The Inductive Bias of Quantum Kernels

Anonymous Author(s)
Affiliation
Address
email

Abstract

It has been hypothesized that quantum computers may lend themselves well to applications in machine learning. In the present work, we analyze function classes defined via *quantum kernels*. Quantum computers offer the possibility to efficiently compute inner products of exponentially large density operators that are classically hard to compute. However, having an exponentially large feature space renders the problem of generalization hard. Furthermore, being able to evaluate inner products in high dimensional spaces efficiently by itself does not guarantee a quantum advantage, as already classically tractable kernels can correspond to high- or infinite-dimensional reproducing kernel Hilbert spaces (RKHS).

We analyze the spectral properties of quantum kernels and find that we can expect an advantage if their RKHS is low dimensional and contains functions that are hard to compute classically. If the target function is known to lie in this class, this implies a quantum advantage, as the quantum computer can encode this *inductive bias*, whereas there is no classically efficient way to constrain the function class in the same way. However, we show that finding suitable quantum kernels is not easy because the kernel evaluation might require exponentially many measurements.

In conclusion, our message is a somewhat sobering one: we conjecture that quantum machine learning models can offer speed-ups only if we manage to encode knowledge about the problem at hand into quantum circuits, while encoding the same bias into a classical model would be hard. These situations may plausibly occur when learning on data generated by a quantum process, however, they appear to be harder to come by for classical datasets.

1 Introduction

In recent years, much attention has been dedicated to studies of how small and noisy quantum devices [1] could be used for near term applications to showcase the power of quantum computers. Besides fundamental demonstrations [2], potential applications that have been discussed are in quantum chemistry [3], discrete optimization [4] and machine learning (ML) [5–12].

Initiated by the seminal HHL algorithm [13], early work in quantum machine learning (QML) was focused on speeding up linear algebra subroutines, commonly used in ML, offering the perspective of a runtime logarithmic in the problem size [14–17]. However, most of these works have an inverse polynomial scaling of the runtime in the error and it was shown rigorously by Ciliberto et al. [18] that due to the quantum mechanical measurement process a runtime complexity $O(\sqrt{n})$ is necessary for convergence rate $1/\sqrt{n}$.

Rather than speeding up linear algebra subroutines, we focus on more recent suggestions that use a quantum device to define and implement the function class and do the optimization on a classical computer. There are two ways to that: the first are so-called *Quantum Neural Networks* (QNN) or parametrized quantum circuits [5–7] which can be trained via gradient based optimization [5, 19–23]. The second approach is to use a predefined way of encoding the data in the quantum system and

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

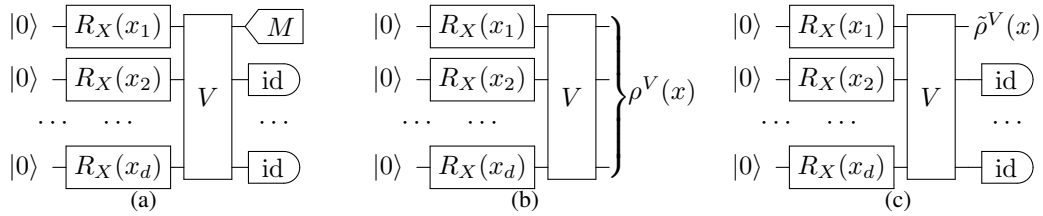

Figure 1: **Quantum advantage via inductive bias:** (a) Data generating quantum circuit $f(x) = \mathrm{Tr}\left[\rho^V(x)(M \otimes \mathrm{id})\right] = \mathrm{Tr}\left[\tilde{\rho}^V(x)M\right]$. (b) The full quantum kernel $k(x, x') = \mathrm{Tr}\left[\rho^V(x)\rho^V(x')\right]$ is too general and cannot learn f efficiently. (c) The biased quantum kernel $q(x, x') = \mathrm{Tr}\left[\tilde{\rho}^V(x)\tilde{\rho}^V(x')\right]$ meaningfully constrains the function space and allows to learn f with little data.

defining a *quantum kernel* as the inner product of two quantum states [7–11]. These two approaches essentially provide a parametric and a non-parametric path to quantum machine learning, which are closely related to each other [11]. Since the optimization of QNNs is non-convex and suffers from so-called Barren Plateaus [24], we here focus on quantum kernels, which allow for convex problems and thus lend themselves more readily to theoretical analysis.

The central idea of using a QML model is that it enables to do computations that are exponentially hard classically. However, also in classical ML, kernel methods allow us to implicitly work with high- or infinite dimensional function spaces [25, 26]. Thus, purely studying the expressivity of QML models [27] is not sufficient to understand when we can expect speed-ups. Only recently first steps where taken into this direction [10, 12, 28]. Assuming classical hardness of computing discrete logarithms, Liu et al. [10] proposed a task based on the computation of the discrete logarithm where the quantum computer, equipped with the right feature mapping, can learn the target function with exponentially less data than any classical (efficient) algorithm. Similarly, Huang et al. [12] analyzed generalization bounds and realized that the expressivity of quantum models can hinder generalization. They proposed a heuristic to optimize the labels of a dataset such that it can be learned well by a quantum computer but not a classical machine.

In this work, we relate the discussion of quantum advantages to the classical concept of *inductive bias*. The *no free lunch* theorem informally states that no learning algorithm can outperform other algorithms on all problems. This implies that an algorithm that performs well on one type of problem necessarily performs poorly on other problems. A standard inductive bias in ML is to prefer functions that are continuous. An algorithm with that bias, however, will then struggle to learn functions that are discontinuous. For a QML model to have an edge over classical ML models, we could thus ensure that it is equipped with an inductive bias that cannot be encoded (efficiently) with a classical machine. If a given dataset fits this inductive bias, the QML model will outperform any classical algorithm. For kernel methods, the qualitative concept of inductive bias can be formalized by analyzing the spectrum of the kernel and relating it to the target function [25, 29–33].

Our main contribution is the analysis of the inductive bias of quantum machine learning models based on the spectral properties of quantum kernels. First, we show that quantum kernel methods will fail to generalize as soon as the data embedding into the quantum Hilbert space is too expressive (Theorem 1). Then we note that projecting the quantum kernel appropriately allows to construct inductive biases that are hard to create classically (Figure 1). However, our Theorem 2 also implies that estimating the biased kernel requires exponential measurements, a phenomenon reminiscent of the Barren plateaus observed in quantum neural networks. Finally we show experiments supporting our main claims.

While our work gives guidance to find a quantum advantage in ML, this yields no recipe for obtaining a quantum advantage on a classical dataset. We conjecture that unless we have a clear idea how the data generating process can be described with a quantum computer, we cannot expect an advantage by using a quantum model in place of a classical machine learning model.

2 Supervised learning

We briefly introduce the setting and notation for supervised learning as a preparation for our analysis of quantum mechanical methods in this context. The goal of supervised learning is the estimation of

a functional mechanism based on data generated from this mechanism. For concreteness we focus on the regression setting where we assume data is generated according to $Y = f^*(X) + \varepsilon$ where ε denotes zero-mean noise. We focus on $X \in \mathcal{X} \subset \mathbb{R}^d$, $Y \in \mathbb{R}$. We denote the joint probability law of (X, Y) by $\mathcal{D}$ and we are given n i.i.d. observations D_n from $\mathcal{D}$. We will refer to the marginal law of X as μ, define the L^2_μ inner product $\langle f, g\rangle = \int f(x)g(x)\,\mu(\mathrm{d}x)$ and denote the corresponding norm by $\|\cdot\|$. The least square risk and the empirical risk of some hypothesis $h : \mathcal{X} \to \mathbb{R}$ is defined by $R(h) = \mathbb{E}_{\mathcal{D}}\left[(h(X) - Y)^2\right]$ and $R_n(h) = \mathbb{E}_{D_n}\left[(h(X) - Y)^2\right]$.

In supervised machine learning, one typically considers a hypothesis space H of functions $h : \mathcal{X} \to \mathbb{R}$ and tries to infer $\operatorname{argmin}_{h\in H} R(h)$ (assuming for simplicity that the minimizer exists). Typically this is done by (regularized) empirical risk minimization $\operatorname{argmin}_{h\in H} R_n(h) + \lambda\Omega(h)$, where $\lambda > 0$ and Ω determine the regularization. The risk of h can then be decomposed in generalization and training error $R(h) = (R(h) - R_n(h)) + R_n(h)$.

Kernel ridge regression. We will focus on solving the regression problem over a reproducing kernel Hilbert space (RKHS) [25, 26]. An RKHS $\mathcal{F}$ associated with a positive definite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is the space of functions such that for all $x \in \mathcal{X}$ and $h \in \mathcal{F}$ the *reproducing* property $h(x) = \langle h, k(x, \cdot)\rangle_{\mathcal{F}}$ holds. Kernel ridge regression regularizes the RKHS norm, i.e., $\Omega(h) = \|h\|^2_{\mathcal{F}}$. With observations $\{(x^{(i)}, y^{(i)})\}_{i=1}^n$ we can compute the kernel matrix $K(X, X)_{ij} = k(x^{(i)}, x^{(j)})$ and the Representer Theorem [34] ensures that the empirical risk minimizer of kernel ridge regression is of the form $\hat{f}^\lambda_n(\cdot) = \sum_{i=1}^n \alpha_i k(x^{(i)}, \cdot)$, with $\alpha = (K(X, X) + \lambda\,\mathrm{id})^{-1}y$. The goal of our work is to study when a (quantum) kernel is suitable for learning a particular problem. The central object to study this is the integral operator.

Spectral properties and inductive bias. For kernel k and marginal law μ, the integral operator K, is defined as $(Kf)(x) = \int k(x, x')f(x')\mu(\mathrm{d}x')$. Mercer's Theorem ensures that there exist a spectral decomposition of K with (possibly infinitely many) eigenvalues γ_i (ordered non-increasingly) and corresponding eigenfunctions ϕ_i, which are orthonormal in L^2_μ, i.e., $\langle\phi_i, \phi_j\rangle = \delta_{i,j}$. We will assume that $\mathrm{Tr}\,[K] = \sum_i \gamma_i = 1$ which we can ensure by rescaling the kernel. We can then write $k(x, x') = \sum_i \gamma_i\phi_i(x)\phi_i(x')$. While the functions ϕ form a basis of $\mathcal{F}$ they might not completely span L^2_μ. In this case we simply complete the basis and implicitly take $\gamma = 0$ for the added functions. Then we can decompose functions in L^2_μ as

$$f(x) = \sum\nolimits_i a_i\phi_i(x). \tag{1}$$

We have $\|f\|^2 = \sum_i a_i^2$ and $\|f\|^2_{\mathcal{F}} = \sum_i \frac{a_i^2}{\gamma_i}$ (if $f \in \mathcal{F}$). Kernel ridge regression penalizes the RKHS norm of functions. The components corresponding to zero eigenvalues are infinitely penalized and cannot be learned since they are not in the RKHS. For large regularization λ the solution $\hat{f}^\lambda_n$ is heavily *biased* towards learning only the coefficients of the principal components (corresponding to the largest eigenvalues) and keeps the other coefficients small (at the risk of *underfitting*). Decreasing the regularization allows ridge regression to also fit the other components, however, at the potential risk of overfitting to the noise in the empirical data. Finding good choices of λ thus balances this *bias-variance* tradeoff.

We are less concerned with the choice of λ, but rather with the spectral properties of a kernel that allow for a quantum advantage. Similar to the above considerations, a target function f can easily be learned if it is well *aligned* with the principal components of a kernel. In the easiest case, the kernel only has a single non-zero eigenvalue and is just $k(x, x') = f(x)f(x')$. Such a construction is arguably the simplest path to a quantum advantage in ML.

Example 1 (Trivial Quantum Advantage)**.** Let f be a scalar function that is easily computable on a quantum device but requires exponential resources to approximate classically. Generate data as $Y = f(X) + \epsilon$. The kernel $k(x, x') = f(x)f(x')$ then has an exponential advantage for learning f from data.

To go beyond this trivial case, we introduce two qualitative measures to judge the quality of a kernel for learning the function f. The *kernel target alignment* of Cristianini et al. [30] is

$$A(k, f) = \frac{\langle k, f \otimes f\rangle}{\langle k, k\rangle^{1/2}\,\langle f \otimes f, f \otimes f\rangle^{1/2}} = \frac{\sum_i \gamma_i a_i^2}{(\sum_i \gamma_i^2)^{1/2}\sum_i a_i^2} \tag{2}$$

and measures how well the kernel fits f. If $A = 1$, learning reduces to estimating a single real parameter, whereas for $A = 0$, learning is infeasible. We note that the kernel target alignment also weighs the contributions of f depending on the corresponding eigenvalue, i.e., the alignment is better if large $|a_i|$ correspond to large γ_i. The kernel target alignment was used extensively to optimize kernel functions [31] and recently also used to optimize quantum kernels [35].

In a similar spirit, the *task-model alignment* of Canatar et al. [32] measures how much of the signal of f is captured in the first i principal components: $C(i) = \sum_{j \leq i} a_j^2 (\sum_j a_j^2)^{-1}$. The slower $C(i)$ approaches 1, the harder it is to learn as the target function is more spread over the eigenfunctions.

3 Quantum computation in machine learning

In this section we introduce hypothesis spaces containing functions whose output is given by the result of a quantum computation. For a general introduction to concepts of quantum computation we refer to the book of Nielsen and Chuang [36].

We will consider quantum systems comprising $d \in \mathbb{N}$ qubits. Discussing such systems and their algebraic properties does not require in-depth knowledge of quantum mechanics. A *pure state* of a single qubit is described by vector $(\alpha, \beta)^\top \in \mathbb{C}^2$ s.t. $|\alpha|^2 + |\beta|^2 = 1$ and we write $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$, where $\{|0\rangle, |1\rangle\}$ forms the computational basis. A d qubit pure state lives in the tensor product of the single qubit state spaces, i.e., it is described by a normalized vector in $\mathbb{C}^{2^d}$. A *mixed state* of a d-qubit system can be described by a density operator $\rho \in \mathbb{C}^{2^d \times 2^d}$, i.e., a positive definite matrix ($\rho \geq 0$) with unit trace ($\mathrm{Tr}[\rho] = 1$). For a pure state $|\psi\rangle$ the corresponding density operator is $\rho = |\psi\rangle \langle\psi|$ (here, $\langle\psi|$ is the complex conjugate transpose of $|\psi\rangle$). A general density operator can be thought of as a classical probabilistic mixture of pure states. We can extract information from ρ by estimating (through repeated measurements) the expectation of a suitable *observable*, i.e., a Hermitian operator $M = M^\dagger$ (where the adjoint $(\cdot)^\dagger$ is the complex conjugate of the transpose), as

$$\mathrm{Tr}[\rho M]. \tag{3}$$

Put simply, the potential advantage of a quantum computer arises from its state space being exponentially large in the number of qubits d, thus computing general expressions like (3) on a classical computer is exponentially hard. However, besides the huge obstacles in building quantum devices with high fidelity, the fact that the outcome of the quantum computation (3) has to be estimated from measurements often prohibits to easily harness this power. We will discuss this in the context of quantum kernels in Section 4.

We consider parameter dependent quantum states $\rho(x) = U(x)\rho_0 U^\dagger(x)$ that are generated by evolving the initial state ρ_0 with the data dependent unitary transformation $U(x)$ [7, 11]. Most often we will without loss of generality assume that the initial state is $\rho_0 = (|0\rangle \langle 0|)^{\otimes d}$. We then define quantum machine learning models via observables M of the data dependent state

$$f_M(x) = \mathrm{Tr}\left[U(x)\rho_0 U^\dagger(x) M\right] = \mathrm{Tr}[\rho(x) M]. \tag{4}$$

In the following we introduce the two most common function classes suggested for quantum machine learning. We note that there also exist proposals that do not fit into the form of Eq. (4) [27, 35, 37].

Quantum neural networks. A "quantum neural network" (QNN) is defined via a *variational quantum circuit* (VQC) [6, 38, 39]. Here the observable M_θ is parametrized by $p \in \mathbb{N}$ classical parameters $\theta \in \Theta \subseteq \mathbb{R}^p$. This defines a parametric function class $\mathcal{F}_\Theta = \{f_{M_\theta} | \theta \in \Theta\}$. The most common ansatz is to consider $M_\theta = U(\Theta) M U^\dagger(\Theta)$ where $U(\Theta) = \prod_i U(\theta_i)$ is the composition of unitary evolutions each acting on few qubits. For this and other common models of the parametric circuit it is possible to analytically compute gradients and specific optimizers for quantum circuits based on gradient descent have been developed [5, 19–23]. Nevertheless, the optimization is usually a non-convex problem and suffers from additional difficulties due to oftentimes exponentially (in d) vanishing gradients [24]. This hinders a theoretical analysis. Note that the non-convexity does not arise from the fact that the QNN can learn non-linear functions, but rather because the observable M_θ depends non-linearly on the parameters. In fact, the QNN functions are linear in the *fixed* feature mapping $\rho(x)$. Therefore the analogy to classical neural networks is somewhat incomplete.

Quantum kernels. The class of functions we consider are RKHS functions where the kernel is expressed by a quantum computation. The key observation is that (4) is linear in $\rho(x)$. Instead of optimizing over the parametric function class $\mathcal{F}_\Theta$, we can define the nonparametric class of functions $\mathcal{F} = \{f_M | f_M(\cdot) = \mathrm{Tr}\left[\rho(\cdot)M\right], M = M^\dagger\}$.[1] To endow this function class with the structure of an RKHS, observe that the expression $\mathrm{Tr}\left[\rho_1\rho_2\right]$ defines a scalar product on density matrices. We then define kernels via the inner product of data-dependent density matrices:

Definition 1 (Quantum Kernel [7, 8, 11])**.** Let $\rho : x \mapsto \rho(x)$ be a fixed feature mapping from $\mathcal{X}$ to density matrices. Then the corresponding *quantum kernel* is $k(x, x') = \mathrm{Tr}\left[\rho(x)\rho(x')\right]$.

The Representer Theorem [34] reduces the empirical risk minimization over the exponentially large function class $\mathcal{F}$ to an optimization problem with a set of parameters whose dimensionality corresponds to the training set size. Since the ridge regression objective is convex (and so are many other common objective functions in ML), this can be solved efficiently with a classical computer.

In the described setting, the quantum computer is only used to estimate the kernel. For pure state encodings, this is done by inverting the data encoding transformation (taking its conjugate transpose) and measuring the probability that the resulting state equals the initial state ρ_0. To see this we use the cyclic property of the trace $k(x, x') = \mathrm{Tr}\left[\rho(x)\rho(x')\right] = \mathrm{Tr}\left[U(x)\rho_0 U^\dagger(x)\, U(x')\rho_0 U^\dagger(x')\right] = \mathrm{Tr}\left[\left(U^\dagger(x')U(x)\rho_0 U^\dagger(x)U(x')\right)\ \rho_0\right]$. If $\rho_0 = (|0\rangle\langle 0|)^{\otimes d}$, then $k(x, x')$ corresponds to the probability of observing every qubit in the '0' state after the initial state was evolved with $U^\dagger(x')U(x)$. To evaluate the kernel, we thus need to estimate this probability from a finite number of measurements. For our theoretical analysis we work with the exact value of the kernel and in our experiments we also simulate the full quantum state. We discuss the difficulties related to measurements in Sec. 4.

4 The inductive bias of simple quantum kernels

We now study the inductive bias for simple quantum kernels and their learning performance. We first give a high level discussion of a general hurdle for quantum machine learning models to surpass classical methods and then analyze two specific kernel approaches in more detail.

Continuity in classical machine learning. Arguably the most important bias in nonparametric regression are continuity assumptions on the regression function. This becomes particularly apparent in, e.g., nearest neighbour regression or random regression forests [40] where the regression function is a weighted average of close points. Here we want to briefly discuss two potential barriers that might prevent major improvements of quantum machine learning methods compared to classical methods. First there is a long list of results concerning the minimax optimality of kernel methods for regression problems [41–43]. In particular these results show that asymptotically the convergence of kernel ridge regression of, e.g., Sobolev functions reaches the statistical limits which also apply to any quantum method. A second barrier concerns the approximation of quantum methods by classical methods. Suppose we are given a quantum kernel $k(x, x')$ on $\mathbb{R}^d$ that is continuous (so that functions in the corresponding RKHS are continuous) and translation invariant. In this case it is known that the learning performance of k can be well approximated using random Fourier features [44] where ε^{-1} features (up to logarithmic terms) are necessary to get the same risk as kernel ridge regression up to $O(\varepsilon + n^{-1/2})$ [45]. Moreover random Fourier feature regression can be efficiently implemented (in polynomial time) on a classical computer. Therefore efficient classical learning algorithms with similar performance as the quantum model exist in this case. It might, however, be difficult to construct the classical model for a given quantum circuit. For non-translation invariant kernels (which are more relevant in quantum machine learning) random feature methods are less studied but there are some extensions which might be applicable in similar settings [46].

A simple quantum kernel. We now restrict our attention to rather simple kernels to facilitate a theoretical analysis. As indicated above we consider data in $\mathcal{X} \subset \mathbb{R}^d$ and we assume that the distribution μ of the data factorizes over the coordinates (i.e. μ can be written as $\mu = \bigotimes \mu_i$). This data is embedded in a d-qubit quantum circuit. Let us emphasize here that the RKHS based on a quantum state of d-qubits is at most 4^d dimensional, i.e., finite dimensional and in the infinite data

[1] $\mathcal{F}$ is defined for a fixed feature mapping $x \mapsto \rho(x)$. Although M is finite dimensional and thus $\mathcal{F}$ can be seen as a parametric function class, we will be interested in the case where M is exponentially large in d and we can only access functions from $\mathcal{F}$ implicitly. Therefore we refer to it as nonparametric class of functions.

limit $n \to \infty$ standard convergence guarantees from parametric statistics apply. Here we consider growing dimension $d \to \infty$, and sample size polynomial in the dimension $n = n(d) \in \text{Poly}(d)$. In particular the sample size $n \ll 4^d$ will be much smaller than the dimension of the feature space and bounds from the parametric literature do not apply.

Here we consider embeddings where each coordinate is embedded into a single qubit using the map φ_i followed by an arbitrary unitary transformation V, so that we can express the embedding in the quantum Hilbert space as $|\psi(x)\rangle = V \bigotimes |\varphi_i(x_i)\rangle$ with corresponding density matrix (feature map) $\rho(x) = |\psi(x)\rangle\langle\psi(x)|$. Note that the corresponding kernel $k(x,x') = \text{Tr}[\rho(x)\rho(x')]$ is independent of V and factorizes $k(x,x') = \text{Tr}[\bigotimes \rho_i(x_i) \bigotimes \rho_i(x_i')] = \prod \text{Tr}[\rho_i(x_i)\rho_i(x_i')]$ where $\rho_i(x_i) = |\varphi_i(x_i)\rangle\langle\varphi_i(x_i)|$. The product structure of the kernel allows us to characterize the RKHS generated by k based on the one dimensional case. The embedding of a single variable can be parametrized by complex valued functions $a(x)$, $b(x)$ as

$$|\varphi_i(x)\rangle = a(x)|0\rangle + b(x)|1\rangle. \tag{5}$$

One important object characterizing this embedding turns out to bet the mean density matrix of this embedding given by $\rho_{\mu_i} = \int \rho_i(y)\,\mu_i(\mathrm{d}y) = \int |\varphi_i(y)\rangle\langle\varphi_i(y)|\,\mu_i(\mathrm{d}y)$. This can be identified with the kernel mean embedding of the distribution [47]. Note that for factorizing base measure μ the factorization $\rho_\mu = \bigotimes \rho_{\mu_i}$ holds. Let us give a concrete example to clarify the setting, see Fig. 1(b).

Example 2. [11, Example III.1.] We consider the cosine kernel where $a(x) = \cos(x/2)$, $b(x) = i\sin(x/2)$. This embedding can be realized using a single quantum $R_X(x) = \exp\left(-i\frac{x}{2}\sigma_x\right)$ gate such that $|\psi(x)\rangle = R_X(x)|0\rangle = \cos(x/2)|0\rangle + i\sin(x/2)|1\rangle$. In this case the kernel is given by

$$k(x,x') = |\langle 0|R_X^\dagger(x)R_X(x)|0\rangle|^2 = |\cos(\tfrac{x}{2})\cos(\tfrac{x'}{2}) + \sin(\tfrac{x}{2})\sin(\tfrac{x'}{2})|^2 = \cos(\tfrac{x-x'}{2})^2. \tag{6}$$

As a reference measure μ we consider the uniform measure on $[-\pi,\pi]$. Then the mean density matrix is the completely mixed state $\rho_\mu = \frac{1}{2}\,\mathrm{id}$. For $\mathbb{R}^d$ valued data whose coordinates are encoded independently the kernel is given by $k(x,x') = \prod \cos^2\left((x_i - x_i')/2\right)$ and $\rho_\mu = 2^{-d}\mathrm{id}_{2^d\times 2^d}$. We emphasize that due to the kernel trick this kernel can be evaluated classically in runtime $O(d)$.

Quantum RKHS. We now characterize the RKHS and the eigenvalues of the integral operator for quantum kernels. The RKHS consists of all function $f \in \mathcal{F}$ that can be written as $f(x) = \text{Tr}[\rho(x)M]$ where $M \in \mathbb{C}^{2^d\times 2^d}$ is a Hermitian operator. Using this characterization of the finite dimensional RKHS we can rewrite the infinite dimensional eigenvalue problem of the integral operator as a finite dimensional problem. The action of the corresponding integral operator on f can be written as

$$\begin{aligned}(Kf)(x) &= \int f(y)k(y,x)\,\mu(\mathrm{d}y) = \int \text{Tr}[M\rho(y)]\,\text{Tr}[\rho(y)\rho(x)]\;\mu(\mathrm{d}y) \\ &= \int \text{Tr}[(M\otimes\rho(x))(\rho(y)\otimes\rho(y))]\;\mu(\mathrm{d}y) = \text{Tr}\left[(M\otimes\rho(x))\int \rho(y)\otimes\rho(y)\,\mu(\mathrm{d}y)\right]\end{aligned} \tag{7}$$

We denote the operator $O_\mu = \int \rho(y)\otimes\rho(y)\,\mu(\mathrm{d}y)$ for which $\text{Tr}[O_\mu] = 1$ holds. Then we can write

$$(Kf)(x) = \text{Tr}[O_\mu(M\otimes\rho(x))] = \text{Tr}[O_\mu(M\otimes\mathrm{id})(\mathrm{id}\otimes\rho(x))] = \text{Tr}[\text{Tr}_1[O_\mu(M\otimes\mathrm{id})]\,\rho(x)].$$

The eigenvalues of K can now be identified with the eigenvalues of the linear map T_μ mapping $M \to \text{Tr}_1[O_\mu(M\otimes\mathrm{id})]$. As shown in the appendix there is an eigendecomposition such that $T_\mu(M) = \sum \lambda_i A_i \text{Tr}[A_i M]$ where A_i are orthonormal Hermitian matrices (for details, a proof and an example we refer to Appendix C). The eigenfunctions of K are given by $f_i(x) = \text{Tr}[\rho(x)A_i]$.

We now state a bound that controls the largest eigenvalue of the integral operator K in terms of the eigenvalues of the mean density matrix ρ_μ (Proof in Appendix C.2).

Lemma 1. *The largest eigenvalue γ_{max} of K satisfies the bound $\gamma_{max} \leq \sqrt{Tr\left[\rho_\mu^2\right]}$.*

The lemma above shows that the squared eigenvalues of K are bounded by $\text{Tr}\left[\rho_\mu^2\right]$, an expression known as the *purity* [36] of the density matrix ρ_μ, which measures the diversity of the data embedding. On the other hand the eigenvalues of K are closely related to the learning guarantees of kernel ridge regression. In particular, standard generalization bounds for kernel ridge regression [48] become vacuous when γ_{max} is exponentially smaller than the training sample size (if $\text{Tr}[K] = 1$ which holds for pure state embeddings). The next result shows that this is not just a matter of bounds.

Theorem 1. *Suppose the purity of the embeddings ρ_{μ_i} satisfies Tr $\left[\rho_{\mu_i}^2\right] \leq \delta < 1$ and training sample size satisfies $n \leq d^k$. Then there exists $d_0 = d_0(\delta, k, \varepsilon)$ such that for all $d \geq d_0$ no function can be learned using kernel ridge regression with the d-qubit kernel $k(x, x') =$ Tr $[\rho(x)\rho(x')]$ in the sense that for any f, with probability at least $1 - \varepsilon$ for all λ*

$$R(\hat{f}_n^\lambda) \geq (1 - \varepsilon)\|f\|^2. \tag{8}$$

Theorem 1 (Proof in Appendix D) implies that generalization is only possible when the mean embedding of most coordinates is close to a pure state, i.e. the embedding $x \to |\varphi_i(x)\rangle$ is almost constant. To make learning from data feasible we cannot use the full expressive power of the quantum Hilbert space but instead only very restricted embeddings allow to learn from data. This generalizes an observation already made in [12]. Since also classical methods allow to handle high-dimensional and infinite dimensional RKHS the same problem occurs for classical kernels where one solution is to adapt the bandwidth of the kernel to control the expressivity of the RKHS. In principle this is also possible in the quantum context, e.g., for the cosine embedding.

Biased kernels. We have discussed that without any inductive bias, the introduced quantum kernel cannot learn any function for large d. One suggestion to reduce the expressive power of the kernel is the use of *projected* kernels [12]. They are defined using reduced density matrices given by $\tilde{\rho}_m^V(x) = \text{Tr}_{m+1\ldots d}\left[\rho^V(x)\right]$ where $\text{Tr}_{m+1\ldots d}[\cdot]$ denotes the partial trace over qubits $m+1$ to d (definition in Appendix A) . Then they consider the usual quantum kernel for this embedding $q_m^V(x, x') = \text{Tr}\left[\tilde{\rho}_m^V(x)\tilde{\rho}_m^V(x)\right]$. Physically, this corresponds to just measuring the first m qubits and the functions f in the RKHS can be written in terms of a hermitian operator M acting on m qubits so that $f(x) = \text{Tr}\left[\rho^V(x)(M \otimes \text{id})\right] = \text{Tr}\left[\tilde{\rho}_m^V(x)M\right]$. If V is sufficiently complex it is assumed that f is hard to compute classically [49].

Indeed above procedure reduces the generalization gap. But this comes at the price of an increased *approximation error* if the remaining RKHS cannot fully express the target function f^* anymore, i.e., the learned function *underfits*. Without any reason to believe that the target function is well represented via the projected kernel, we cannot hope for a performance improvement by simply reducing the size of the RKHS in an arbitrary way. However, if we *know* something about the data generating process than this can lead to a meaningful inductive bias. For the projected kernel this could be that we *know* that the target function can be expressed as $f^*(x) = \text{Tr}\left[\tilde{\rho}_m^V(x)M^*\right]$, see Fig. 1. In this case using q_m^V improves the generalization error without increasing the approximation error. To emphasize this, we will henceforth refer to q_m^V as *biased kernel*.

We now investigate the RKHS for reduced density matrices where V is a Haar-distributed random unitary matrix (proof in Appendix E).

Theorem 2. *Suppose V is distributed according to the Haar measure on the group of unitary matrices. Fix m. Then the following two statements hold:*

a) The reduced density operator satisfies with high probability $\tilde{\rho}_m^V = 2^{-m}\text{id} + O(2^{-d/2})$ and the projected kernel satisfies with high probability $q_m^V(x, x') = 2^{-m} + O(2^{-d/2})$ as $d \to \infty$.

b) Let $T_{\mu,m}^V$ denote the linear integral operator for the kernel q_m^V as defined above. Then the averaged operator $\mathbb{E}_V\left[T_{\mu,m}^V\right]$ has one eigenvalue $2^{-m} + O(2^{-d})$ whose eigenfunction is constant (up to higher order terms) and $2^m - 1$ eigenvalues $2^{-m-d} + O(2^{-2d})$.

The second result for the averaged kernel is not in itself meaningful as this operator has no direct meaning. However, we expect a similar result to hold with high probability for fixed V, but a proof would require the evaluation of 8-th order polynomials over the unitary group. Nevertheless the result indicates that for generic rotations V, learning with the biased kernel q_m^V is possible as soon as the training sample size satisfies $n \geq \dim(\mathcal{F})$ (which is bounded by $\dim(\mathcal{F}) \leq 4^m$).

Let us now focus on the case $m = 1$, that is the biased kernel is solely defined via the first qubit. Assuming that Theorem 2b) also holds for fixed V we can assume that the biased kernel has the form

$$q(x, x') \equiv q_1^V(x, x') = \gamma_0\phi_0(x)\phi_0(x') + \sum\nolimits_{i=1}^{3} \gamma_i\phi_i(x)\phi_i(x'), \tag{9}$$

where $\gamma_0 = 1/2 + \mathcal{O}(2^{-d})$ and $\phi_0(x) = 1$ is the constant function up to exponentially small terms. For $i = 1, 2, 3$ we have $\gamma_i = \mathcal{O}(2^{d-1}) = \mathcal{O}(2^{-d})$ (Fig. 2) and ϕ_i is a function that conjectured to be

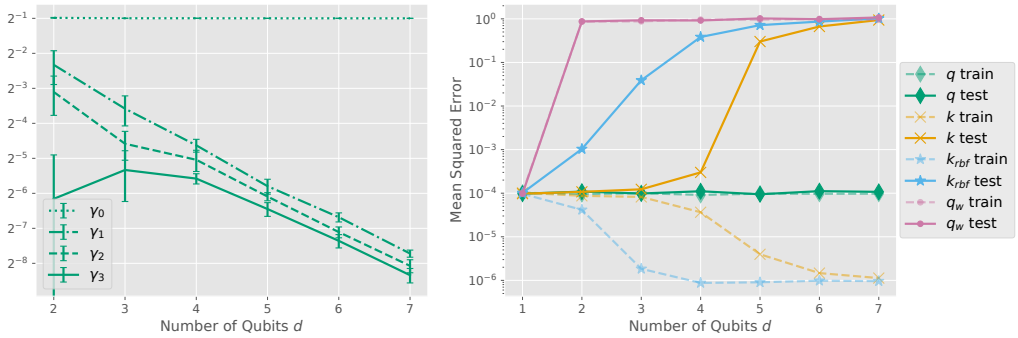

Figure 2: **Left:** Spectral behavior of biased kernel q, see Theorem 2b) and Equation (9) **Right:** The biased kernel q, equipped with prior knowledge, easily learns the function for arbitrary number of qubits and achieves optimal mean squared error (MSE). Models that are ignorant to the structure of f^* fail to learn the function.

exponentially hard in d to compute classically [49]. It is thus impossible to include a bias towards those three eigenfunctions classically. On the other hand we can include a strong bias towards the constant eigenfunction also classically. The straightforward way to do this is to center the data in the RKHS (see Appendix B for details).

Barren plateaus. Another conclusion from Theorem 2a) is that the fluctuations of the reduced density matrix around its mean are exponentially vanishing in the number of qubits. In practice the value of the kernel would be determined by measurements and exponentially many measurements are necessary to obtain exponential accuracy of the kernel function. Therefore the theorem suggests that it is not possible in practice to learn anything beyond the constant function from generic biased kernels for (modestly) large values of d. This observation is closely related to the fact that for many quantum neural networks architectures, the gradient of the parameters with respect to the loss is exponentially vanishing with the system size d, a phenomenon known as *Barren plateaus* [24, 50].

5 Experiments

Since for small d we can simulate the biased kernel efficiently, we illustrate our theoretical findings in the following experiments. Our implementation, building on standard open source packages [51, 52], is available in the supplementary information. We consider the case described above where we *know* that the data was generated by measuring an observable on the first qubit, i.e., $f^*(x) = \mathrm{Tr}\left[\tilde{\rho}_1^V(x)M^*\right]$, but we do not know M^*, see Fig. 1. We use the full kernel k and the biased kernel q for the case $m = 1$. To show the effect of selecting the wrong bias, we also include the behavior of a biased kernel defined only on the second qubit, denoted as q_w. As a classical reference we also include the performance of a radial basis function kernel $k_{\mathrm{rbf}}(x, x') = \exp(-\|x - x'\|^2/2)$. For the experiments we fix a single qubit observable $M^* = \sigma_z$ and perform the experiment for varying number d of qubits. First we draw a random unitary V. The dataset is then generated by drawing $N = 200$ realizations $\{x^{(i)}\}_{i=1}^N$ from the d dimensional uniform distribution on $[0, 2\pi]^d$. We then define the labels as $y^{(i)} = cf^*(x^{(i)}) + \epsilon^{(i)}$, where $f^*(x) = \mathrm{Tr}\left[\tilde{\rho}^V(x)\sigma_z\right]$, $\epsilon^{(i)}$ is Gaussian noise with $\mathrm{Var}[\epsilon] = 10^{-4}$, and c is chosen such that $\mathrm{Var}[f(X)] = 1$. Keeping the variances fixed ensures that we can interpret the behavior for varying d.

We first verify our findings from Theorem 2b) and Equation (9) by estimating the spectrum of q. Fig. 2 (left) shows that Theorem 2b) also holds for individual V with high probability. We then use $2/3$ of the data for training kernel ridge regression (we fit the mean seperately) with preset regularization, and use $1/3$ to estimate the test error. We average the results over ten random seeds (random V, $x^{(i)}, \epsilon^{(i)}$) and results are reported in the right panel of Fig. 2. This showcases that as the number of qubits increases, it is impossible to learn f^* without the appropriate spectral bias. k and k_{rbf} have to little bias and overfit, whereas q_w has the wrong bias and severly underfits. The performance of q_w underlines that randomly biasing the kernel does not significantly improve the

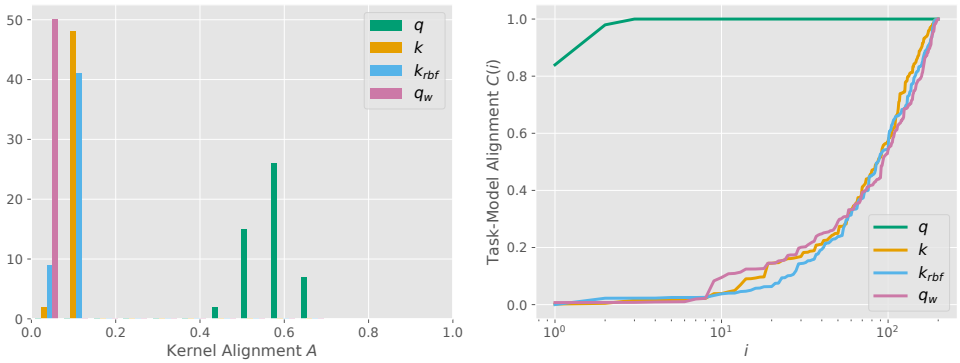

Figure 3: Kernel target alignment (left) and task model alignment (right) for $d = 7$.

performance over the full kernel k. In the appendix we show that this is not due to a bad choice of regularization, by reporting cherry-picked results over a range of regularizations.

To further illustrate the spectral properties, we empirically estimate the kernel target alignment [30] and the task-model alignment [32] that we introduced in Sec. 2. By using the centered kernel matrix (see App. B) and centering the data we can ignore the first eigenvalue in (9) corresponding the constant function. In Figure 3 (left) we show the empirical (centered) kernel target alignment for 50 random seeds. The biased kernel is the only one well aligned with the task. The right panel of Fig. 3 shows the task model alignment. This shows that f^* can be completely expressed with the first four components of the biased kernel, while the other kernels essentially need the entire spectrum (we use a sample size of 200, hence the empirical kernel matrix is only 200 dimensional) and thus are unable to learn. Note that the kernel q_w is four dimensional, and so higher contributions correspond to functions outside its RKHS that it actually cannot even learn at all.

6 Discussion

We provided an analysis of the reproducing kernel Hilbert space (RKHS) and the inductive bias of quantum kernel methods. Rather than the dimensionality of the RKHS, its spectral properties determine whether learning is feasible. Working with exponentially large RKHS comes with the risk of having a correspondingly small inductive bias. This situation indeed occurs for naive quantum encodings, and hinders learning unless datasets are of exponential size. To enable learning, we necessarily need to consider models with a stronger inductive bias. Encoding a bias towards continuous functions is likely not a promising path for a quantum advantage, as this is where classical machine learning models excel.

Our results suggest that we can only achieve a quantum advantage if we *know* something about the data generating process and cannot efficiently encode this classically, yet are able use this information to bias the quantum model. We indeed observe an exponential advantage in the case where we know that the data comes from a single qubit observable and constrain the RKHS accordingly. However, we find that evaluating the kernel requires exponentially many measurements, an issue related to Barren Plateaus encountered in quantum neural networks.

With fully error-corrected quantum computers it becomes feasible to define kernels with a strong bias that do not require exponentially many measurements. An example of this kind was recently presented by Liu et al. [10]: Here one knows that the target function is extremely simple after computing the discrete logarithm. A quantum computer is able to encode this inductive bias by using an efficient algorithm for computing the discrete logarithm.

However, even for fully coherent quantum computers it is unclear how we can reasonably encode a strong inductive bias about a classical dataset (e.g., images of cancer cells, climate-data, etc.). The situation might be better when working with *quantum data*, i.e., data that is collected via observations of systems at a quantum mechanical scale. To summarize, we conclude that there is no indication that quantum machine learning will substantially improve supervised learning on classical datasets.

References

[1] John Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.

[2] Frank Arute, Kunal Arya, Ryan Babbush, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[3] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5:4213, 2014.

[4] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv:1411.4028*, 2014.

[5] Kosuke Mitarai, Makoto Negoro, Masahiro Kitagawa, and Keisuke Fujii. Quantum circuit learning. *Phys. Rev. A*, 98:032309, 2018.

[6] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. *arXiv:1802.06002*, 2018.

[7] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209, 2019.

[8] Maria Schuld and Nathan Killoran. Quantum machine learning in feature Hilbert spaces. *Phys. Rev. Lett.*, 122:040504, 2019.

[9] Jonas M. Kübler, Krikamol Muandet, and Bernhard Schölkopf. Quantum mean embedding of probability distributions. *Phys. Rev. Research*, 1:033159, 2019.

[10] Yunchao Liu, Srinivasan Arunachalam, and Kristan Temme. A rigorous and robust quantum speed-up in supervised machine learning. *arXiv:2010.02174*, 2020.

[11] Maria Schuld. Quantum machine learning models are kernel methods. *arXiv:2101.11020*, 2021.

[12] Hsin-Yuan Huang, Michael Broughton, Masoud Mohseni, Ryan Babbush, Sergio Boixo, Hartmut Neven, and Jarrod R. McClean. Power of data in quantum machine learning. *Nature Communications*, 12(1):2631, 2021.

[13] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103(15):150502, 2009.

[14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Phys. Rev. Lett.*, 113(13), 2014.

[15] Nathan Wiebe, Daniel Braun, and Seth Lloyd. Quantum algorithm for data fitting. *Phys. Rev. Lett.*, 109:050505, 2012.

[16] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum algorithms for supervised and unsupervised machine learning. *arXiv:1307.0411*, 2013.

[17] Iordanis Kerenidis and Anupam Prakash. Quantum gradient descent for linear systems and least squares. *Phys. Rev. A*, 101:022316, 2020.

[18] Carlo Ciliberto, Andrea Rocchetto, Alessandro Rudi, and Leonard Wossnig. Statistical limits of supervised quantum learning. *Physical Review A*, 102(4), 2020.

[19] Gian Giacomo Guerreschi and Mikhail Smelyanskiy. Practical optimization for hybrid quantum-classical algorithms. *arXiv:1701.01450*, 2017.

[20] Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99(3):032331, 2019.

[21] James Stokes, Josh Izaac, Nathan Killoran, and Giuseppe Carleo. Quantum natural gradient. *Quantum*, 4:269, 2020.

[22] Ryan Sweke, Frederik Wilde, Johannes Jakob Meyer, Maria Schuld, Paul K Fährmann, Barthélémy Meynard-Piganeau, and Jens Eisert. Stochastic gradient descent for hybrid quantum-classical optimization. *Quantum*, 4:314, 2020.

[23] Jonas M. Kübler, Andrew Arrasmith, Lukasz Cincio, and Patrick J. Coles. An Adaptive Optimizer for Measurement-Frugal Variational Algorithms. *Quantum*, 4:263, 2020.

[24] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9:4812, 2018.

[25] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, USA, 2002.

[26] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[27] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A*, 103:032430, 2021.

[28] Hsin-Yuan Huang, Richard Kueng, and John Preskill. Information-theoretic bounds on quantum advantage in machine learning. *Phys. Rev. Lett.*, 126:190505, 2021.

[29] R. C. Williamson, A. J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. *IEEE Transactions on Information Theory*, 47(6):2516–2532, 2001.

[30] Nello Cristianini, Jaz Kandola, Andre Elisseeff, and John Shawe-Taylor. On kernel target alignment. In Dawn E. Holmes and Lakhmi C. Jain, editors, *Innovations in Machine Learning: Theory and Applications*, pages 205–256. Springer Berlin Heidelberg, 2006.

[31] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13(28):795–828, 2012.

[32] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature Communications*, 12(1):2914, 2021.

[33] Arthur Jacot, Berfin Simsek, Francesco Spadaro, Clement Hongler, and Franck Gabriel. Kernel alignment risk estimator: Risk prediction from training data. In *NeurIPS*, 2020.

[34] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *COLT*, 2001.

[35] Thomas Hubregtsen, David Wierichs, Elies Gil-Fuster, Peter-Jan H. S. Derks, Paul K. Faehrmann, and Johannes Jakob Meyer. Training quantum embedding kernels on near-term quantum computers. *arXiv:2105.02276*, 2021.

[36] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[37] Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I. Latorre. Data re-uploading for a universal quantum classifier. *Quantum*, 4:226, 2020.

[38] M. Cerezo, Andrew Arrasmith, Ryan Babbush, et al. Variational quantum algorithms. *arXiv:2012.09265*, 2020.

[39] Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, et al. Noisy intermediate-scale quantum (nisq) algorithms. *arXiv:2101.08448*, 2021.

[40] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, 2001.

[41] A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

[42] Ingo Steinwart, Don R. Hush, and Clint Scovel. Optimal rates for regularized least squares regression. In *COLT*, 2009.

[43] Simon Fischer and Ingo Steinwart. Sobolev norm learning rates for regularized least-squares algorithms. *Journal of Machine Learning Research*, 21(205):1–38, 2020.

[44] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NeurIPS*, 2007.

[45] Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic. Towards a unified analysis of random fourier features. In *ICML*, 2019.

[46] Jean-Francois Ton, Seth Flaxman, Dino Sejdinovic, and Samir Bhatt. Spatial mapping with gaussian processes and nonstationary fourier features. *Spatial Statistics*, 28:59–78, 2018.

[47] Krikamol Muandet, Kenji Fukumizu, Bharath K. Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Found. Trends Mach. Learn.*, 10(1-2):1–141, 2017.

[48] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive computation and machine learning. MIT Press, 2012.

[49] Aram W. Harrow and Ashley Montanaro. Quantum computational supremacy. *Nature*, 549 (7671):203–209, 2017.

[50] M Cerezo, Akira Sone, Tyler Volkoff, Lukasz Cincio, and Patrick J Coles. Cost-function-dependent barren plateaus in shallow quantum neural networks. *arXiv:2001.00550*, 2020.

[51] F. Pedregosa, G. Varoquaux, Gramfort, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[52] Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, and Nathan Killoran. Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv:1811.04968*, 2018.

[53] Vern I. Paulsen and Mrinal Raghupathi. *An Introduction to the Theory of Reproducing Kernel Hilbert Spaces*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2016.

[54] Z. Puchała and J.A. Miszczak. Symbolic integration with respect to the haar measure on the unitary groups. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, 65(No 1): 21–27, 2017.

[55] Christoph Dankert, Richard Cleve, Joseph Emerson, and Etera Livine. Exact and approximate unitary 2-designs and their application to fidelity estimation. *Phys. Rev. A*, 80:012304, 2009.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See in particular end of Section 4 and Section 6
   (c) Did you discuss any potential negative societal impacts of your work? [No] This work conduct fundamental research towards understanding Quantum Machine Learning. We do not see any direct negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] In the Appendix (Section D and Section E)

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code is submitted with the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] For details please look in Section 5 and at the implementation.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We report errorbars in the left panel of Fig. 2, but omitted them otherwise for better readability.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] The experiments can be run on a standard desktop computer in a few hours, without the need for GPUs.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We cite the main software packages that are used.
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]