Quantum kernels with squeezed-state encoding for machine learning

Long Hin Li,† Dan-Bo Zhang,‡ and Z. D. Wang†‡

1Guangdong-Hong Kong Joint Laboratory of Quantum Matter, Department of Physics, and HKU-UCAS Joint Institute for Theoretical and Computational Physics at Hong Kong, The University of Hong Kong, Pokfulam Road, Hong Kong, China
2Guangdong-Hong Kong Joint Laboratory of Quantum Matter, Frontier Research Institute for Physics, South China Normal University, Guangzhou 510006, China
3Guangdong Provincial Key Laboratory of Quantum Engineering and Quantum Materials, School of Physics and Telecommunication Engineering, South China Normal University, Guangzhou 510006, China

(Dated: January 21, 2022)

Kernel methods are powerful for machine learning, as they can represent data in feature spaces that similarities between samples may be faithfully captured. Recently, it is realized that machine learning enhanced by quantum computing is closely related to kernel methods, where the exponentially large Hilbert space turns to be a feature space more expressive than classical ones. In this paper, we generalize quantum kernel methods by encoding data into continuous-variable quantum states, which can benefit from the infinite-dimensional Hilbert space of continuous variables. Specifically, we propose squeezed-state encoding, in which data is encoded as either in the amplitude or the phase. The kernels can be calculated on a quantum computer and then are combined with classical machine learning, e.g., support vector machine, for training and predicting tasks. Their comparisons with other classical kernels are also addressed. Lastly, we discuss physical implementations of squeezed-state encoding for machine learning in quantum platforms such as trapped ions.

I. Introduction

Machine learning can be enhanced by quantum computing utilizing the incredible power of quantum computers for information processing [1–10]. The first breakthrough comes from an exponential quantum speedup for solving linear equations or matrix inversion [11, 12], which lies at the heart of many machine learning methods. While quantum computers are good at linear algebra since they in nature perform unitary transformations that are linear in nature, it is not enough for machine learning of complicated data, where highly non-linear transformations are often anticipated [13]. Another important ingredient for quantum-enhanced machine learning, only recognized very recently, is to introduce non-linearity when encoding classical data into quantum states [7, 8, 14–18]. Such a procedure is called quantum feature maps [7, 8], which enjoy the exponential large Hilbert space as a feature space for expressing data.

The quantum feature maps are closely related to classical kernel methods in the same spirit as they aim to use feature spaces, such that similarities between samples can be faithfully captured [19, 20]. One striking difference is that feature maps on a quantum computer are explicitly constructed [18]. This gives the flexibility of designing or even training powerful quantum kernels that may provide quantum advantages for machine learning. Varies encoding schemes for quantum kernels have been explored. Typically, data vectors are encoded as parameters of quantum gates, such as rotation angles of single-qubit gates. By loading the same data repeatedly into different quantum gates, the quantum data state will be a non-linear function of the data, which consists of both different high-frequency and low-frequency components [17]. Those types of quantum kernels are rather different from classical ones, and while useful applications can be expected, they would have limitations (e.g., similarity between two samples may not be nicely approximated as a periodic function of data vector). On the other hand, quantum computing and quantum machine learning with continuous variables can inherit many properties of classical machine learning [8, 21–24], e.g., the widely-applied Gaussian kernel can be realized on a quantum computer by coherent-state encoding. Moreover, squeezed-state encoding has been proposed in Ref. [8]. This suggests that quantum feature maps by encoding data as continuous-variable states can be promising, by exploiting the infinite dimensionality of continuous-variable for encoding high-density information.

In this paper, we focus on one important class of continuous-variable states, the squeezed state, as a quantum kernel for machine learning. We first introduce the squeezing vacuum state, and two different encoding schemes, the amplitude encoding and the phase encoding. Those different data encoding methods can give different kernels, which have their unique properties for modeling data. The kernel can be calculated as an inner product between two quantum states that encode the two data, which further serves as input for machine learning models, such as support vector machines. We will then test those kernels and also compare them with some classical kernels on different datasets classification tasks. Finally,
we propose a physical platform for implementation and also discuss further possibilities to expand the kernel libraries.

II. Quantum kernels with squeezed-state encoding

Let us first introduce some notations. For a sample represented by a data vector \( x \), its quantum feature map is realized by encoding the data into a quantum state \( |\psi_x\rangle \). The quantum kernel of two samples \( x \) and \( x' \) is an inner product between their corresponding quantum states, namely \( K(x, x') = \langle \psi_x | \psi_{x'} \rangle \).

A squeezed vacuum state is a minimum uncertainty state satisfying \( \Delta \hat{x} \Delta \hat{\rho} = \frac{1}{4} \), where \( \hat{x} \) and \( \hat{\rho} \) and conjugate quadrature operators with either quadrature variance squeezed \([25]\). It can be obtained from the vacuum state \( |0\rangle \),

\[
|z\rangle_s = \hat{S}(z) |0\rangle
\]  

with squeezing operator defined as \( \hat{S}(z) = e^{z \hat{a} - \dagger z \hat{a}} \). The position uncertainty is "squeezed" if the amplitude \( r \) of the squeezing parameter \( z = re^{i\theta} \) is positive. The momentum uncertainty will be enlarged by the same proportion to keep their product constant. The squeezed state can be expanded in terms of Fock state basis by using the property \( \hat{S}(z) \hat{a} \hat{S}^\dagger(z) = \hat{a} \cosh r + \hat{a}^\dagger \sinh r \),

\[
|z\rangle_s = \frac{1}{\cosh r} \sum_{n=0}^{\infty} (-1)^n \sqrt{(2n)!} e^{in\theta} \tanh^n r |2n\rangle
\]  

Here we introduce the notation separating the amplitude and phase of the squeezed vacuum state \( |z\rangle = |r, \theta\rangle \). The inner product between squeezed vacuum states is

\[
\langle r, \theta | r', \theta' \rangle_s = \sqrt{\frac{\text{sech} r \text{sech} r'}{1 - e^{i(\theta' - \theta)}} \text{tanh} r \text{tanh} r'}
\]  

The inner product will serve as a measure of similarity between data, equivalent to the kernel in machine learning \([20]\). This provides a bridge from quantum computing to machine learning. In many instances, we use basic encoding \( |\psi_x\rangle = |x\rangle \) with \( |x\rangle \) a tensor product of binary qubits or amplitude encoding \( |\psi_x\rangle = \sum_{n=0}^{N} x_n |n\rangle \) to encode data in quantum algorithms. Apart from the discrete variable qubit, encoding with coherent states are also possible \([24]\). Different feature maps are able to capture different data patterns. And with the greater amount of possible feature maps, more data patterns can be approximated. For example, the amplitude encoding gives the usual dot product and basic encoding on coherent state gives Gaussian kernel.

A. Phase encoding

Given an \( N \)-dimensional dataset \( \{x_m\} \), we can encode data into the phase parameters of \( N \) squeezed vacuum modes with the amplitude as a hyperparameter \([8]\). The encoding was \( \phi : x \rightarrow |c, x\rangle_s \), with \( |c, x\rangle_s \) defined in Eq. \( (2) \) and is referred as squeezing feature map with phase encoding. We call this kernel the squeezing phase kernel. The kernel plot for different data entries is a periodic function, as shown in Fig. 1. The similarity between data valued at multiples of \( 2\pi \) separations is high. This shares similar properties to the exponential sine squared kernel in machine learning

\[
K_{\text{es}}(x, x') = \exp \left( -\frac{2}{l^2} \sin^2 (|x - x'|/p) \right),
\]  

where \( l, p \) are hyperparameters of the length scale and the period, respectively. These kinds of kernels are suitable for data that has features of periodicity. For example, the weather forecasting data and the stock market data. From Fig. 1 the squeezing phase kernel has sharper peaks compared to the exponential sine squared kernel, and may be more suitable for modeling functions with sudden changes.

\[
\text{FIG. 1. Comparison between exponential sine squared kernel for different } l \text{ from Eq. } (4) \text{ (dash line) and squeezing phase kernel for different } c \text{ from Eq. } (5) \text{ (solid line).}
\]  

In addition, for coherent-state encoding \([24]\), data can also be encoded into the argument of coherent state that the kernel becomes (the real part of the inner product),

\[
\Re\langle c, x | c, x' \rangle_c = e^{-c^2 - c^2 e^{i(x' - x)}}
\]  

\[
\Re\langle c, x | c, x' \rangle_c = e^{-c^2 - c^2 \cos (x' - x)} \cos (c^2 \sin (x' - x))
\]

The hyperparameter \( c \) has the similar function as the length scale \( l \) in the exponential sine squared kernel. We call this kernel the coherent phase kernel. Although there are no parameters for tuning the desired period, this can always be done by rescaling the data with some multiplication factor. Therefore, both phase encoding of squeezing and coherent feature maps are periodic kernels, which can be used to model functions that are periodic, similar to the exponential sine squared kernel.
B. Amplitude encoding

Another way to encode into squeezed states is to encode the data into the amplitude of squeezing parameter \( \phi : x \rightarrow |x, 0\rangle \), where the phase is set as zero. There is no hyperparameter in this case since the exponential factor vanishes for every equal value in the squeezing vacuum inner product (Eq. (3)) and we set it as zero. The resulting kernel is

\[
\langle x, 0|x', 0\rangle_a = \sqrt{\frac{\text{sech} x \text{sech} x'}{1 - \tanh x \tanh x'}}
\]

(7)

We call this squeezing amplitude kernel. The kernel is symmetric and positive semidefinite, and has a shape similar to the Gaussian kernel, as shown in Fig. 2. The inner product formed by the basis encoding of the coherent state into the amplitude has been shown to have the similar functional form of Gaussian kernel \(^{24}\), also with no extra hyperparameter,

\[
\langle x, 0|x', 0\rangle_c = \exp \left( -\frac{1}{2}|x - x'|^2 \right)
\]

(8)

The squeezing kernel has a larger spread compared to the Gaussian kernel. It can be used to model functions with larger variance, which varies more slowly with respect to the distance from a center point.

![FIG. 2. Comparison between Gaussian kernel (left) and squeezing amplitude kernel (right).](image)

III. Machine learning with quantum kernels

We now apply quantum kernels with squeezed-state encoding for machine learning. The kernels are obtained through a quantum circuit of evaluating the inner product between two quantum states, and then serve as the input for the support vector machine for training and prediction. We further test those quantum kernels on some standard datasets and compare the performances with those of classical kernels.

A. Support vector machine with quantum kernels

There are two main streams of quantum algorithms where the kernel method applies. The first kind is solely based on quantum computers. Famous examples include linear regression \(^{12, 22}\), quantum support vector machine \(^3\), quantum principal component analysis \(^2\). The other class is the classical-quantum hybrid approach that a part of computational task is left to classical devices. In this regime, the optimization of the machine learning algorithm is usually outsourced to classical computers. Quantum computers are only required to generate quantum gates that encode data, and then to calculate the inner product between quantum states. This approach is referred to as quantum kernel estimation \(^8\), which is easier to achieve than the first approach as both the circuit depth and the quantum resources required are less. We can use quantum computers to calculate the proposed quantum kernels above, and the result can be used by classical computers for training traditional machine learning models like support vector machines, principal component analysis, clustering, and Gaussian processes. We take the support vector machine as an example.

Firstly, the inner product can be calculated on a quantum computer in two ways. The first is to use the swap test. Taking the squeezing kernel as an example. We need to prepare the quantum state

\[
|\Psi\rangle = \frac{1}{\sqrt{2}} (|0\rangle|z\rangle_a + |1\rangle|z'\rangle_a)
\]

then take a \( \sigma_x \) measurement on the ancillary qubit, to measure the probability of getting +1. The inner product can be calculated by \( 2p - 1 \). Another approach to calculating the inner product is to apply two consecutive squeezing operators \( S(z) S(z') \), and measure the probability that the qumode is in vacuum state \( |0\rangle \). This avoids the construction of states that entangle qubit and qumode in Eq. (9).

Secondly, with kernels at hand, we continue the procedure of machine learning on classical computers. To optimize an SVM classifier, we have to maximize the Lagrangian dual function \(^{19, 20}\)

\[
L(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} y_i y_j \alpha_i \alpha_j K(x_i, x_j)
\]

subject to the constraint \( \sum_{i=1}^{M} \alpha_i y_i = 0 \) and \( \alpha_i \geq 0 \) for every \( i \). Optimized values of \( \alpha_i \) and \( b \) are derived from Karush-Kuhn-Tucker (KKT) conditions,

\[
y_i y_j K(x_i, x_j) = 1
\]

(11)

\[
y_i \left( \sum_{j \in S} \alpha_j y_j K(x_j, x_i) + b \right) = 1
\]

(12)

\[
b = \frac{1}{N_S} \sum_{i \in S} y_i - \sum_{j \in S} \alpha_j y_j K(x_j, x_i)
\]

(13)

\[
L\left(\vec{a}\right) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{M} y_i y_j \alpha_i \alpha_j K(x_i, x_j)
\]

subject to the constraint \( \sum_{i=1}^{M} \alpha_i y_i = 0 \) and \( \alpha_i \geq 0 \) for every \( i \). Optimized values of \( \alpha_i \) and \( b \) are derived from Karush-Kuhn-Tucker (KKT) conditions,

\[
y_i y_j K(x_i, x_j) = 1
\]
where $S$ denotes the set of support vectors. Then, prediction can be calculated by
$$y(x) = \sum_{i \in S} \alpha_i y_i K(x_i, x) + b \quad (14)$$

Therefore, a support vector classifier can be trained after computing the kernel for each pair of data $K(x_j, x_i)$ on the quantum computer, and similar procedures apply to other traditional kernel-based machine learning models.

### B. Results

As an illustration purpose, we test and compare those kernels mentioned in Sec. II for supervised learning of classification on some standard datasets from scikit-learn [26]. Those kernels are used to calculate randomly generated sample datasets of each 200 data, and the SVM model was trained with the kernel. Then, another set of 200 data is used for validation to calculate the performance score. The results are shown in Fig. 3 for comparing squeezing amplitude kernel with the standard Gaussian kernel, and in Fig. 4 for comparing the coherent phase kernel and exponentiated sine squared kernel. Their corresponding performance scores are listed in Table I and Table II respectively. The decision boundary of the squeezing amplitude kernel is comparable to the boundary created by the standard Gaussian kernel. The variance of the squeezing amplitude kernel is slightly lower than the Gaussian kernel but still demonstrated the ability to capture the highly nonlinear behaviour of the dataset. On the other hand, the decision boundary created for coherent phase kernel and exponentiated sine squared kernels are different for each case but they shared the common shape in general. These demonstrated that the two kernels will be useful in different data patterns.

|                | Blobs | Moons | Circles |
|----------------|-------|-------|---------|
| Squeezing amplitude | 0.915 | 0.95  | 0.795   |
| Gaussian         | 0.9   | 1.0   | 0.79    |

TABLE I. The accuracy of validation corresponds to Figure 3.

|                | Moons | Spiral | Sine     |
|----------------|-------|--------|----------|
| Coherent phase | 1.0   | 0.995  | 0.96     |
| Exp sine squared | 1.0  | 1.0    | 0.95     |

TABLE II. The accuracy of validation corresponds to Figure 4.

### C. Physical implementation

The whole machine learning procedure only requires the quantum computer at the stage of evaluating quantum kernels, and we briefly discuss its physical implementation on a quantum system. The inner product has to be calculated on a physical system that allows manipulation of both continuous variables and a discrete-variable qubit as control. Promising candidates of quantum platforms capable of hybrid variable quantum computing include superconducting circuit and trapped ion quantum computer [27] [32]. We take trapped ion as an example.
The motional state of the ion can be used as continuous variables, and various motional states have been realized experimentally \[30-32\]. Remarkably, squeezed vacuum state \(|z\rangle_s\) can be created by irradiating two Raman beams with some specific frequency difference. The squeezing parameter will grow exponentially with driving time for the Raman transition that occurs during the process. Moreover, internal levels of an ion can serve as a qubit and its coupling to the motional modes can implement the quantum circuit to generate the state in Eq. [9] that is central to the evaluation of quantum kernels.

### IV. Conclusions

We have illustrated the idea of using different continuous-variable quantum states to generate new non-linear kernels that can be used to learn different data patterns in machine learning. By encoding the data into either amplitude or argument of the parameter of squeezing, different kernels can be obtained. We have applied those quantum kernels with classical support vector machines for classification tasks on some standard datasets, and their performances are comparable or better to typical classical kernels. Furthermore, we have suggested a quantum platform, the trapped ion quantum computer, which can implement the quantum algorithm involving both discrete-variable qubits and continuous variables. We mention that the squeezing and displacement operator suggested in this article belong to a larger family called Gaussian operations, and general Gaussian-state encoding may be used as quantum kernels for machine learning. Moreover, since Gaussian-state can be characterized by the covariance matrix, a fully exploitation of the infinite dimensionality of continuous variables should refer to non-Gaussian quantum states. The case of non-Gaussian state encoding and its applications as quantum kernels with quantum advantages will be investigated in the future.

### Acknowledgments

This work was supported by the Key-Area Research and Development Program of GuangDong Province (Grant No. 2019B030330001) and the Collaborative Research Fund of Hong Kong (No. C6005-17G).

---

[1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature 549, 195 (2017).
[2] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nat. Phys. 10, 631 (2014).
[3] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. 113, 130503 (2014).
[4] V. Dunjko, J. M. Taylor, and H. J. Briegel, Quantum-enhanced machine learning, Phys. Rev. Lett. 117, 130501 (2016).
[5] S. Lloyd, S. Garnerone, and P. Zanardi, Quantum algorithms for topological and geometric analysis of data, Nature Communications 7, 10138 (2016).
[6] S. Lloyd and C. Weedbrook, Quantum generative adversarial learning, Phys. Rev. Lett. 121, 040502 (2018).
[7] V. Havlíček, A. D. Corcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature 567, 209 (2019).
[8] M. Schuld and N. Killoran, Quantum machine learning in feature hilbert spaces, Phys. Rev. Lett. 122, 040504 (2019).
[9] H.-Y. R. Huang, M. B. Broughton, M. Mohseni, R. Babush, S. Boixo, H. Neven, and J. R. McQueen, Power of data in quantum machine learning, Nature Communications 12, 1 (2021).
[10] Y. Liu, S. Arunachalam, and K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nature Physics 10.1038/s41567-021-01287-z (2021).
[11] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. 103, 150502 (2009).
[12] N. Wiebe, D. Braun, and S. Lloyd, Quantum algorithm for data fitting, Phys. Rev. Lett. 109, 050505 (2012).
[13] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning (MIT Press, 2016).
[14] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, Quantum circuit learning, Phys. Rev. A 98, 032309 (2018).
[15] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, Quantum embeddings for machine learning (2020), arXiv:2001.03622 [quant-ph].
[16] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, Data re-uploading for a universal quantum classifier, Quantum 4, 226 (2020).
[17] M. Schuld, R. Sweke, and J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Phys. Rev. A 103, 032430 (2021).
[18] M. Schuld, Supervised quantum machine learning models are kernel methods (2021), arXiv:2101.11020 [quant-ph].
[19] C. M. Bishop, Pattern recognition and machine learning, Vol. 1 (Springer, 2006).
[20] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, The Elements of Statistical Learning 2nd ed. (Springer, 2009).
[21] H. K. Lau, R. Pooser, G. Siopsis, and C. Weedbrook, Quantum machine learning over infinite dimensions, Phys. Rev. Lett. 118, 080501 (2017).
[22] D.-B. Zhang, Z.-Y. Xue, S.-L. Zhu, and Z. D. Wang, Realizing quantum linear regression with auxiliary qumodes, Phys. Rev. A 99, 012331 (2019).
[23] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, Continuous-variable quantum neural networks, Phys. Rev. Research 1, 033063 (2019).
[24] D.-B. Zhang, S.-L. Zhu, and Z. D. Wang, Protocol for implementing quantum nonparametric learning with trapped ions, Phys. Rev. Lett. 124, 010506 (2020).

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

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12, 2825 (2011).

[27] D. Leibfried, R. Blatt, C. Monroe, and D. Wineland, Quantum dynamics of single trapped ions, Rev. Mod. Phys. 75, 281 (2003).

[28] H. Häffner, C. F. Roos, and R. Blatt, Quantum computing with trapped ions, Physics Reports 469, 155 (2008).

[29] C. Monroe and J. Kim, Scaling the ion trap quantum processor, Science 339, 1164 (2013).

[30] L. Ortiz-Gutiérrez, B. Gabrielly, L. F. Muñoz, K. T. Pereira, J. G. Filgueiras, and A. S. Villar, Continuous variables quantum computation over the vibrational modes of a single trapped ion, Opt. Commun. 397, 166 (2017).

[31] J. Zhang, M. Um, D. Lv, J.-N. Zhang, L.-M. Duan, and K. Kim, Noon states of nine quantized vibrations in two radial modes of a trapped ion, Phys. Rev. Lett. 121, 160502 (2018).

[32] C. Flühmann, T. L. Nguyen, M. Marinelli, V. Negnevitsky, K. Mehta, and J. P. Home, Encoding a qubit in a trapped-ion mechanical oscillator, Nature 566, 513 (2019).