Learning from Conditional Distributions via Dual Kernel Embeddings

Bo Dai\textsuperscript{1}, Niao He\textsuperscript{2}, Yunpeng Pan\textsuperscript{1}, Byron Boots\textsuperscript{1}, Le Song\textsuperscript{1}

\textsuperscript{1} Georgia Institute of Technology
\{bodai, ypan37\}@gatech.edu, \{lsong, bboots\}@cc.gatech.edu
\textsuperscript{2} University of Illinois at Urbana-Champaign
niaohe@illinois.edu

July 18, 2016

Abstract

In many machine learning problems, such as policy evaluation in reinforcement learning and learning with invariance, each data point $x$ itself is a conditional distribution $p(z|x)$, and we want to learn a function $f$ which links these conditional distributions to target values $y$. The learning problem becomes very challenging when we only have limited samples or in the extreme case only one sample from each conditional distribution $p(z|x)$. Commonly used approaches either assume that $z$ is independent of $x$, or require an overwhelmingly large sample size from each conditional distribution.

To address these challenges, we propose a novel approach which reformulates the original problem into a min-max optimization problem. In the new view, we only need to deal with the kernel embedding of the joint distribution $p(z,x)$ which is easy to estimate. Furthermore, we design an efficient learning algorithm based on mirror descent stochastic approximation, and establish the sample complexity for learning from conditional distributions. Finally, numerical experiments in both synthetic and real data show that our method can significantly improve over the previous state-of-the-arts.

1 Introduction

In many learning problems, the input $x \in \mathcal{X}$ is continuous and associated with a conditional distribution $p(z|x)$. We want to learn a function $f$ which links the conditional distribution to target $y \in \mathcal{Y}$:

$$\min_{f \in \mathcal{F}} L(f) = \mathbb{E}_{x,y}[\ell(y, \mathbb{E}_{z|x}[f(z,x)])]$$

(1)

where $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$ is a convex loss function, and $f(z,x) = \langle f, \psi(z,x) \rangle$ is a linear function in a suitably chosen (nonlinear) feature space. Here are two examples:

- \textbf{Policy evaluation in reinforcement learning.} Estimating the value function, also called policy evaluation, is recognized as a fundamental task in reinforcement learning. Given a policy $\pi$, the value function $V_\pi(s)$ over the state $s$ of an agent is the fixed point of the Bellman equation $V_\pi(s) = \mathbb{E}_{s',a}[R(s,a) + \gamma V_\pi(s')]$ where $R(s,a)$ is a reward function. The value function can be learned from data by minimizing the mean-square Bellman error (MSBE) [3, 31]:

$$\min_{\pi} \mathbb{E}_s[(V_\pi(s) - \mathbb{E}_{s',a}[R(s,a) + \gamma V_\pi(s')])^2] = \mathbb{E}_{s \sim \mu(s)} \left( \mathbb{E}_{a \sim \pi_b(a|s), s' \sim p(s'|s,a)}[\rho(a|s)\delta(s',a,s)] \right)^2$$

(2)

where $\mu(s)$ is the state distribution, $\pi_b$ the behavior policy, $\rho(a|s) = \frac{\pi_b(a|s)}{\pi_b(a|s)}$ the importance weight, $\delta(s',a,s) = V_\pi(s) - R(s,a) - \gamma V_\pi(s')$, and $p(s'|s,a)$ is the transition of the Markov decision process.
Typically, one parameterizes \( V^\pi(s) \) as a linear function of features defined on \( s \), namely \( V^\pi(s) = (V^\pi, \psi(s)) \). Therefore, this optimization becomes a special case of (1), if we set \( x = s \), \( z = (s', a) \), \( y = 0 \), \( \ell(y, u) = u^2 \), and \( f(z, x) = \delta(s, a, s') = (V^\pi, \psi(s) - \gamma \psi(s')) - R(s, a) \). Due to the online fashion of MDP, we usually observe only one successor state \( s' \) for each action \( a \), i.e., only one sample from the conditional distribution given \( s \).

**Learning with invariance.** Incorporating prior on invariance into learning procedure is crucial for computer vision [24], speech recognition [11] and many other applications. The goal of invariance learning is to estimate a function which minimizes the expected risk while at the same time preserving consistency over a group of operations \( g = \{g_j\}_{j=1}^{\infty} \). [20] shows that this can be accomplished through solving the following optimization problem

\[
\min_{f \in \mathcal{H}} \mathbb{E}_{x,y} [\ell(y, \mathbb{E}_z [\ell(f, \psi(z)) \mid g])] + \left(\nu / 2\right) \|f\|^2_{\mathcal{H}}
\]

where \( \mathcal{H} \) is the reproducing kernel Hilbert space (RKHS) corresponding to some group-invariant Haar-Integral kernel \( k \) and \( \psi(\cdot) \) is the feature map. Clearly, (3) is a special case of (1) with the conditional probability being some normalized Haar measure \( \mu(g(x)) \) given \( x \). Due to computation and memory constraints, in practice, one can only afford to generate a few virtual samples from each data point \( x \).

**Challenges.** Despite the prevalence of learning problems in the form of (1), solving such problem remains very challenging for two reasons: (i) we often have limited samples or in the extreme case only one sample from each conditional distribution \( p(z|x) \), making it difficult to accurately estimate the conditional expectation. (ii) the conditional expectation is nested inside the loss function, making the problem quite different from the traditional stochastic optimization setting. This type of problem is called compositional stochastic programming, where very few results have been established.

**Related work.** A simple option to address (1) is use sample average approximation (SAA), and solve instead \( \min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} [\ell(y_i, \frac{1}{M} \sum_{j=1}^{M} f(z_{ij}, x_i))] \), where \( \{(x_i, y_i)\}_{i=1}^{N} \sim p(x, y) \), and \( \{z_{ij}\}_{j=1}^{M} \sim p(z|x_i) \) for each \( x_i \). To ensure an excess risk of \( \epsilon \), both \( N \) and \( M \) need be at least as large as \( \mathcal{O}(1/\epsilon^2) \), making the overall sample required to be \( \mathcal{O}(1/\epsilon^4) \); see [22, 32] and references therein. Hence, when \( M \) is small, SAA would provide poor results. A second option is resorting to stochastic gradient methods (SGD). One can construct a stochastic estimate of the gradient using \( \nabla f L = \nabla \ell(y, (f, \hat{\phi}(x))) \hat{\phi}(x) \), where \( \hat{\phi}(x) \) is an estimate of \( \mathbb{E}_{z|x}[\psi(z, x)] \) for any \( x \). To ensure convergence, one needs to make the bias of the stochastic gradient small, i.e., a large amount of samples from the conditional distribution is, again, needed.

Another commonly used approach is to first represent the conditional distributions as the so-called kernel conditional embedding, and then perform a supervised learning step on the embedded distribution [28, 10]. Such an approach is not ideal in the sense that it is a two-step procedure, and the estimation of kernel conditional embedding itself is a very difficult task.

A recent work [32], is closely related to but fundamentally distinct from our problem of interest. In [32], the authors consider problems in the form,

\[
\min_{f \in \mathcal{F}} L(f) = \mathbb{E}_y [\ell(y, \mathbb{E}_z [f(z)])]
\]

where \( f(z) = g(z, w) \) is some smooth function parameterized by \( w \in \mathbb{R}^d \). Although the problem allows to handle dependent pairs \( (x, z) \), one should note that solving problem (4) is much easier than our problem of interest (1) since estimating the inner expectation in (1) does not require learning from conditional distribution at all. The authors provide an algorithm that combines stochastic gradient descent with moving average estimation for the inner expectation, and achieves an overall \( \mathcal{O}(1/\epsilon^{3.5}) \) sample complexity for smooth convex loss functions. However, such an algorithm, does not apply to the more general and difficult situation we consider in this paper.
Overview of our approach and contribution. To address the above challenges, we propose an approach called dual kernel embedding. The key idea is to reformulate the problem into a min-max or saddle point problem by utilizing the Fenchel dual of the loss function. We observe that with smooth loss function and continuous conditional distributions, the dual variable becomes a continuous function of $x$. We therefore can parameterize it as a function in the reproducing kernel Hilbert space (RKHS) induced by any universal kernel, where the information about the marginal distribution $p(x)$ and conditional distribution $p(z|x)$ can be aggregated via a kernel embedding of the joint distribution $p(x, z)$. Furthermore, we propose an efficient algorithm based on stochastic approximation to solve the resulting saddle point problem over RKHS spaces, and establish finite-sample analysis of the generic learning problem from conditional distributions.

Compared to previous applicable approaches, a marked advantage of the proposed method is that it requires only one sample from each conditional distribution. Under suitable conditions, the overall sample complexity reduces to $O(1/\epsilon^2)$ in contrast to the $O(1/\epsilon^4)$ complexity required by SAA. As a by-product, even in the degenerate case $4$, this implies an $O(1/\epsilon^2)$ sample complexity when inner function is linear, which already surpasses the result obtained in $32$ and is known to be unimprovable. Furthermore, our algorithm is generic for the family of problem of learning from conditional distribution, and can be adapted to problems with different loss functions and function spaces.

Interestingly, our proposed method also provides some new insight to the related applications. In reinforcement learning settings, our method provides the first algorithm that truly minimizes the mean-square Bellman error with both theoretical guarantees and sample efficiency. We show that some existing algorithms, e.g., gradient-TD2 by $30, 17$, are special cases of our algorithm. In the invariance learning setting, our method also provides a unified view of several existing methods for encoding invariances. Finally, numerical experiments in both synthetic and real data show that our method can significantly improve over the previous state-of-the-arts.

2 Preliminaries

We first briefly introduce our notation on kernel and kernel embedding. Let $\Omega \subset \mathbb{R}^d$ be some input space and $k : \Omega \times \Omega \to \mathbb{R}$ be a positive definite kernel function. We use $k(x, \cdot)$, $k_x$ and $\phi(x)$ interchangeably to denote of the feature map of kernel $k$. Then $k$ induces a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, which has the property $h(x) = \langle h, \phi(x) \rangle_{\mathcal{H}}, \forall h \in \mathcal{H}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the inner product and $\|h\|_{\mathcal{H}}^2 := \langle h, h \rangle_{\mathcal{H}}$ is the norm in $\mathcal{H}$. We denote all continuous functions on $\Omega$ as $C(\Omega)$ and $\| \cdot \|_{\infty}$ as the maximum norm. We call $k$ a universal kernel if $\mathcal{H}$ is dense in $C(\Omega')$ for any compact set $\Omega' \subseteq \Omega$, i.e., for any $\epsilon > 0$ and $u \in C(\Omega')$, there exists $h \in \mathcal{H}$, such that $\|u - h\|_{\infty} \leq \epsilon$. Examples of universal kernel include the Gaussian kernel, $k(x, x') = \exp(-\|x - x'\|_2^2)$, and Laplace kernel, $k(x, x') = \exp(-\|x - x'\|_1)$.

Function approximation using RKHS. Let $\mathcal{H}^\delta := \{ h \in \mathcal{H} : \|h\|_{\mathcal{H}}^2 \leq \delta \}$ be a bounded ball in the RKHS, and we define the error of approximating a continuous function $u$ by a function $h$ in RKHS $\mathcal{H}^\delta$ as

$$E(\delta) := \sup_{u \in C(\Omega)} \inf_{h \in \mathcal{H}^\delta} \|u - h\|_{\infty}.$$  \hspace{1cm} (5)

One can immediately see that $E(\delta)$ decreases as $\delta$ increases and vanishes to zero as $\delta$ goes to infinity. If $C(\Omega)$ is restricted to the set of uniformly bounded continuous functions, then $E(\delta)$ is also bounded. The approximation property, i.e., dependence on $\delta$ remains an open question for general RKHS, but has been carefully established for special kernels. For example, with the kernel $k(x, x') = 1/(1 + \exp(\langle x, x' \rangle))$ induced by the sigmoidal activation function, we have $E(\delta) = O(\delta^{-2/(d+1)} \log(\delta))$ for Lipschitz continuous function space $C(\Omega)$ $27$.

\footnote{The rate is also known to be unimprovable by $9$.}
Hilbert space embedding of distributions. Hilbert space embeddings of distributions are mappings of distributions into potentially infinite dimensional feature spaces \[27\],

\[ \mu_x := \mathbb{E}_x [\phi(x)] = \int_{\Omega} \phi(x)p(x)dx : \mathcal{P} \mapsto \mathcal{H} \]  

where the distribution is mapped to its expected feature map, i.e., to a point in a feature space. Kernel embedding of distributions has rich representational power. Some feature map can make the mapping injective \[29\], meaning that if two distributions, \( p(X) \) and \( q(X) \), are different, they are mapped to two distinct points in the feature space. For instance, when \( X = \mathbb{R}^d \), the feature spaces of many commonly used kernels, such as the Gaussian RBF kernel, will generate injective embedding. We can also embed the joint distribution \( X \) injective \[28\], meaning that if two distributions, \( p(x,y) \) and \( q(x,y) \), are different, they are mapped to two distinct points in a feature space. Based on embedding of joint distributions, kernel embedding of conditional distributions can be defined as \( \mathcal{U}_{z|x} := C_{zz}C_{xx}^{-1} \) as an operator \( \mathcal{H} \mapsto \mathcal{G} \) \[28\].

3 Dual Kernel Embedding of Learning from Conditional Distributions

To solve problem \[1\], we propose a novel and sample-efficient method by leveraging Fenchel duality and kernel embedding to bypass the difficulties with nested expectation and the need for overwhelmingly large sample from conditional distributions.

3.1 Saddle Point Reformulation

Let \( \ell^*_y(\cdot) \) be the convex conjugate of the loss function \( \ell_y(\cdot) := \ell(y, \cdot) \) in \[1\]. That is, \( \ell^*_y(u) = \max_v \{ uv - \ell_y(v) \} \). Invoking the Fenchel duality, our problem of interest can be reformulated as

\[ \min_{f \in \mathcal{F}} \mathbb{E}_{xy} \left[ \max_{u(x)} \left\{ \mathbb{E}_{z|x}[f(z,x)|u(x)] \cdot u(x) - \ell^*_y(u(x)) \right\} \right]. \tag{8} \]

Note that the dual variables \( u(x) \) can be conceived as a (possibly non-continuous) function over \( \Omega \). Based on this observation, we can further rewrite \[8\] as

\[ \min_{f \in \mathcal{F}} \max_{u(.) \in G(\Omega)} \Phi(f,u) := \mathbb{E}_{z|x}[f(z,x) \cdot u(x)] - \mathbb{E}_{xy}[\ell^*_y(u(x))], \tag{9} \]

where \( G(\Omega) = \{ u(.) : \Omega \rightarrow \mathbb{R} \} \). The problem of interest, now reduces to a stochastic saddle point problem, without nested expectation, but with an additional functional dual space to optimize over. Such reformulation has not been exploited in the compositional stochastic programming literature to the best of our knowledge.

By definition, \( \Phi(f,u) \) is always concave in \( u \) for any fixed \( f \). Since \( f(z,x) = \langle f, \psi(z,x) \rangle \) is a linear function, \( \Phi(f,u) \) is also convex in \( f \) for any fixed \( u \). Our reformulation \[9\] is indeed a convex-concave saddle point problem. Nevertheless, optimizing the saddle point problem over a functional dual space remains unfriendly due to the lack of compact representability of the dual function.

3.2 Dual Continuation

Although the reformulation in \[9\] gives us more structure of the problem, it is not yet tractable. This is because the dual function \( u(.) \) can be an arbitrary function which we do not know how to represent efficiently. In the following, we will introduce a tractable representation for \[9\].

First, we will define the function \( u^*(\cdot) : \Omega \rightarrow \mathbb{R} \) as the optimal dual function if for any \( x \in \Omega \),

\[ u^*(x) \in \arg\max_{u \in \mathbb{R}} \left\{ u \cdot \mathbb{E}_{z|x}[f(z,x)] - \ell^*_y(u) \right\}. \]
Note the optimal dual function is well-defined since the optimal set is nonempty. Furthermore, \( u^*(x) \) is related to the conditional distribution via \( u^*(x) \in \partial \ell_{y}(E_{z|x}(f(z,x))) \). This can be simply derived from convexity of loss function and Fenchel’s inequality; see [14] for a more formal argument. Particularly, when \( \ell_{y}(u) = \frac{1}{2}u^2 \), the optimal dual function \( u^*(x) = E_{z|x}(f(z,x)) \). Depending on the property of the loss function \( \ell_{y}(\cdot) \), we can further derive that:

**Proposition 1** If \( \ell_{y}(\cdot) \) is continuously differentiable for any \( y \) and both \( f(z,x) \) and \( p(z|x) \) are continuous in \( x \) for any \( z \), then \( u^*(\cdot) \) is unique and continuous.

For instance, when it comes to the policy evaluation problem in (2), the corresponding optimal dual function is continuous as long as the reward function is continuous. See proofs in Appendix B.2.

The fact that the optimal dual function is a continuous function has interesting consequences. As we mentioned earlier, the space of dual functions can be arbitrary and difficult to represent. Now we can simply restrict the parametrization to the space of continuous functions. It is a smaller and more tractable function space, but it is not missing out the global optimum of the optimization problem in (9). This also provides us a basis for using RKHS to approximate these functions, and simply optimizing over functions in RKHS.

### 3.3 Kernel Embedding

Recall that with the universal kernel, we can approximate any bounded continuous function with reasonably small error. In the rest of the paper, we assume conditions described in Proposition 1 always hold; namely, the optimal dual function is indeed continuous. Thus we approximate the dual space \( \mathcal{G}(\Omega) \) by the bounded RKHS \( \mathcal{H}^b \) induced by a universal kernel \( k \). That is \( u(x) = \langle u, \phi(x) \rangle_{\mathcal{H}} \), and we can rewrite the saddle point problem in (9) as

\[
\min_{f \in \mathcal{F}} \max_{u \in \mathcal{H}^b} \Phi(f, u) = E_{x,z}[\langle f, \psi(z, x), y \rangle_{\mathcal{F}} \cdot \langle u, \phi(x) \rangle_{\mathcal{H}}] - E_{x,y}[\ell_{y}(\langle u, \phi(x) \rangle_{\mathcal{H}})],
\]

where we also assume that the function \( f \) is a linear in some suitably chosen nonlinear feature space \( \mathcal{F} \), and use the joint embedding of \( p(z, x) \) as \( C_{xz} = E_{x,z}[\psi(z, x) \otimes \phi(x)] \). Note that now there is no need to access the conditional distribution \( p(z|x) \), or the conditional embedding operator \( \mathcal{U}_{z|x} \).

The new saddle point approximation (10) based on dual kernel embedding allows us to get away from the fundamental difficulty with insufficient sampling from the conditional distribution. Given a pair of sample \((x, y, z)\), where \((x, y) \sim p(x,y)\) and \(z \sim p(z|x)\), we can now easily construct an unbiased stochastic estimate for the gradient, namely, \( \nabla_{u} \Phi_{x,y,z}(f, u) = \psi(z, x) u(x) \) and \( \nabla_{u} \Phi_{x,y,z}(f, u) = \nabla_{u} \Phi_{x,y,z}(f, u) = [f(z, x) - \nabla \ell_{y}(u(x))] \phi(x) \).

For simplicity of notation, we use \( \nabla \) to denote the subgradient as well as the gradient. With the unbiased stochastic gradient, we are now able to solve the approximation problem (10) by resorting to the powerful mirror descent stochastic approximation framework [22].

### 3.4 Sample-Efficient Algorithm

The algorithm is summarized in Algorithm 1. At each iteration, the algorithm performs a projected gradient step both for the primal variable \( f \) and dual variable \( u \) based on the unbiased stochastic gradient. The proposed algorithm avoids the need for overwhelmingly large sample sizes from the conditional distributions when estimating of the gradient. At each iteration, only one sample from the conditional distribution is required in our algorithm!

Throughout our discussion, we make the following standard assumptions:

**Assumption 1** There exists constant scalars \( C_{\mathcal{F}}, M_{\mathcal{F}}, M \), such that for any \( f \in \mathcal{F}, u \in \mathcal{H}^b \),

\[
E_{z,x}[\|f(z,x)\|_{2}^{2}] \leq M_{\mathcal{F}}, \quad E_{x,z}[\|\psi(z, x)\|_{2}^{2}] \leq C_{\mathcal{F}}, \quad E_{y}[\|\nabla \ell_{y}(u)\|_{2}^{2}] \leq c_{\ell}.
\]

**Assumption 2** There exists constant \( \kappa > 0 \) such that \( k(w, w') \leq \kappa \) for any \( w, w' \in \Omega \).
Assumption 1 and 2 basically suggest that the variance of our stochastic gradient estimate is always bounded. Note that we do not assume any strongly convexity or concavity of the saddle point problem, or Lipschitz smoothness. Hence, we set the output as the average of intermediate solutions weighted by the learning rates \( \gamma_i \), as often used in the literature, to ensure the convergence of the algorithm.

Define the accuracy of any candidate solution \((\bar{f}, \bar{u})\) to the saddle point problem as

\[
\epsilon_{\text{gap}}(\bar{f}, \bar{u}) := \max_{u \in \mathcal{H}^t} \Phi(\bar{f}, u) - \min_{f \in \mathcal{F}} \Phi(f, \bar{u}).
\]

We have the following convergence result.

**Theorem 1** Under Assumptions 1 and 2 the solution \((\bar{f}_t, \bar{u}_t)\) after \(t\) steps of the algorithm with stepsizes being \(\gamma_t = \frac{\gamma}{\sqrt{t}} (\gamma > 0)\) satisfies:

\[
E[\epsilon_{\text{gap}}(\bar{f}_t, \bar{u}_t)] \leq \left[ \frac{(2D_F^2 + 4\delta)/\gamma + \gamma C(\delta, \kappa)}{\sqrt{t}} \right] \frac{1}{\kappa}.
\]

where \(D_F^2 = \sup_{f \in \mathcal{F}} \frac{1}{2} \| f_0 - f \|_2^2 \) and \(C(\delta, \kappa) = \kappa(5M_F + c_\delta) + \frac{1}{8}(\delta + \kappa)^2 C_F \).

The above theorem implies that our algorithm achieves an overall \(O(1/\sqrt{t})\) convergence rate, which is known to be unimprovable already for traditional stochastic optimization with general convex loss function. We further observe that

**Proposition 2** If \(f(z, x)\) is uniformly bounded by \(C\) and \(\ell^*_u(u)\) is uniformly \(K\)-Lipschitz continuous in \(u\), then for any \(y\), \(\Phi(f, u)\) is \((C + K)\)-Lipschitz continuous on \(\mathcal{G}(\Omega)\) with respect to \(\| \cdot \|_\infty\), i.e.

\[
|\Phi(f, u_1) - \Phi(f, u_2)| \leq (C + K)\|u_1 - u_2\|_\infty, \forall u_1, u_2 \in \mathcal{G}(\Omega).
\]

Let \(f_*\) be the optimal solution to (1). Invoking the Lipschitz continuity of \(\Phi\) and using standard arguments of decomposing the objective, we have

\[
L(\bar{f}_t) - L(f_*) \leq \epsilon_{\text{gap}}(\bar{f}_t, \bar{u}_t) + 2(C + K)\mathcal{E}(\delta).
\]

Combining Proposition 2 and Theorem 1 we finally conclude that under the conditions therein,

\[
E[L(\bar{f}_t) - L(f_*)] \leq O\left(\frac{\delta^{3/2}}{\sqrt{t}} + \mathcal{E}(\delta)\right).
\]

There is clearly a delicate tradeoff between the optimization error and approximation error. Using large \(\delta\) increases the optimization error but decreases the approximation error. When \(\delta\) is moderately large (which is expected in the situation when the optimal dual function has small magnitude), our dual kernel embedding algorithm can achieve an overall \(O(1/\epsilon^2)\) sample complexity when solving learning problems in the form of (1). For the analysis details, please refer to Appendix B.

### 4 Extension to Regularized Objective and Random Features

Despite the sample efficiency, there are two potential downsides of the algorithm - the computation overhead caused by projecting to the dual space and the memory requirement for maintaining the dual function. To this end, we introduce the regularized objective, which has been widely used as an alternative to the constrained problem in machine learning literatures, and is proven to be more robust often times in practice.
Motivated by [7], we also provide a doubly stochastic variant of the algorithm, by leveraging random feature to approximate kernel functions.

**Regularized objective.** Instead of solving (10), we consider the alternative reformulation by penalizing the norm of the dual function,

\[
\min_{f \in F} \max_{u \in H} \Phi(f, u) - \frac{\lambda}{2} \|u\|_H^2
\]  

(14)

It is well-known that there is a one-to-one relation between \(\delta\) and \(\lambda\) such that the optimal solutions to (10) and (14) are the same. The objective can also be regarded as a smoothed approximation to the original problem of our interest, see [23]. Problem (14) can be solved efficiently via our Algorithm 1 by simply revoking the projection on the dual space.

**Random feature approximation.** Instead of working with the infinite-dimensional spaces, we could also directly apply random feature approximation [26] to RKHS and reduce it to a \(m\)-dimensional feature space. Let \(\psi(x) = [\psi_{\omega_1}(x); \ldots; \psi_{\omega_m}(x)]\), where \(\omega_1, \ldots, \omega_m\) are i.i.d samples from \(\mathbb{P}(\omega)\). We can further approximate (10) by

\[
\min_{f \in F} \max_{\beta \in \mathbb{R}^m} \Phi(f, \beta^T \psi) - \frac{\mu}{2} \|\beta\|^2
\]  

(15)

Notice that solving (15) will introduce an extra approximation error term in the order of \(O(1/\sqrt{m})\) and thus we do not lose any generalization ability if \(m\) is sufficiently large. The random feature variant of our algorithm can be found in Appendix E.

**Extension to doubly SGD.** Like most kernel methods, our Embedding-SGD algorithm requires to maintain the samples \(\{x_i\}_{i=1}^t\) when representing the dual function. However, this can be avoided by exploiting random features and the “doubly” stochastic gradient idea introduced in [7]. Let \(k(x, x') = \int \psi_{\omega_i}(x) \psi_{\omega_i}(x') d\mathbb{P}(\omega)\) for some nonnegative measure \(\mathbb{P}(\omega)\). By replacing \(\phi(x)\) with random feature map \(\psi_{\omega_i}(x_i) \psi_{\omega_i}(\cdot)\), we maintain a biased estimate of the gradient for reducing the memory cost to \(O(1)\) at each iteration. Although the gradient estimation is no longer unbiased, it can be shown that the algorithm still converges to the same optimal solution in the corresponding RKHS with the optimal rate [7]. The doubly stochastic variant of our algorithm can be found in Algorithm 6 in Appendix F.

**Deep kernel embedding.** While our dual kernel embedding scheme relies on a bounded RKHS w.r.t. \(\|\cdot\|_H\)-norm. The embedding space for continuous functions is fully adapted to other choices of functional spaces, e.g. the convex neural networks with monotonic activation functions under the variation norm, as discussed in [2]. Under such embedding, we could treat the corresponding saddle point approximation with incremental Frank Wolfe algorithms instead of stochastic gradient methods and obtain similar generalization bounds. We leave out the details here since this is beyond our scope.

5 Applications

In this section, we discuss in details how the dual kernel embedding can be applied to solve two important learning problems in machine learning, i.e., policy evaluation in reinforcement learning and learning with invariance, which are the special cases of the optimization [1]. We provide sample-efficient algorithms tailored for the respective learning scenarios. Due to the space limit, we focus only on algorithms with kernel embedding. Extended algorithms with random feature and doubly SGD can be found in Appendix E and F.

5.1 Reinforcement Learning

**Policy evaluation.** Given an MDP represented by \((S, A, P, R)\), the goal is to estimate the value function that minimizes the mean-square Bellman error [2]. We consider the more advanced off-policy setting. Let
\(\mu(s)\) be the state distribution, \(\pi_b\) the behavior policy and \(\rho(a|s) = \frac{\pi(a|s)}{\pi_b(a|s)}\) the importance weight. The learning problem \([2]\) can be rewritten as

\[
\min_{V} L(V) = \mathbb{E}_{s \sim \mu(s)} \left[ \left( \mathbb{E}_{s',a}[\rho(a|s)\delta(s',s)] \right)^2 \right]
\]

(16)

where \(\delta(s',s) = R(s,a) + \gamma V(s') - V(s)\), successor state \(s' \sim P(s'|s,a)\), actions \(a \sim \pi_b(a|s)\). Applying dual kernel embedding, we end up solving the saddle point problem

\[
\min_{V} \max_{\Phi} \Phi(V,u) := \mathbb{E}_{s,s'}[\rho(a|s) (R(s,a) + \gamma V(s') - V(s)) u(s)] - \frac{1}{2} \mathbb{E}[u^2(s)].
\]

(17)

We restrict the dual function to RKHS \(\mathcal{H}\) induced by a universal kernel \(k\) and the value function to RKHS \(\mathcal{H}\) induced by another kernel \(\tilde{k}\). Note that \(k\) and \(\tilde{k}\) are allowed to be different. Let \(\phi(s)\) and \(\tilde{\phi}(s)\) denote the corresponding feature maps. At each iteration, our Embedding-SGD algorithm uses one sample \((s,a,s')\) and updates the solution as follows

\[
V_{i+1} = V_i - \gamma_i \rho(a|s) u_i(s) (\gamma \tilde{\phi}(s') - \tilde{\phi}(s))
\]

\[
u_{i+1} = u_i + \gamma_i [\rho(a|s) \delta_i(s',s) - u_i(s)] \phi(s)
\]

Note that \(\gamma\) is the discount factor, \(\{\gamma_i\}\) are the learning rates. The overall algorithm is summarized in Algorithm 2 in Appendix C.

**Remark.** If we use the same kernel for parametrizing value function and dual function, i.e. \(V(s) = \theta^T \psi(s)\) and \(u(s) = \beta^T \tilde{\psi}(s)\), our saddle point problem (17) reduces to \(\min_{\theta} \|\mathbb{E}_{s,a,s'}[\rho(a|s)\psi(s')]\|^2_{\mathcal{H}[\psi \psi^\top]}^{-1}\). This is exactly the same as the MSPBE objective proposed in [30] of gradient-TD2. In this perspective, gradient-TD2 is simply a special case of our Algorithm 2. However, in view of our theory, there is really no need to set the same kernel for the value and dual functions. As further demonstrated in our experiments, using different kernels can improve the performances. See details in Section 6.1.

Our algorithm is also fundamentally different from the TD algorithm even in the finite state case. The TD algorithm updates the state-value function directly by an estimate of the temporal difference based on one pair of samples, while our algorithm updates the state-value function based on accumulated estimate of the temporal difference, which intuitively is more robust.

### 5.2 Learning with Invariant Representations

**Invariance learning.** The goal is to solve the invariance learning problem \([3]\). Applying the dual kernel embedding, we end up solving the saddle point problem

\[
\min_{f} \max_{u} \Phi(f,u) := \mathbb{E}_{x}[f(z) \cdot u(x)] - \mathbb{E}_{xy}[\ell_x^u(u(x))] + \frac{\nu}{2} \|f\|^2_{\mathcal{H}}.
\]

(18)

where \(\mathcal{H}\) is the RKHS corresponding to the generating kernel for the group-invariant Haar-Integration kernel and \(\mathcal{H}\) is the RKHS to the universal kernel introduced in our method. At each iteration, our algorithm obtains one sample \((x_i,y_i)\) and one virtual sample \(z_i \sim \rho(y(x_i))\), and works as follows:

\[
f_{i+1} = (1 - \nu \gamma_i) f_i - \gamma_i u_i(x_i) \tilde{\phi}(z_i)
\]

\[
u_{i+1} = u_i + \gamma_i [f_i(z_i) - \nabla \ell_x^u(u_i(x_i))] \phi(x_i)
\]

**Remark.** The proposed algorithm bares some similarities to virtual sample techniques \([21, 18]\) in the sense that they both create examples with prior knowledge to incorporate invariance. However, these algorithms are notably different. In fact, the virtual sample technique can be viewed as optimizing an upper bound of the objective \([3]\) by simply moving the conditional expectation outside, while our algorithm directly works on the original objective.

**Remark.** Recall the Haar-Integral kernel can be rewritten as \(\tilde{k}(x,x') = \langle \mathbb{E}_{z|x} [\psi(z)], \mathbb{E}_{z'|x'} [\psi(z')] \rangle_{\mathcal{H}}\), which can be viewed as a special case of Hilbertian metric on probability measures \([13]\). From this perspective,
injecting invariance by Haar-Integration kernel can be understood as learning with distributions over the invariant representations of observations. Other kernels defined for distributions, e.g., the probability product kernel [15], can be also used in incorporating invariance. Robust learning [5] can also be viewed as incorporating invariance prior w.r.t. the perturbation distribution into learning procedure. Therefore, rather than resorting to robust optimization techniques [5, 4], the proposed algorithm for learning with invariance serves as a viable alternative for robust learning.

6 Experiments

We test the proposed algorithm for two applications, i.e., policy evaluation and learning with invariant representation. We focus mainly on our basic kernel embedding algorithm in this section. For full details of our experimental setups, please refer to Appendix D. More experimental results of our extended algorithms using random feature approximation can be found in Appendix G.

6.1 Experiments on Policy Evaluation

We compare the proposed algorithm to several prevailing algorithms in reinforcement learning, including gradient-TD2 (GTD2) [30, 17], residual gradient (RG) [3] and kernel MDP [11] in terms of mean square Bellman error. Unlike online algorithms such as gradient-TD2 and residual gradient, the kernel MDP algorithm requires to visit the entire dataset when estimating the embedding and inner expectation in each iteration. We conduct experiments for policy evaluation on several benchmark datasets, including navigation, cart-pole swing up and PUMC-560 manipulation. We use Gaussian kernel for all algorithms and perform a parameter sweep over all step-size parameters and kernel bandwidth. All algorithms are evaluated based on the average performance of 10 runs.

Navigation. We first justify the proposed algorithm on the navigation experiment in an unbounded room, which extends the discretized MDP in [11] to continuous-state continuous-action MDP. Specifically, the reward is $R(s) = \exp(-100\|s\|^2)$ centered in the middle of the room and $s \sim \mathcal{N}(0, 0.2I)$. We evaluate the deterministic policy $\pi(s) = -0.2sR(s)$, following the gradient of the reward function. The transition distribution follows Gaussian distribution, $p(s'|a, s) = \mathcal{N}(s + a, 0.1I)$. Results are reported in Figure 1(a).

Cart-pole swing up. We consider a cart-pole (CP) system which consist of a cart and a pendulum. The CP is an under-actuated system with only one control act on the cart. The goal is to swing-up the pendulum from the initial position (point down). The reward will be maximum if the pendulum is swing up to $\pi$ angle, please refer to Appendix D for detailed form. We evaluate the linear policy $\pi(s) = As + b$ where $A \in \mathbb{R}^{1 \times 4}$ and $b \in \mathbb{R}^{1 \times 1}$ are policy parameters that are obtained by an external optimizer. Results are reported in Figure 1(b).

Figure 1: Policy evaluation.

![Figure 1](image1.png)

![Figure 1](image2.png)

![Figure 1](image3.png)
PUMA-560 manipulation. PUMA-560 is a robotic arm that has 6 degrees of freedom with 6 actuators on each joint. The task is to steer the end-effector to the desired position and orientation with zero velocity. The reward function is maximum if the arm is located to the desired position, please refer to Appendix D for detailed form. Similarly, we evaluate the linear policy $\pi(s) = As + b$ where $A \in \mathbb{R}^{6 \times 12}$ and $b \in \mathbb{R}^{6 \times 1}$ are policy parameters that are obtained by an external optimizer. Results are reported in Figure 1(c).

In all experiments, the proposed algorithm consistently beats all competitors, including the state-of-the-art algorithm, gradient-TD2, for policy evaluation.

6.2 Experiments on Invariance Learning

To justify the algorithm for learning with invariance, we test the algorithm on two tasks. We first apply the algorithm to robust learning problem where the inputs are contaminated, and then, we conduct comparison on molecular eneretics prediction problem [19]. We compare the proposed algorithm with SGD with virtual samples technique [24, 18] and SGD with finite sample average for inner expectation (SGD-SAA). We also use Gaussian kernel in all tasks. To demonstrate the sample-efficiency of our algorithm, 10 virtual samples are generated for each datum in training phase. The algorithms are terminated after sweeping 10 rounds of whole dataset. We emphasize that the SGD with virtual sample is optimizing an upper bound of the objective. We plot its result with dot line instead.

Noisy measurement. We generate a synthetic dataset by
\[ \bar{x} \sim U([-0.5, 0.5]); \quad x = \bar{x} + 0.05e \]
\[ y = 3\bar{x}^2 + (\sin(3.53\pi\bar{x}) + \cos(7.7\pi\bar{x})) \exp(-1.6\pi|\bar{x}|) + 0.01e, \]
where the contamination $e \sim \mathcal{N}(0, 1)$. Only $(x, y)$ are provided to learning method, while $\bar{x}$ is unknown. The virtual samples are sampled from $z \sim \mathcal{N}(x, 0.05^2)$ for each observation. The dataset is illustrate in Appendix C. The 10 runs average results are illustrated in Figure 2(a). The proposed algorithm achieves average MSE as low as 0.0029 after visit 0.1M data, significantly better than the alternatives.

QuantumMachine. We test the proposed algorithm for learning with invariance task on QuantumMachine 5-fold dataset for atomization energy prediction. We follow the same setting in [19] where the data points are represented by Coulomb matrices, and the virtual samples are generated by random permutation as in [19]. The average results are shown in Figure 2(b). SGD-SAA performs better at early stages. This may be because the error in SAA estimation dominated by the variance in stochastic gradient. While in the proposed algorithm, the dual function is not well-estimated. However, in later stage, the proposed algorithm
achieves a significant better solution, while SGD-SAA stuck in an inferior solution due to the inaccurate inner expectation estimation.

7 Conclusion

We propose a novel sample-efficient algorithm, Embedding-SGD, for addressing machine learning problems involved with inputs given by conditional distributions. Our algorithm benefits from a fresh employ of saddle point and kernel embedding techniques, to mitigate the difficulty with limited samples from conditional distribution as well as the presence of nested expectations. To our best knowledge, among all existing algorithms able to solve such problems, this is the first algorithm that allows to take only one sample at a time from the conditional distribution and comes with provable theoretical guarantee.

We apply the proposed algorithm to solve two fundamental problems in machine learning, i.e., policy evaluation in reinforcement learning and learning with invariance. The proposed algorithm achieves the state-of-the-art performances on these two tasks comparing to the existing algorithms. As we discussed in Appendix [F], the algorithm is also applicable to control problem in reinforcement learning.

In addition to its wide applicability, our algorithm is also very versatile and amenable for all kinds of enhancement. To reduce memory cost and achieve better performance, the algorithm can be easily extended with doubly stochastic gradient trick or back-propagation to embrace random feature embedding or even more complicated nonlinear feature spaces. Some additional experiments are provided in Appendix and more will be carried out in our future work.

Acknowledgements

This project was supported in part by NSF/NIH BIGDATA 1R01GM108341, ONR N00014-15-1-2340, NSF IIS-1218749, and NSF CAREER IIS-1350983.

References

[1] F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio. Unsupervised learning of invariant representations in hierarchical architectures. CoRR, abs/1311.4158, 2013.

[2] F. Bach. Breaking the curse of dimensionality with convex neural networks. CoRR, abs/1412.8690, 2014.

[3] L. Baird. Residual algorithms: reinforcement learning with function approximation. In ICML, 1995.

[4] A. Ben-Tal and L. E. G. A. Nemirovski. Robust Optimization. Princeton University Press, 2008.

[5] C. Bhattacharyya, K. S. Pannagadatta, and A. J. Smola. A second order cone programming formulation for classifying missing data. In NIPS, 2005.

[6] L. Bottou. Online learning and stochastic approximations. On-line learning in neural networks, 17(9):142, 1998.

[7] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In NIPS, 2014.

[8] A. Devinatz. Integral representation of pd functions. Trans. AMS, 74(1):56–77, 1953.

[9] R. DeVore, R. Howard, and C. Micchelli. Optimal nonlinear approximation. Manuscripta Mathematica, 1989.
[10] S. Grunewalder, G. Lever, L. Baldassarre, S. Patterson, A. Gretton, and M. Pontil. Conditional mean embeddings as regressors. In ICML, 2012.

[11] S. Grunewalder, G. Lever, L. Baldassarre, M. Pontil, and A. Gretton. Modeling transition dynamics in MDPs with RKHS embeddings. In ICML, 2012.

[12] M. Hein and O. Bousquet. Kernels, associated structures, and generalizations. Technical Report, 2004.

[13] M. Hein and O. Bousquet. Hilbertian metrics and positive definite kernels on probability measures. In Z. Ghahramani and R. Cowell, editors, AISTATS, 2005.

[14] J.-B. Hiriart-Urruty and C. Lemaréchal. Fundamentals of convex analysis. Springer Science & Business Media, 2012.

[15] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. JMLR, 5:819–844, 2004.

[16] J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. IEEE Transactions on Signal Processing, 52(8), Aug 2004.

[17] B. Liu, J. Liu, M. Ghatamzadeh, S. Mahadevan, and M. Petrik. Finite-sample analysis of proximal gradient td algorithms. In UAI, 2015.

[18] G. Loosli, S. Canu, and L. Bottou. Training invariant support vector machines with selective sampling. In Large Scale Kernel Machines, 2007.

[19] G. Montavon, K. Hansen, S. Fazli, M. Rupp, F. Biegler, A. Ziehe, A. Tkatchenko, A. von Lilienfeld, and K.-R. Müller. Learning invariant representations of molecules for atomization energy prediction. In NIPS, 2012.

[20] Y. Mroueh, S. Voinea, and T. A. Poggio. Learning with group invariant features: A kernel perspective. In NIPS, 2015.

[21] R. Munos. Error bounds for approximate policy iteration. In ICML, 2003.

[22] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. SIAM J. on Optimization, 19(4):1574–1609, Jan. 2009.

[23] Y. Nesterov. Smooth minimization of non-smooth functions. Math. Program., 103(1):127–152, 2005.

[24] P. Niyogi, F. Girosi, and T. Poggio. Incorporating prior knowledge in machine learning by creating virtual examples. Proceedings of IEEE, 86(11):2196–2209, November 1998.

[25] T. J. Perkins and D. Precup. A convergent form of approximate policy iteration. In NIPS, 2002.

[26] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In NIPS, 2009.

[27] A. J. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In ALT, 2007.

[28] L. Song, A. Gretton, and K. Fukumizu. Kernel embeddings of conditional distributions. IEEE Signal Processing Magazine, 2013.

[29] B. Sriperumbudur, A. Gretton, K. Fukumizu, G. Lanckriet, and B. Schölkopf. Injective Hilbert space embeddings of probability measures. In COLT, 2008.

[30] R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvári, and E. Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In ICML, 2009.
[31] R. S. Sutton, H. R. Maei, and C. Szepesvri. A convergent $o(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *NIPS*, 2008.

[32] M. Wang, E. X. Fang, and H. Liu. Stochastic compositional gradient descent: Algorithms for minimizing compositions of expected-value functions. *CoRR* abs/1411.3803, 2014.
Appendix

A Preliminaries: Stochastic Approximation for Saddle Point Problems

Consider the stochastic saddle point (minmax) problem
\[
\min_{x \in X} \max_{y \in Y} \Phi(x, y) = \mathbb{E}[F(x, y, \xi)]
\]
where the expected value function \( f(x, y) \) is convex in \( x \) and concave in \( y \), and domains \( X, Y \) are convex closed. Let \( z = [x, y] \) and \( G(z, \xi) = [\nabla F_x(x, y, \xi); -\nabla F_y(x, y, \xi)] \) be the stochastic gradient for any input point \( z \) and sample \( \xi \). Let \( \| \cdot \| \) be a norm defined on the embedding Hilbert space of \( Z = X \times Y \), and \( D(z, z') := w(z) - w(z') - \nabla w(z')(z - z') \) be a Bregman distance on \( Z \) defined by a 1-strongly convex (w.r.t. the norm \( \| \cdot \| \)) and continuously differentiable function \( w(z) \). For instance, when \( w(z) = \frac{1}{2} \| z \|^2 \), the Bregman distance becomes \( D(z, z') = \frac{1}{2} \| z - z' \|^2 \).

**Mirror descent SA.** The mirror descent stochastic approximation [22] works as follows:
\[
z_i = \arg\min_{z \in Z} \{ D(z, z_i) + \gamma_i G(z_i, \xi_i) \}, \quad i = 1, \ldots, t.
\]
The quality of an approximate solution \( \bar{z} = (\bar{x}, \bar{y}) \) is defined by the error
\[
\epsilon_{\text{gap}}(\bar{x}, \bar{y}) := \max_{y \in Y} \Phi(\bar{x}, y) - \Phi^* + \Phi^* - \min_{x \in X} \Phi(\bar{x}, \bar{y}) = \max_{y \in Y} \Phi(\bar{x}, y) - \min_{x \in X} \Phi(x, \bar{y}),
\]
where \( \Phi^* \) denotes the optimal value. Let \( \bar{z}_t := \sum_{i=1}^t \frac{z_i}{\gamma_i} \), the convergence properties of this weighted averaging solution is as follows.

**Lemma 1** ([22]) Suppose \( \mathbb{E}[\| G(z_i, \xi_i) \|_2^2] \leq M^2, \forall i \), we have
\[
\mathbb{E}[\epsilon_{\text{gap}}(\bar{x}_t, \bar{y}_t)] \leq \frac{2 \max_{z \in Z} D(z, z_1) + \frac{5}{2} M^2 \sum_{i=1}^t \gamma_i^2}{\sum_{i=1}^t \gamma_i}.
\]
In particular, when \( \gamma_i = \frac{2}{\sqrt{t}}, \forall i = 1, \ldots, t \), we have
\[
\mathbb{E}[\epsilon_{\text{gap}}(\bar{x}_t, \bar{y}_t)] \leq (2 \max_{z \in Z} D(z, z_1)/\gamma + \frac{5}{2} M^2 \gamma) \frac{1}{\sqrt{t}}.
\]
Moreover, suppose \( D^2 = \max_{z \in Z} D(z, z_1) \) and \( M^2 \) are known, by setting \( \gamma = \frac{2D}{\sqrt{MD}} \), we further have
\[
\mathbb{E}[\epsilon_{\text{gap}}(\bar{x}_t, \bar{y}_t)] \leq \frac{2\sqrt{5}DM}{\sqrt{t}}.
\]
To summarize, the mirror descent stochastic approximation achieves an \( O(1/\sqrt{t}) \) convergence rate (also known to be unimprovable [22]). Our Embedding-SGD algorithm builds upon on this framework to solve the saddle point approximation problem [10].

B Convergence Analysis for Embedding-SGD

B.1 Decomposition of generalization error

Let \( f_* \) be the optimal solution to our objective. Denote \( \bar{L}(f) = \max_{u \in H^s} \phi(f, u) \). Invoking the Lipschitz continuity of \( \Phi \), \( L(f) - \bar{L}(f) \leq (K + C)\mathcal{E}(\delta), \forall f \). Therefore,
\[
L(\bar{f}_t) - L(f_*) = L(\bar{f}_t) - \bar{L}(\bar{f}_t) + \bar{L}(\bar{f}_t) - \bar{L}(f_*) + \bar{L}(f_*) - L(f_*) \leq \epsilon_{\text{gap}}(\bar{f}_t, u_t) + 2(K + C)\mathcal{E}(\delta).
\]
We present in below our algorithm customized for policy evaluation problem in (16):

C Embedding-SGD for Policy Evaluation and Invariant Learning

B.2 Approximation error

Proof of Proposition 1

The continuity properties of optimal dual function follows directly from the fact that $u^*(x) \in \partial \ell_y(\mathbb{E}_{x}[f(z,x)])$. Suppose $\ell_y$ is differentiable for any $u$, then $u^*(x) = \ell_y'(f(z,x)p(z|x)dz)$. If $f(z,x)$ and $p(z|x)$ is continuous in $x$ for any $z$, then $\mathbb{E}_{z|x}[f(z,x)]$ is continuous in $x$. Since $\ell_y'$ is also continuous, the composition $u^*(x)$ is therefore continuous as well. ■

Moreover, suppose $\ell_y$ is $L$-Lipschitz differentiable, $f(z,x)$ is uniformly $M_f$-Lipschitz continuous, $p(z|x)$ is $M_p$-Lipschitz continuous in $x$. Then

$$|u^*(x_1) - u^*(x_2)| = |\ell_y'(\int f(z,x_1)p(z|x_1)dz) - \ell_y'(\int f(z,x_2)p(z|x_2)dz)|$$

$$\leq L \int |f(z,x_1) - f(z,x_2)|p(z|x_1)dz + L \int |f(z,x_2)| \cdot |p(z|x_1) - p(z|x_2)|dz$$

$$\leq LM_f|x_1 - x_2| + LM_p|x_1 - x_2| \sup_x |f(z,x)|dz$$

If for any $f(z,x)$ is Lebesgue integrable and $\int |f(z,x)|dz$ is uniformly bounded, then $u^*(x)$ is also Lipschitz-continuous.

Now that $u^*(x)$ is continuous, the error of approximating a continuous function $u$ by a function $h$ in RKHS $\mathcal{H}^\delta$ is bounded by $\mathcal{E}(\delta)$.

B.3 Optimization error

Proof of Theorem 1

Our proof builds on results of stochastic approximation discussed in the previous section. Let $M_1$ and $M_2$ be such that for any $f \in \{f_i\}_{i=1}^t$ and $u \in \{u_i\}_{i=1}^t$,

$$\mathbb{E}_{x,y,z}[|\nabla f(x,y)\hat{\Phi}_{x,y,z}(f,u)||^2_{\mathcal{H}}] \leq M_1^2$$

$$\mathbb{E}_{x,y,z}[|\nabla u\hat{\Phi}_{x,y,z}(f,u)||^2_{\mathcal{H}}] \leq M_2^2$$

Then from Lemma 1 we have

$$\mathbb{E}[\epsilon_{\text{gap}}(\bar{f}_t, \bar{y}_t)] \leq \frac{2(D_1^2 + D_2^2) + \frac{\gamma}{2} (M_1^2 + M_2^2) \sum_{i=1}^t \gamma_i}{\sum_{i=1}^t \gamma_i} \tag{19}$$

where $D_1^2 = \sup_{f \in \mathcal{F}} \frac{1}{2}||f_1 - f||^2_{\mathcal{H}}$ and $D_2^2 = \sup_{u \in \mathcal{U}} \frac{1}{2}||u_1 - u||^2_{\mathcal{L}} \leq 2\delta$. It remains to find upper bounds for $M_1$ and $M_2$. Note that since $||k(w, w')||_{\infty} \leq \kappa$ for any $w$ and $w'$,

$$\mathbb{E}[||\nabla u \hat{\Phi}_{x,y,z}(f,u)||^2_{\mathcal{H}}] \leq \kappa \mathbb{E}[||f(z,x) - \nabla \ell_y'(u(x))||^2] \leq 2\kappa (M_F + c_L)$$

Since $u(x) = (u(\cdot), k(x, \cdot))_{\mathcal{H}}$, from Young’s inequality, we have $|u(x)| \leq \frac{1}{\gamma} u^2_{\mathcal{L}} + \frac{1}{\gamma} k(x, \cdot)_{\mathcal{L}} \leq \frac{1}{\gamma} (\delta + \kappa)$, for any $w \in \Omega$.

$$\mathbb{E}_{x,y,z}[||\nabla f \hat{\Phi}_{x,y,z}(f,u)||^2_{\mathcal{H}}] = \mathbb{E}[||\psi(z,x)||^2_{\mathcal{F}} u(x)^2] \leq \frac{1}{\gamma^2} (\delta + \kappa)^2 C_F$$

Plugging in $M_1^2 = 2\kappa (M_F + c_L)$ and $M_2^2 = \frac{1}{\gamma^2} (\delta + \kappa)^2 C_F$ to (19) and setting $\gamma = \gamma/\sqrt{t}$, we arrive at (12). ■

C Embedding-SGD for Policy Evaluation and Invariant Learning

C.1 Policy Evaluation

We present in below our algorithm customized for policy evaluation problem in (16):
Algorithm 2 Embedding-SGD for Policy Evaluation

Input: $\pi(\cdot), \mu, k, \hat{k}$.

1: for $i = 1, \ldots, t$ do
2: Sample $s \sim \mu(s), a \sim \pi_k(a|s), s' \sim p(s'|s, a)$.
3: $V_{i+1} = V_i - \gamma_i \rho(a|s) u_i(s)(\gamma_k(s', \cdot) - \hat{k}(s, \cdot))$.
4: $u_{i+1} = u_i + \gamma_i [\rho(a|s)(R(s, a) + \gamma V_i(s') - V_i(s)) - u_i(s)] k(s, \cdot)$.
5: end for

C.2 Invariant Learning

We present in below our algorithm customized invariant learning problem in \cite{18}:

Algorithm 3 Embedding-SGD for Learning with Invariance

Input: $p(x, y), p(z|x), \nu, k, \hat{k}$.

1: for $i = 1, \ldots, t$ do
2: Sample $(x_i, y_i) \sim p(x, y), z_i \sim p(z|x_i)$.
3: $\ell_{i+1} = (1 - \nu \gamma_i) \ell_i - \gamma_i u(x_i) \hat{k}(z_i, \cdot)$.
4: $u_{i+1} = u_i + \gamma_i [\ell_i(z_i) - \nabla \ell_{y_i}(u_i(x_i))] k(x_i, \cdot)$.
5: end for

D Experimental Details

D.1 Policy Evaluation

We evaluated all the algorithms in terms of mean square Bellman error on the testing states. On each state $s$, the mean square Bellman error is estimated with 100 next states $s'$ samples. We set the $\lambda = 1$ in kernel conditional embedding for kernel MDP.

Navigation. The batch size is set to be 20. The stepsize is set to be $\frac{\eta}{n_0 + \sqrt{t}}$ following the theorem. \{\eta, n_0\} $\in \{0.1, 1.1, 10\}$. We adopted Gaussian kernel and select the best primal and dual kernel bandwidth in range \{0.01, 0.05, 0.1, 0.15, 0.2\}. The $\gamma$ in MDP is set to be 0.9.

Cart–pole swing up. The batch size is set to be 20. The stepsize parameters are chosen in range \{\eta, n_0\} $\in \{0.05, 1, 100\}$. We adopted Gaussian kernel and the primal and dual kernel bandwidth are selected by median trick with coefficient in \{0.1, 0.5, 1, 5\}. The trajectory is finite, so the $\gamma$ is set to be 1. The reward is $R(s) = \frac{1}{2}(s_1^2 + s_2^2 + s_3^2 + 5(s_4 - \pi)^2)$ where the states $s_1, s_2, s_3, s_4$ are the cart position, cart velocity, pendulum velocity and pendulum angular position.

PUMA-560 manipulation. The batch size is set to be 20. Step-size parameters are chosen in range \{\eta, n_0\} $\in \{0.05, 1, 100\}$. We adopted Gaussian kernel and the primal and dual kernel bandwidth are selected by median trick with coefficient in \{0.1, 0.5, 1, 5\}. The trajectory is finite, so the $\gamma$ is set to be 1. The reward is $R(s) = \frac{1}{2}(\sum_{i=1}^{4}(s_i - \frac{\pi}{2})^2 + \sum_{i=5}^{6}(s_i + \frac{\pi}{2})^2 + \sum_{i=7}^{12}s_i^2)$ where $s_1, \ldots, s_6$ and $s_7, \ldots, s_{12}$ are joint angles and velocities, respectively.

D.2 Learning with Invariance

Noisy in measurements. We select the best $\eta \in \{0.1, 1, 10\}$ and $n_0 \in \{1, 10, 100\}$. We use Gaussian kernel for both primal and dual function, whose bandwidth $\sigma$ are selected from \{0.05, 0.1, 0.15, 0.2\}. We set the batch size to be 50. In testing phase, the observation is noisyless.

QuantumMachine. We selected the stepsize parameters $\eta \in \{0.1, 0.5, 1\}$ and $n_0 \in \{100, 1000\}$. We adopted Gaussian kernel whose bandwidth is selected by median trick with coefficient in \{0.1, 0.25, 0.5, 1\}. The batch size is set to be 1000. To illustrate the benefits of sample efficiency, we generated 10 virtual samples in training phase and 20 in testing phase.
E  Dual Random Feature Embeddings

In this section, we specify the customized algorithms for policy evaluation and learning with invariance leveraging random feature to approximate kernel function. For positive definite kernel, \( k(x, x') = \int \phi_w(x)\phi_w(x')dP(w) \) \cite{r12}, where random feature \( \phi_w(x) : \mathcal{X} \rightarrow \mathbb{R} \) from \( L_2(\Omega, \mathbb{P}) \). Therefore, we can approximate the function \( f \in \mathcal{H} \) with Monte-Carlo approximation \( \hat{f} \in \mathcal{H}^m = \{ \sum_{i=1}^m \beta_i \phi_w(\cdot) \} \| \beta \|_2 \leq C \} \) where \( \{ w_i \}_{i=1}^m \) sampled from \( \mathbb{P}(\omega) \) \cite{r20}. With such approximation, we obtain the corresponding dual random feature embeddings variants.

**Remark:** It should be emphasized that although we only specify the algorithms with either kernel or random feature representation for both primal and dual functions, the parametrization choice of dual function is independent to the form of primal function. Therefore, the algorithm can be extended to hybrid setting where one of the primal and dual functions uses kernel representation, while the other one use the random feature representation.

E.1 Policy Evaluation

Denote the random feature for \( \hat{k}(\cdot, \cdot) \) and \( k(\cdot, \cdot) \) as \( \phi_w(\cdot) \) and \( \psi_w(\cdot) \) with respect to distribution \( \mathbb{P}(\omega) \) and \( \mathbb{P}(\omega) \), respectively, we approximate \( V(\cdot) \) and \( u(\cdot) \) by \( \hat{V}(\cdot) = \theta^\top \phi(\cdot) \) and \( \hat{u}(\cdot) = \eta^\top \psi(\cdot) \), where \( \theta \in \mathbb{R}^{m \times 1} \), \( \phi(\cdot) = [\phi_{w1}(\cdot), \phi_{w2}(\cdot), \ldots, \phi_{wm}(\cdot)]^\top \in \mathbb{R}^{m \times 1} \) with \( \{ w_i \}_{i=1}^m \sim \mathbb{P}(\omega) \) and \( \eta \in \mathbb{R}^{p \times 1} \), \( \psi(\cdot) = [\psi_{w1}(\cdot), \psi_{w2}(\cdot), \ldots, \psi_{wm}(\cdot)]^\top \in \mathbb{R}^{p \times 1} \) with \( \{ w_i \}_{i=1}^m \sim \mathbb{P}(\omega) \). By such parametrization, the saddle point optimization corresponding to policy evaluation becomes,

\[
\min_\theta \max_\eta \hat{\Phi}(\theta, \eta) := \mathbb{E}_s[\mathbb{E}_{a'|s} \left[ \rho(a|s)(R(s) + \gamma \theta^\top \phi(s') - \theta^\top \phi(s))\eta^\top \psi(s) \right]] - \frac{1}{2} \mathbb{E}_s[\eta^\top \psi(s)\psi^\top(s)\eta] + \frac{\lambda_1}{2} \| \theta \|^2 - \frac{\lambda_2}{2} \| \eta \|^2.
\]

Therefore, we obtain the variant of the policy evaluation with dual random feature embedding in Algorithm 4.

**Algorithm 4 Random Feature Embedding-SGD for Policy Evaluation**

**Input:** \( \pi(\cdot), \mu(\cdot), \phi(\cdot), \psi(\cdot) \).

1. for \( i = 1, \ldots, t \) do
2. Sample \( s \sim \mu(s), a \sim \pi_b(a|s), s' \sim p(s'|s, a) \).
3. \( \theta_{t+1} = (1 - \gamma_\lambda \lambda_1) \theta_t - \gamma_\rho \rho(a|s)(\gamma \phi(s') - \phi(s))\eta_t \).
4. \( \eta_{t+1} = (1 - \gamma_\lambda \lambda_2) \eta_t + \gamma_i \left[ \rho(a|s)(R(s) + \gamma \theta_t^\top \phi(s') - \theta_t^\top \phi(s))\psi(s) - \psi(s)\psi(s)^\top \eta_t \right] \).
5. end for

E.2 Learning with Invariance

Follow the notations, we approximate the \( f(\cdot) \) and \( u(\cdot) \) by \( \hat{f}(\cdot) = \theta^\top \phi(\cdot) \) and \( \hat{u} = \eta^\top \psi(\cdot) \). Apply the dual random feature embedding to the learning with invariance problem, we obtain the saddle point problem as

\[
\min_\theta \max_\eta \hat{\Phi}(\theta, \eta) := \mathbb{E}_{x,y}[\mathbb{E}_{z|x} \left[ \theta^\top \phi(z)\psi(x)^\top \eta - \ell_y(\eta^\top \psi((x))) \right]] + \frac{\nu}{2} \| \theta \|^2 - \frac{\lambda}{2} \| \eta \|^2.
\]

We achieve the variant of the proposed algorithm with dual random feature embedding for learning with invariance in Algorithm 5.
We tested the algorithm using the same synthetic dataset. The data are illustrated in Figure 4(a), where the blue points are the contaminated data and the red curve is the ground truth function. We followed the same parameter selection procedure in kernel case, while the $2^{10}$ random features are used. The results are shown in Figure 4(b). Compared with kernel embedding, the finite random feature embedding will introduce extra approximation error.
H Reinforcement Learning: Action Control

We can extend the same technique to control problem. Different from the policy evaluation in which the policy is provided, the control problem is trying to learn the optimal policy in terms of reward, i.e., the policy which can achieve the maximum reward.

We define the action-value function \( Q^\pi(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \), which evaluates the value of taking action \( a \) in state \( s \) with policy \( \pi \) for future actions,

\[
Q^\pi(s, a) = \mathbb{E} \left[ \sum_{t=0}^\infty \gamma^t R(s_{t+1}) \bigg| s_0 = s, A_0 = a, \pi \right].
\]

Based on the definition of the \( Q^\pi \) function, we have the Bellman equation as

\[
Q^\pi(s, a) = R(s) + \gamma \mathbb{E}_{s' \sim P(s'|s, a), a' \sim \pi(a'|s')} [Q^\pi(s', a')]|s, a],
\]

therefore, we obtain the optimization as

\[
\min_{Q} L(Q) := \mathbb{E}_{s \sim \mu(s), a \sim \pi(a|s)} \left[ \left( \mathbb{E}_{s', a' \mid s}[R(s) + \gamma Q(s', a')] - Q(s, a) \right)^2 \right]
\]

(22)

whose objective is the same as SARSA. Apply the dual kernel embedding, we obtain

\[
\min_{Q} \max_{u \in \mathcal{H}} \mathbb{E}_{s,a} \left[ \left( \mathbb{E}_{s', a' \mid s}[R(s) + \gamma Q(s', a')] - Q(s, a), u(s, a) \right) \right] - \frac{1}{2} \mathbb{E}_{s,a}[u(s, a)^2],
\]

(24)

which can be solved efficiently. Recall the definition of \( Q \) function, our framework can be also extended to \( \lambda \)-return for the estimation of \( R(s) + \gamma \max_{a'} Q(s', a') \), i.e., \( G(\lambda) = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{t-1} G^{(n)} \) where \( G^{(n)} = \sum_{t=0}^{n} \gamma^t R(s_t) \).

With the proposed \( Q \) function learning procedure, we can achieve the control target by policy iteration in Algorithm 8.
Analogously, we can formulate the objective for computation with respect to the dynamics, $\langle \tau(\cdot, \cdot), f \rangle$. By kernel conditional embedding \[28\], such expectation can be estimated accurately and computed easily. Specifically, given data $\{(s_i, a_i, s'_i)\}_{i=1}^m$, kernel $k: (X \times A) \times (X \times A) \rightarrow \mathbb{R}$ and $\tilde{k}: X \times X \rightarrow \mathbb{R}$, the expectation for $\forall f \in \mathcal{H}$ can be approximated by $\langle \tau((s, a), \cdot), f \rangle$ with $\tau((s, a), \cdot) \in \mathcal{H}$ as
\[
\tau(s, a) = \alpha(s, a) \top \tilde{K}(S', \cdot),
\]
where \( \tilde{K}(S', \cdot) = [\tilde{k}(s'_1, \cdot), \tilde{k}(s'_2, \cdot), \ldots, \tilde{k}(s'_m, \cdot)]^\top \), \( \alpha(s, a) = (K + \lambda mI)^{-1}K((S, A), (s, a)) \), \( K((S, A), (s, a)) = [k((s_1, a_1), (s, a)), k((s_2, a_2), (s, a)), \ldots, k((s_m, a_m), (s, a))]^\top \) and \( K = [k((s_i, a_i), (s_j, a_j))]_{i,j=1}^m \), \( \lambda \) is a regularization parameter.

With the RKHS embedded transition dynamics, we can compute the expectation \( \mathbb{E}_{s' \sim \rho(s'|a,s)}[f(s')] \approx \langle \tau(s,a), f \rangle \). The theoretical property of such estimator is analyzed in [28].

### I.2 Dynamics Random Feature Embeddings

We can leverage the random feature to approximate the kernel function in kernel conditional embedding, which will result the random feature embedding for memory efficiency. Denote the random features for \( \tilde{K} \), we can leverage the random feature to approximate the kernel function in kernel conditional embedding, \( \langle \tilde{k}(\cdot), \tilde{\phi}(\cdot) \rangle \) can be approximated by \( \tau(s,a) \theta \).

### I.3 Policy Evaluation with Dynamics Embeddings

As we introduced, with the kernel and random feature embedded transition dynamics, the expectation can be estimated with linear operator. Instead of plugging the embedded MDP into value iteration for policy evaluation in [11], which is not suitable for dynamics random feature embeddings, we propose new algorithms which apply the (functional) stochastic gradient algorithm to regularized Bellman error (16) with respect to \( V(\cdot) \) in kernel representation or \( \theta \) in random feature representation, we have

\[
\nabla V(\cdot)L = \rho(a|s)(V(s) - (R(s) + \gamma\langle \tau(s,a), V(\cdot) \rangle)(k(s, \cdot) - \gamma \mathbb{E}_{s'|a,s}\[\hat{k}(s', \cdot)\])
\]

\[
\nabla \theta L = \rho(a|s)(\hat{\theta}^\top \phi(s) - (R(s) + \gamma \tau(s,a) \theta)(\phi(s) - \gamma \hat{\tau}(s, a))^\top)
\]

where \( \mathbb{E}_{s'|a,s}[\hat{k}(s', \cdot)] \) can be approximated by \( \tau(a, s) \) [28]. Therefore, plugging the gradient into stochastic gradient results the following two algorithms for kernel/random feature embedded dynamics.

---

**Algorithm 9** SGD with MDP kernel embedding

**Input:** \( \pi(\cdot), \mu(\cdot), \)

1. compute \( \tau = (K + \lambda mI)^{-1} \)
2. for \( i = 1, \ldots, t \) do
3. Sample \( s \sim \mu(s), a \sim \pi_{\theta}(a|s), s' \sim p(s'|s,a) \).
4. \( \hat{v} = K((S, A), (s, a))^\top \tau V_t(S') \).
5. \( \hat{k}(\cdot) = K((S, A), (s, a))^\top \tau \tilde{K}(S', \cdot) \).
6. \( V_{t+1} = (1 - \gamma_i \lambda_1) V_t - \gamma_i \rho(a|s)(V(s') - (R(s) + \gamma \hat{v})(k(s, \cdot) - \gamma \hat{k}(\cdot)). \)
7. end for

---

**Algorithm 10** SGD with MDP random feature embedding

**Input:** \( \pi(\cdot), \mu(\cdot), \phi(\cdot), \psi(\cdot), \)

1. compute \( \tau = (\Psi \Psi^\top + \lambda mI)^{-1}\Psi \Phi^\top \)
2. for \( i = 1, \ldots, t \) do
3. Sample \( s \sim \mu(s), a \sim \pi_{\phi}(a|s), s' \sim p(s'|s,a) \).
4. \( \theta_{i+1} = (1 - \gamma_i \lambda_1) \theta_i - \gamma_i \rho(a|s)(\theta_i^\top \phi(s') - (R(s) + \gamma \psi(s,a)^\top \tau \theta))(\phi(s') - \gamma \psi(s,a)^\top \tau) \).
5. end for