of offline Atari datasets of varying sizes and quality, akin to Agarwal et al. (2020); Kumar et al. (2021). We evaluated on three datasets: **(1)** 1% and 5% samples drawn uniformly at random from DQN replay; **(2)** a dataset with more suboptimal data consisting of the first 10% samples observed by an online DQN. Following Agarwal et al. (2021), we report the interquartile mean (IQM) normalized scores across 17 games over the course of training in Figure 4 and report the IQM average performance in Table 1. Observe that combining DR3 with modern offline RL methods (CQL, REM) attains the best final and average performance across the 17 Atari games tested on, directly improving upon prior methods across all the datasets. When DR3 is used in conjunction with REM, it prevents unlearning and performance degradation with more training. CQL + DR3 improves by **20%** over CQL on final performance and attains **25%** better average performance. We also compare DR3 to the $\mathrm{srank}(\Phi)$ penalty proposed to counter rank collapse (Kumar et al., 2021). Directly taking median normalized score improvements reported by Kumar et al. (2021), CQL + DR3 improves by over **2x** (31.5%) over naïve CQL relative to the srank penalty (14.1%), indicating DR3's efficacy.

Offline RL on robotic manipulation from images. Next, we aim to evaluate the efficacy of DR3 on two image-based robotic manipulation tasks (Singh et al., 2020) (visualized on the right) that require composition of skills (e.g., opening a drawer, closing a drawer, picking an obstructive object, placing an object, etc.) over extended horizons using only a sparse 0-1 reward. As shown in Figure 3, combining DR3 with COG not only improves over COG, but also learns faster and attains a better average performance.



Offline RL on D4RL tasks. Finally, we evaluate DR3 in conjunction with CQL on the harder D4RL (Fu et al., 2020) domains (antmaze, kitchen domains). To assess if DR3 is stable and able to prevent unlearning that eventually appears in CQL, we trained CQL+DR3 for **6x** longer: 2M steps with 3x higher learning rate. Observe in Table 2, that CQL + DR3 outperforms CQL, in some cases substantially, indicating the ability to prevent unlearning that happens for more gradient steps with DR3. Further, we also compare the effect of adding DR3 to a policy constraint method, BRAC (Wu

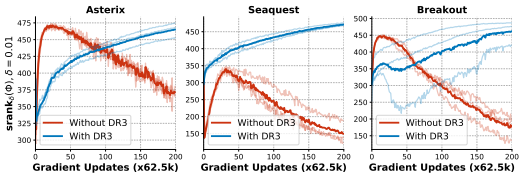


Figure 5: **Trend of effective rank,** $\text{srank}(\Phi)$ of features Φ learned by the Q-function when trained with TD error (red, "Without DR3") and with TD error + DR3 (blue, "With DR3") on three Atari games using the 5% dataset. Note that DR3 alleviates rank collapse observed by Kumar et al. (2021), without explicitly aiming to. Effective rank measures the number of directions with significant singular values (Appendix A.3).

Table 2: **Performance of CQL, CQL + DR3 after 2M gradient steps with a learning rate of 3e-4** for the Q-function averaged over 4 seeds. This is training for **6x** longer compared to CQL defaults. Observe that CQL + DR3 outperforms CQL at 2M steps, indicating is efficacy in inducing stability.

D4RL (-v0) Task	CQL	CQL + DR3
`kitchen-mixed`	14.6 ± 20.5	$\mathbf{37.0 \pm 8.0}$
`kitchen-partial`	29.6 ± 19.6	$\mathbf{43.5 \pm 1.9}$
`kitchen-complete`	22.3 ± 17.5	24.8 ± 15.3
`antmaze-med-diverse`	0.7 ± 0.1	$\mathbf{0.9 \pm 0.1}$
`antmaze-med-play`	0.5 ± 0.4	0.4 ± 0.3
`antmaze-lar-diverse`	0.1 ± 0.0	$\mathbf{0.3 \pm 0.16}$
`antmaze-lar-play`	0.06 ± 0.09	0.1 ± 0.01

et al., 2019). DR3 applied on BRAC improves final median normalized score performance by **13.8** and stability by **8.1** across 15 MuJoCo tasks. Numbers for BRAC can be found in Table F.2.

To summarize, these results indicate that DR3 is a versatile explicit regularizer that improves performance and stability of a wide range of offline RL methods, including conservative methods (e.g, CQL, COG), policy constraint methods (e.g., BRAC) and ensemble-based methods (e.g., REM).

DR3 allows utilizing full capacity as measured via feature ranks. Prior work (Kumar et al., 2021) has shown that implicit regularization can lead to a rank collapse issue in TD-learning, preventing Q-networks from using full capacity. To see if DR3 addresses the rank collapse issue, we follow Kumar et al. (2021) and plot the effective rank of learned features with DR3 in Figure 5. While the value of the effective rank decreases during training with naïve bootstrapping, we find that DR3 addresses this issue allowing the Q-function to use its complete representational capacity, despite no explicit term encouraging this. Additionally, we test the robustness/sensitivity of each layer in the learned Q-network to re-initialization (Zhang et al., 2019) during training and find that DR3 alters the representations trained with TD to behave similarly to supervised learning (Figure A.2).

Comparing explicit regularizers for different choices of noise covariance M. Finally, we investigate the behavior of different implicit regularizers derived via two choices of M in Equation 4 and the corresponding explicit regularizers. While the explicit regularizer we use in practice is a simplifying choice that works well, another choice of M is the covariance matrix induced by label noise, which requires explicit computation of Σ^*_M. Observe in Figure 6 that the explicit regularizers derived for both the choices of M are equally effective. This justifies utilizing our simplified, heuristic choice of setting $\Sigma^*_M = I$ in practice. Results on five Atari games are shown in Appendix A.4.

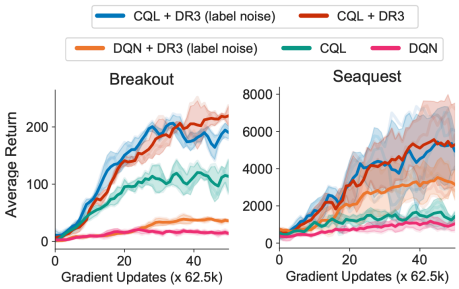


Figure 6: Comparison of the DR3 regularizer corresponding to our simplifying choice of M and M induced by label noise. Note that both of these penalties when applied over CQL improve performance, and generally perform similarly.

7 Discussion

We characterized the implicit preference of TD-learning towards solutions that maximally co-adapt gradients (or features) at consecutive state-action tuples that appear in Bellman backup. This regularization effect is exacerbated when out-of-sample state-action samples are used for the Bellman backup and it can lead to poor policy performance. Inspired by the theory, we propose a practical explicit regularizer, DR3 that aims to counteracts this implicit regularizer. DR3 yields substantial improvements in stability and performance on a wide range of offline RL problems. We believe that understanding the learning dynamics of deep Q-learning and the induced implicit regularization will lead to more robust and stable deep RL algorithms. Furthermore, this understanding can help us to predict the instability issues in value-based RL methods in advance, which can inspire cross-validation and model selection strategies, an important, open challenge in offline RL, for which existing off-policy evaluation techniques are not practically sufficient (Fu et al., 2021). We also note that our analysis does not consider the online RL setting with non-stationary data distributions, and extending our theory and DR3 to online RL is an interesting avenue for future work.

REPRODUCIBILITY STATEMENT

To ensure reproducibility in our empirical results, we provide complete experimental details regarding methods and hyper-parameters used in the Appendix E. Additionally, we follow the recommendations of Agarwal et al. (2021) for reliable evaluation in deep RL and report the statistical uncertainty in reported results including aggregate performance metrics. We provide individual scores in Appendix F and we will release scores for individual runs as well as open-source our code. For theoretical results, we provide explanations of all the assumptions and a complete proof of the claims in Appendix C.

REFERENCES

Joshua Achiam, Ethan Knight, and Pieter Abbeel. Towards characterizing divergence in deep q-learning. *arXiv preprint arXiv:1903.08894*, 2019.

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2020.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. *arXiv preprint arXiv:1802.06509*, 2018.

Emmanuel Bengio, Joelle Pineau, and Doina Precup. Interference and generalization in temporal difference learning. *arXiv preprint arXiv:2003.06350*, 2020.

Guy Blanc, Neha Gupta, Gregory Valiant, and Paul Valiant. Implicit regularization for deep neural networks driven by an ornstein-uhlenbeck like process. In *Conference on learning theory*, pp. 483–513. PMLR, 2020.

Niladri S Chatterji, Behnam Neyshabur, and Hanie Sedghi. The intriguing role of module criticality in the generalization of deep networks. *arXiv preprint arXiv:1912.00528*, 2019.

Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. *ICML*, 2019.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

Alex Damian, Tengyu Ma, and Jason Lee. Label noise sgd provably prefers flat global minimizers. *arXiv preprint arXiv:2106.06530*, 2021.

Daniela Pucci De Farias. *The linear programming approach to approximate dynamic programming: Theory and application*. PhD thesis, 2002.

Yaqi Duan, Zeyu Jia, and Mengdi Wang. Minimax-optimal off-policy evaluation with linear function approximation. In *International Conference on Machine Learning*, pp. 2701–2709. PMLR, 2020.

Ishan Durugkar and Peter Stone. Td learning with constrained gradients. 2018.

Amir-massoud Farahmand, Csaba Szepesvári, and Rémi Munos. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

William Fedus, Prajit Ramachandran, Rishabh Agarwal, Yoshua Bengio, Hugo Larochelle, Mark Rowland, and Will Dabney. Revisiting fundamentals of experience replay. *arXiv preprint arXiv:2007.06700*, 2020.

Justin Fu, Aviral Kumar, Matthew Soh, and Sergey Levine. Diagnosing bottlenecks in deep Q-learning algorithms. *arXiv preprint arXiv:1902.10250*, 2019.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, ziyu wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, Cosmin Paduraru, Sergey Levine, and Thomas Paine. Benchmarks for deep off-policy evaluation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=kWSeGEeHvF8.

Dibya Ghosh and Marc G Bellemare. Representations for stable off-policy reinforcement learning. *arXiv preprint arXiv:2007.05520*, 2020.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gómez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, et al. Rl unplugged: Benchmarks for offline reinforcement learning. 2020.

Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pp. 6151–6159, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018. URL http://arxiv.org/abs/1801.01290.

Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31*. 2018.

Aviral Kumar, Abhishek Gupta, and Sergey Levine. Discor: Corrective feedback in reinforcement learning via distribution correction. *arXiv preprint arXiv:2003.07305*, 2020a.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020b.

Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=O9bnihsFfXU.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Hamid R. Maei, Csaba Szepesvári, Shalabh Bhatnagar, Doina Precup, David Silver, and Richard S. Sutton. Convergent temporal-difference learning with arbitrary smooth function approximation. In *Proceedings of the 22nd International Conference on Neural Information Processing Systems*, 2009.

A Rupam Mahmood, Huizhen Yu, Martha White, and Richard S Sutton. Emphatic temporal-difference learning. *arXiv preprint arXiv:1507.01569*, 2015.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, feb 2015. ISSN 0028-0836.

Rotem Mulayoff and Tomer Michaeli. Unique properties of flat minima in deep networks. In *International Conference on Machine Learning*, pp. 7108–7118. PMLR, 2020.

Tobias Pohlen, Bilal Piot, Todd Hester, Mohammad Gheshlaghi Azar, Dan Horgan, David Budden, Gabriel Barth-Maron, Hado Van Hasselt, John Quan, Mel Večerík, et al. Observe and look further: Achieving consistent performance on atari. *arXiv preprint arXiv:1805.11593*, 2018.

Martin L Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

Avi Singh, Albert Yu, Jonathan Yang, Jesse Zhang, Aviral Kumar, and Sergey Levine. Cog: Connecting new skills to past experience with offline reinforcement learning. *arXiv preprint arXiv:2010.14500*, 2020.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Second edition, 2018.

Richard S. Sutton, A. Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *J. Mach. Learn. Res.*, 17(1):26032631, January 2016. ISSN 1532-4435.

Yuandong Tian, Lantao Yu, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning with dual deep networks. *arXiv preprint arXiv:2010.00578*, 2020.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. *arXiv preprint arXiv:2102.06810*, 2021.

Hado van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *ArXiv*, abs/1812.02648, 2018.

Hado Van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Ruosong Wang, Dean Foster, and Sham M. Kakade. What are the statistical limits of offline {rl} with linear function approximation? In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=30EvkP2aQLD.

Ruosong Wang, Yifan Wu, Ruslan Salakhutdinov, and Sham M Kakade. Instabilities of offline rl with pre-trained neural representation. *arXiv preprint arXiv:2103.04947*, 2021b.

Colin Wei, Jason Lee, Qiang Liu, and Tengyu Ma. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. 2019.

Blake Woodworth, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. *arXiv preprint arXiv:2002.09277*, 2020.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Tengyang Xie and Nan Jiang. Q* approximation schemes for batch reinforcement learning: A eoretical comparison. 2020.

Andrea Zanette. Exponential lower bounds for batch reinforcement learning: Batch rl can be exponentially harder than online rl. *arXiv preprint arXiv:2012.08005*, 2020.

Chiyuan Zhang, Samy Bengio, and Yoram Singer. Are all layers created equal? *arXiv preprint arXiv:1902.01996*, 2019.