Quantum Machine Learning in Feature Hilbert Spaces

Maria Schuld[*] and Nathan Killoran
Xanadu, 372 Richmond Street West, Toronto M5V 2L7, Canada

(Received 29 June 2018; revised manuscript received 7 November 2018; published 1 February 2019)

A basic idea of quantum computing is surprisingly similar to that of kernel methods in machine learning, namely, to efficiently perform computations in an intractably large Hilbert space. In this Letter we explore some theoretical foundations of this link and show how it opens up a new avenue for the design of quantum machine learning algorithms. We interpret the process of encoding inputs in a quantum state as a nonlinear feature map that maps data to quantum Hilbert space. A quantum computer can now analyze the input data in this feature space. Based on this link, we discuss two approaches for building a quantum model for classification. In the first approach, the quantum device estimates inner products of quantum states to compute a classically intractable kernel. The kernel can be fed into any classical kernel method such as a support vector machine. In the second approach, we use a variational quantum circuit as a linear model that classifies data explicitly in Hilbert space. We illustrate these ideas with a feature map based on squeezing in a continuous-variable system, and visualize the working principle with two-dimensional minibenchmark datasets.

DOI: 10.1103/PhysRevLett.122.040504

Introduction.—A goal of most quantum algorithms is to perform efficient computations in a Hilbert space that grows rapidly with the size of a quantum system. "Efficient" means that the number of operations applied to the system grows at most polynomially with the system size. This is pushed to the extreme for continuous-variable systems, where a single operation—for example, squeezing applied to a mode of the electromagnetic field—formally manipulates a quantum state in an infinite-dimensional Hilbert space.

In machine learning, so-called *kernel methods* are a well-established field with a surprisingly similar logic. The idea of kernel methods is to formally embed data into a higher-dimensional (and sometimes infinite-dimensional) *feature space* in which it becomes easier to analyze. A popular example is a support vector machine that draws a decision boundary between two classes of data points by mapping the data into a feature space where it becomes linearly separable. The trick is that the algorithm never explicitly performs computations with vectors in feature space, but uses a so-called *kernel function* that is defined on the domain of the original input data. Just like quantum computing, kernel methods therefore perform implicit computations in a possibly intractably large Hilbert space through the efficient manipulation of data vectors.

Besides this apparent link, kernel methods have hardly been studied in the quantum machine learning literature, a field that (in the definition we employ here) investigates the use of quantum computing as a resource for machine learning. Across the approaches in this young field, which vary from sampling [1–5] to quantum optimization [6,7], linear algebra solvers [8–10], and using quantum circuits as trainable models for inference [11,12], a lot of attention has been paid to recent trends in machine learning such as deep learning and neural networks. Kernel methods, which were most successful in the 1990s, are only mentioned in very few references [9,13,14].

The aim of this Letter is to leverage the relationship between feature maps, kernel methods, and quantum computing. We interpret the process of encoding classical information into a quantum state as a feature map which maps data inputs into the Hilbert space of the quantum system. This leads to two strategies of designing quantum machine learning algorithms. First, we propose to use a simple special-purpose quantum device to estimate inner products of these quantum states, and feed the estimates as a "quantum kernel" into classical kernel models. Second, data can be directly analyzed in the "feature Hilbert space" of quantum states, where simple classifiers such as linear models gain enormous power. Both strategies give rise to classically intractable hybrid quantum machine learning algorithms with near-term quantum technology.

In the following, we will first present the general idea of quantum feature maps and then highlight the resulting quantum machine learning algorithms based on squeezing in a continuous-variable quantum system.

Concepts from kernel theory.—In machine learning we are typically given a dataset of inputs $\mathcal{D} = \{x^1, \ldots, x^M\}$ from a certain input set $\mathcal{X}$, and have to recognize patterns to evaluate or produce previously unseen data. Kernel methods use a distance measure $\kappa(x, x')$—called a kernel—between any two inputs $x, x' \in \mathcal{X}$ in order to construct models that capture the properties of a data distribution.

Definition 1.—Let $\mathcal{X}$ be a nonempty set, called the input set. A function $\kappa: \mathcal{X} \times \mathcal{X} \to \mathbb{C}$ is called a kernel if the Gram matrix K with entries $K_{m,m'} = \kappa(x^m, x^{m'})$ is positive semidefinite, in other words, if for any finite subset $\{x^1, \ldots, x^M\} \subseteq \mathcal{X}$ with $M \geq 2$ and $c_1, \ldots, c_M \in \mathbb{C}$, $\sum_{m,m'=1}^{M} c_m c_{m'}^* \kappa(x^m, x^{m'}) \geq 0$.

Another way to define kernels is that they are inner products in a feature space of the data [15]. A data point is mapped to feature space by a *feature map*.

Definition 2.—Let $\mathcal{F}$ be a Hilbert space, called the feature space, $\mathcal{X}$ an input set, and x a sample from the input set. A feature map is a map $\phi: \mathcal{X} \to \mathcal{F}$ from inputs to vectors in the Hilbert space. The vectors $\phi(x) \in \mathcal{F}$ are called feature vectors.

The feature space is usually of much higher dimension than the original space. If the feature map is a nonlinear function, it changes the relative position between data points, and a dataset can become a lot easier to classify in feature space.

By definition of the inner product, every feature map gives rise to a kernel.

Theorem 1.—Let ϕ: $\mathcal{X} \to \mathcal{F}$ be a feature map. The inner product of two inputs mapped to feature space defines a kernel via $\kappa(x, x') \coloneqq \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$, where $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ is the inner product defined on $\mathcal{F}$.

The connection between feature maps and kernels means that every feature map corresponds to a distance measure in input space by means of the inner product of feature vectors.

While a given kernel can be related to many different feature spaces, kernel theory also defines a unique Hilbert space associated with each kernel, the *reproducing kernel Hilbert space* or RKHS [16,17] (for a definition see Supplemental Material, Sec. I [18]). Since a feature map gives rise to a kernel and a kernel gives rise to a reproducing kernel Hilbert space, we can construct a unique reproducing kernel Hilbert space for any given feature map.

Theorem 2.—Let ϕ: $\mathcal{X} \to \mathcal{F}$ be a feature map over an input set $\mathcal{X}$, giving rise to a complex kernel $\kappa(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$. The corresponding reproducing kernel Hilbert space has the form

$$\begin{aligned} \mathcal{R}_\kappa = \{ f: \mathcal{X} \to \mathbb{C} | \\ f(x) = \langle w, \phi(x) \rangle_{\mathcal{F}}, \ \forall \ x \in \mathcal{X}, w \in \mathcal{F} \}. \end{aligned} \tag{1}$$

The functions $\langle w, \cdot \rangle$ in the RKHS associated with feature map ϕ can be interpreted as linear models, for which $w \in \mathcal{F}$ defines a hyperplane in feature space. If $\phi(x)$ is orthogonal to w, then x lies on the decision boundary, whereas a positive [negative] inner product states that $\phi(x)$ lies on the left [right] side of the hyperplane.

In machine learning these rather formal concepts gain relevance because of the *representer theorem* [28] (see Supplemental Material, Sec. I [18]). The representer theorem states that for a common family of machine learning optimization problems over functions in an RKHS (such as our linear models), the solution can be represented as an expansion of kernel functions,

$$f^*(x) = \sum_{m=1}^{M} \alpha_m \kappa(x, x^m). \tag{2}$$

Consequently, instead of explicitly optimizing over an infinite-dimensional RKHS we can directly start with the ansatz of Eq. (2) and solve the convex optimization problem of finding the parameters α_m. In short, linear models in the RKHS are often equivalent to kernelized models in the input space.

Quantum feature maps.—We will now build the bridge to quantum computing. Assume we want to encode some input x from an input set $\mathcal{X}$ into a quantum state that is described by a vector $|\phi(x)\rangle$ and which lives in Hilbert space $\mathcal{F}$. This procedure of "input encoding" fulfills the definition of a feature map ϕ: $\mathcal{X} \to \mathcal{F}$, which we call a *quantum feature map* here. According to Theorem 1 we can derive a kernel κ from this feature map. By virtue of Theorem 2, the kernel is the reproducing kernel of an RKHS $\mathcal{R}_\kappa$ as defined in Eq. (1). The functions in $\mathcal{R}_\kappa$ are the inner products of the "feature-mapped" input data and a vector $|w\rangle \in \mathcal{F}$, which defines a linear model

$$f(x; w) = \langle w | \phi(x) \rangle. \tag{3}$$

Note that we use Dirac brackets $\langle \cdot | \cdot \rangle$ instead of the inner product $\langle \cdot, \cdot \rangle$ to signify that we are calculating inner products in a quantum Hilbert space. Finally, the representer theorem guarantees that the minimizer $\min_w C(w, \mathcal{D})$ of common empirical risks (such as least squares) can be expressed by Eq. (2). The simple idea of interpreting $x \to |\phi(x)\rangle$ as a feature map therefore allows us to make use of the rich theory of kernel methods and gives rise to machine learning models that can be expressed by inner products of quantum states. Note that if the state $|\phi(x)\rangle$ has complex amplitudes, we can always construct a real kernel by taking the absolute square of the inner product. A more general discussion about how to link quantum and kernel theory can be found in the Supplemental Material, Sec. II [18].

From the perspective of quantum computing, a quantum feature map $x \to |\phi(x)\rangle$ corresponds to a state preparation circuit $U_\phi(x)$ that acts on a ground or vacuum state $|0 \cdots 0\rangle$ of a Hilbert space $\mathcal{F}$ as $U_\phi(x)|0 \cdots 0\rangle = |\phi(x)\rangle$. We will call $U_\phi(x)$ the *feature-embedding circuit*. Examples of common information encoding techniques and their associated quantum feature maps can be found in the Supplemental Material, Sec. III [18].

Building a quantum classifier.—Figure 1 shows two different strategies of designing a quantum machine learning algorithm based on the preceding combination of quantum computing and kernel theory. On the one hand, we can use the quantum computer to estimate the inner

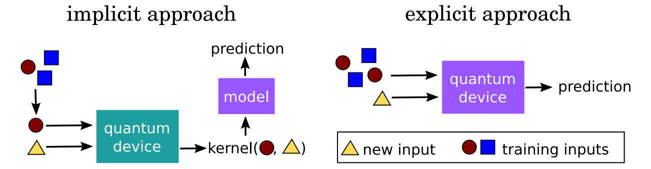

FIG. 1. Illustration of the two approaches to use quantum feature maps for supervised learning. The implicit approach uses the quantum device to evaluate the kernel function, while in the explicit approach, the model is solely computed by the quantum device.

products $\kappa(x, x') = \langle\phi(x)|\phi(x')\rangle$ from a kernel-dependent model as in Eq. (2), which we call the *implicit approach*. This strategy requires a quantum computer that can do two things: implement $U_\phi(x)$ for any $x \in \mathcal{X}$ and estimate inner products between quantum states (for example using a SWAP test routine). The computation of the model from those kernel estimates, as well as the training algorithm is left to a classical device. This is an excellent strategy in the context of intermediate-term quantum technologies [29], where we are interested in using a quantum computer only for small routines of limited gate count, and compute as much as possible on the classical hardware. Note that in the long term, quantum computers could also be used to learn the parameters α_m by computing the inverse of the kernel Gram matrix, which has been investigated in Refs. [9,30].

On the other hand one can bypass the representer theorem and explicitly perform the classification in the feature Hilbert space of the quantum system. We call this the *explicit approach*. For example, this can mean to find a state $|w\rangle$ that defines a model as in Eq. (3). To do so, we can prepare $|w\rangle$ by a variational circuit $W = W(\theta)$ that depends on trainable parameters θ. Quantum-classical hybrid training [31,32] of θ can learn the optimal model $|w(\theta)\rangle = W(\theta)|0\rangle$. The ansatz we choose for the variational circuit defines the space of possible models and can act as regularization (see also [33]). Below, we will follow a slightly more general strategy and compute a state $W(\theta)U_\phi|0\cdots 0\rangle$, from which measurements determine the output of the model. Depending on the measurement, this is not necessarily a linear model in feature Hilbert space.

Squeezing as a feature map.—In the remainder of this work we explore these two approaches with a toy example. We use the process of squeezing in continuous-variable quantum systems as a feature map [34] (for an investigation with coherent states instead, see also [13]). This leads to a classically tractable kernel—thus facilitating the analysis—that has to our knowledge not been studied in classical machine learning.

Squeezing decreases the uncertainty for one observable from a pair of commuting observables with a continuous spectrum. The Hilbert space $\mathcal{F}$ of such a system can be expressed as an infinite-dimensional *Fock space* with basis $\{|0\rangle, |1\rangle, \ldots\}$. The resulting quantum machine learning algorithm is particularly suitable for photonic quantum computers. However, the reader will not require special knowledge in the field of CV quantum computing to follow the analysis.

A *squeezed vacuum state* of the electromagnetic field is defined as

$$|z\rangle = \frac{1}{\sqrt{\cosh(r)}} \sum_{n=0}^{\infty} \frac{\sqrt{(2n)!}}{2^n n!} [-e^{i\varphi} \tanh(r)]^n |2n\rangle,$$

where $\{|n\rangle\}$ denotes the Fock basis and $z = re^{i\varphi}$ is the complex *squeezing factor* with absolute value r and phase φ. It will be useful to introduce the notation $|z\rangle = |(r, \varphi)\rangle$. We can interpret $x \to |\phi(x)\rangle = |(c, x)\rangle$ as a feature map from a one-dimensional real input space $x \in \mathbb{R}$ to Fock space. Here, c is a constant hyperparameter that determines the strength of the squeezing, and x is associated with the phase. Moreover, when given multidimensional inputs in a dataset of vectors $x = (x_1, \ldots, x_N)^T \in \mathbb{R}^N$, we can define the joint state of N squeezed vacuum modes,

$$\phi : x \to |(c, x)\rangle, \tag{4}$$

with $|(c, x)\rangle = |(c, x_1)\rangle \otimes \cdots \otimes |(c, x_N)\rangle \in \mathcal{F}$, as a feature map, where $\mathcal{F}$ is now a multimode Fock space. We call this feature map the *squeezing feature map with phase encoding*.

The kernel

$$\kappa(x, x'; c) = \prod_{i=1}^{N} \langle (c, x_i)|c, x_i'\rangle \tag{5}$$

with

$$\langle (c, x_i)|c, x_i'\rangle = \sqrt{\frac{\operatorname{sech} c \operatorname{sech} c}{1 - e^{i(x_i' - x_i)} \tanh c \tanh c}}, \tag{6}$$

derived from this feature map [35] is easy to compute on a classical computer. It is plotted in Fig. 2, where we see that the hyperparameter c determines the variance of the kernel function. Furthermore, the squeezing feature map has the property of mapping data into a space where they become linearly separable (for a proof see Supplemental Material, Sec. IV [18]).

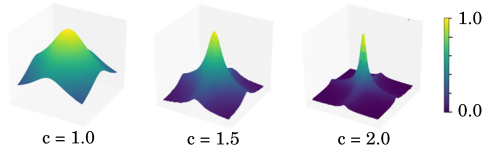

FIG. 2. Shape of the squeezing kernel function $\kappa_{\text{sq}}(x, x')$ from Eq. (5) for different squeezing strength hyperparameters c. The input x is fixed at (0,0) and x' is varied. The plots show the interval $[-1, 1]$ on both horizontal axes.

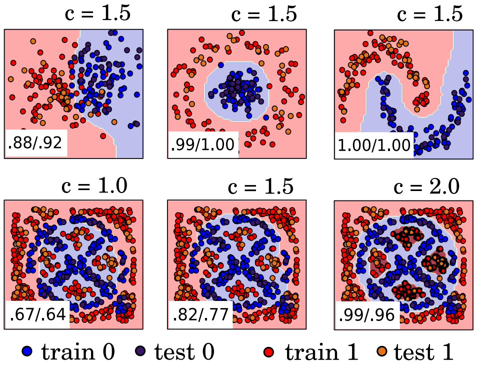

FIG. 3. Decision boundary of a support vector machine with the custom kernel from Eq. (5). The shaded areas show the decision regions for class 0 (blue) and class 1 (red), and each plot shows the rate of correct classifications on the training set or test set. The first row plots three standard two-dimensional datasets: "circles," "moons," and "blobs," each with 150 test and 50 training samples. The second row illustrates that increasing the squeezing hyperparameter c changes the classification performance. Here, we use a dataset of 500 training and 100 test samples. Training was performed with Python's *scikit-learn* SVC classifier using a custom kernel which implements the overlap of Eq. (6).

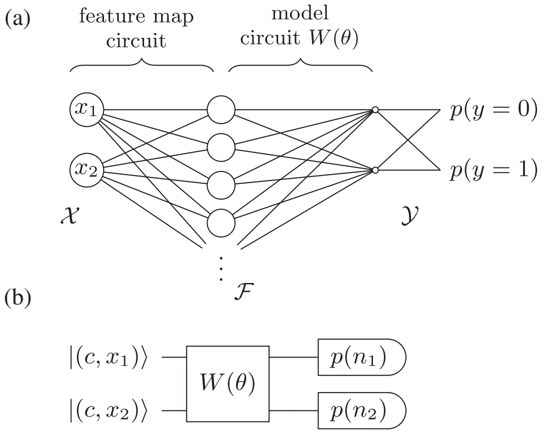

FIG. 4. (a) Representation of the Fock-space classifier in the graphical language of quantum neural networks. A vector $(x_1,x_2)^T$ from the input space $\mathcal{X}$ gets mapped into the feature Hilbert space $\mathcal{F}$ which is the infinite-dimensional two-mode Fock space of the quantum system. The variational circuit including photon detection measurement reduces the "infinite hidden layer" to two outputs. (b) The variational circuit $W(\theta)$ consists of repetitions of a gate layer described in the Supplemental Material, Sec. V [18].

Implicit approach.—Remember that in the implicit approach, we evaluate the kernel in Eq. (5) with a quantum computer and feed it into a classical kernel method. Instead of using a real quantum device, we exploit the fact that the kernel can be efficiently computed classically, and use it as a custom kernel in a support vector machine. Figure 3 shows that such a model easily learns the decision boundary of two-dimensional minibenchmark datasets.

While the results of these simulations show that the principle works, a goal is to find more sophisticated kernels. Although quantum computers could offer constant speed advantages, they become indispensable if the feature map circuit U_ϕ together with the kernel $\kappa(x,x') = \langle 0\cdots 0|U_\phi^\dagger(x)U_\phi(x')|0\cdots 0\rangle$ is classically intractable. In continuous-variable quantum computing, so called *Gaussian gates* (including squeezing) can be efficiently simulated by a classical computer [36]. In order to do something more interesting, one needs non-Gaussian elements to the circuit such as a cubic phase gate [37,38] or photon number measurements [39]. An example [40] for a feature map that is believed to be classically intractable is an IQP [41] or CV-IQP [42] circuit.

It is an interesting open question which classically intractable input embeddings lead to powerful kernels for classical models such as support vector machines.

Explicit approach.—In the explicit approach defined above, we use a variational circuit $W(\theta)$ on top of the feature map circuit to build a "Fock-space" classifier. For our two-dimensional squeezing example, this can be done as follows. We start with two vacuum modes $|0\rangle\otimes|0\rangle$. To classify a data input $x\in\mathbb{R}^2$, first map the input to a quantum state $|c,x\rangle = |c,x_1\rangle\otimes|c,x_2\rangle$ by performing a squeezing operation on each of the modes. Second, apply the variational circuit $W(\theta)$ to $|c,x\rangle$. Third, repeated photon number measurements give us estimates of the probability $p(n_1,n_2)$ of measuring Fock state $|n_1,n_2\rangle$. Interpret the normalized probability $p(2,0)/[p(2,0)+p(0,2)]$ as the probability that the model predicts class $y=0$. The final label is the class with the higher probability. We can interpret this circuit in the graphical representation of neural networks as shown at the top in Fig. 4.

Since the data in $\mathcal{F}$ is linearly separable, there is a circuit W for which we obtain 100% accuracy on the training set. However, we realistically only have a limited circuit depth

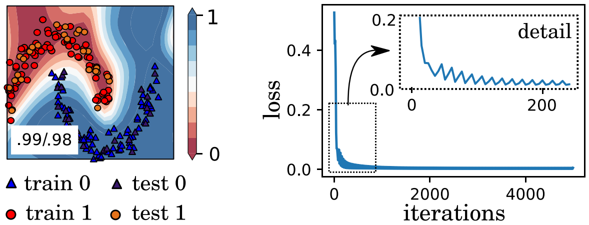

FIG. 5. Fock-space classifier presented in Fig. 4 and the text for the moons dataset. The shaded areas show the probability $p(y=1)$ of predicting class 1. The model has been trained for 5000 steps with stochastic gradient descent of batch size 5, an adaptive learning rate and a square-loss cost function with a gentle l_2 regularization applied to all weights. The loss drops predominantly in the first 200 steps (left). The simulations were performed with the Quantum Machine Learning Toolbox (QMLT) application for the STRAWBERRYFIELDS software platform [43].

and number of parameters. We therefore need to find a good ansatz for the variational circuit $W(\theta)$, so that the model has enough flexibility to generalize. The ansatz we choose consists of layers of universal continuous-variable quantum gates and is further explained in the Supplemental Material, Sec. V [18].

To show that the Fock-space classifier works in principle, we plot the decision boundary for the moons data in Fig. 5, using four repetitions of the layer in the variational circuit, and 32 parameters in total. The training loss shows that after about 200 iterations of a stochastic gradient descent algorithm, the loss converges to almost zero.

Conclusion.—In this Letter we use quantum circuits to map data to quantum Hilbert spaces, which serve as feature spaces for the data. We propose two strategies inspired by kernel theory to find patterns in the data. First, by estimating intractable quantum kernels on the quantum device and feeding them into a classical kernel method. Second, we can use quantum models based on variational circuits to learn models that process the feature vectors. These simple models gain power from "outsourcing" the nonlinearity into the procedure of encoding inputs into a quantum state, or the quantum feature map. We expect that this combination of quantum computing and kernel theory will help to design quantum machine learning algorithms for near-term quantum devices which offer potential quantum speedups.

[*] maria@xanadu.ai

[1] G. Verdon, M. Broughton, and J. Biamonte, arXiv:1712.05304.

[2] M. H. Amin, Phys. Rev. A **92**, 052323 (2015).

[3] M. Benedetti, J. Realpe-Gómez, R. Biswas, and A. Perdomo-Ortiz, Phys. Rev. X **7**, 041052 (2017).

[4] G. H. Low, T. J. Yoder, and I. L. Chuang, Phys. Rev. A **89**, 062315 (2014).

[5] P. Wittek and C. Gogolin, Sci. Rep. **7**, 45672 (2017).

[6] V. Denchev, N. Ding, H. Neven, and S. Vishwanathan, in *Proceedings of the 29th International Conference on Machine Learning (ICML-12)* (Omnipress, 2012), pp. 863–870.

[7] B. OGorman, R. Babbush, A. Perdomo-Ortiz, A. Aspuru-Guzik, and V. Smelyanskiy, Eur. Phys. J. Spec. Top. **224**, 163 (2015).

[8] N. Wiebe, D. Braun, and S. Lloyd, Phys. Rev. Lett. **109**, 050505 (2012).

[9] P. Rebentrost, M. Mohseni, and S. Lloyd, Phys. Rev. Lett. **113**, 130503 (2014).

[10] M. Schuld, I. Sinayskiy, and F. Petruccione, Phys. Rev. A **94**, 022342 (2016).

[11] K. H. Wan, O. Dahlsten, H. Kristjánsson, R. Gardner, and M. Kim, npj Quantum Inf. **3**, 36 (2017).

[12] E. Farhi and H. Neven, arXiv:1802.06002.

[13] R. Chatterjee and T. Yu, Quantum Inf. Commun. **17**, 1292 (2017).

[14] M. Schuld, M. Fingerhuth, and F. Petruccione, Europhys. Lett. **119**, 60002 (2017).

[15] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (MIT Press, Cambridge, MA, 2002).

[16] T. Hofmann, B. Schölkopf, and A. J. Smola, Ann. Stat. **36**, 1171 (2008).

[17] N. Aronszajn, Trans. Am. Math. Soc. **68**, 337 (1950).

[18] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.122.040504 for more details, such as an investigation of the link between quantum mechanics and kernel theory, examples of quantum feature maps, proof that the squeezing feature map makes data linearly separable, as well as more details on the architecture of the explicit feature map circuit, which includes Refs. [19–27].

[19] B. J Mercer, Phil. Trans. R. Soc. A **209**, 415 (1909).

[20] A. Berlinet and C. Thomas-Agnan, *Reproducing Kernel Hilbert Spaces in Probability and Statistics* (Springer Science & Business Media, New York, 2011).

[21] T. Griffiths and A. Yuille, *The Probabilistic Mind: Prospects for Bayesian Cognitive Science* (Oxford University Press, Oxford, 2008), p. 33.

[22] R. D. la Madrid, Eur. J. Phys. **26**, 287 (2005).

[23] J. R. Klauder and B.-S. Skagerstam, *Coherent States: Applications in Physics and Mathematical Physics* (World Scientific, Singapore, 1985).

[24] S. Wang, J. Math. Res. **7**, 175 (2015).

[25] A. B. Novikoff, On convergence proofs for perceptrons, Stanford Research Institute, Technical Report No. AD0298258, 1963.

[26] L. Hogben, *Handbook of Linear Algebra* (Taylor & Francis, Boca Raton, 2006).

[27] F. Flamini, N. Spagnolo, N. Viggianiello, A. Crespi, R. Osellame, and F. Sciarrino, Sci. Rep. **7**, 15133 (2017).

[28] B. Schölkopf, R. Herbrich, and A. Smola, in *Computational Learning Theory* (Springer, New York, 2001), pp. 416–426.

[29] J. Preskill, Quantum Phys. **2**, 79 (2018).

[30] M. Schuld, I. Sinayskiy, and F. Petruccione, Phys. Rev. A **94**, 022342 (2016).

[31] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, New J. Phys. **18**, 023023 (2016).

[32] G. G. Guerreschi and M. Smelyanskiy, arXiv:1701.01450.

[33] E. Stoudenmire and D. J. Schwab, in *30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain* (2016), pp. 4799–4807.

[34] C. Weedbrook, S. Pirandola, R. García-Patrón, N. J. Cerf, T. C. Ralph, J. H. Shapiro, and S. Lloyd, Rev. Mod. Phys. **84**, 621 (2012).

[35] S. M. Barnett and P. M. Radmore, *Methods in Theoretical Quantum Optics* (Oxford University Press, New York, 2002), Vol. 15.

[36] S. D. Bartlett, B. C. Sanders, S. L. Braunstein, and K. Nemoto, in *Quantum Information with Continuous Variables* (Springer, New York, 2002), pp. 47–55.

[37] D. Gottesman, A. Kitaev, and J. Preskill, Phys. Rev. A **64**, 012310 (2001).

[38] S. Lloyd and S. L. Braunstein, in *Quantum Information with Continuous Variables* (Springer, New York, 1999), pp. 9–17.

[39] S. D. Bartlett and B. C. Sanders, Phys. Rev. A **65**, 042304 (2002).

[40] V. Havlicek, A. D. Córcoles, K. Temme, A. W. Harrow, J. M. Chow, and J. M. Gambetta, arXiv:1804.11326.

[41] M. J. Bremner, A. Montanaro, and D. J. Shepherd, Phys. Rev. Lett. **117**, 080501 (2016).

[42] T. Douce, D. Markham, E. Kashefi, E. Diamanti, T. Coudreau, P. Milman, P. van Loock, and G. Ferrini, Phys. Rev. Lett. **118**, 070503 (2017).

[43] N. Killoran, J. Izaac, N. Quesada, V. Bergholm, M. Amy, and C. Weedbrook, arXiv:1804.03159.