Continuous variable quantum MNIST classifiers

Classical-quantum hybrid quantum neural networks

Sophie Choe[1]

[1] *Electrical and Computer Engineering, Portland State University, Portland, OR*

Abstract

In this paper, classical and continuous variable (CV) quantum neural network hybrid multi-classifiers are presented using the MNIST dataset. The combination of cutoff dimension and probability measurement method in the CV model allows a quantum circuit to produce output vectors of size n^m where $n =$cutoff dimension and $m =$the number of qumodes. They are then translated as one-hot encoded labels, padded with $n^m - 10$ zeros. The total of eight different classifiers are built using $2, 3, \ldots, 8-$ qumodes, based on the binary classifier architecture proposed in "Continuous variable quantum neural networks" [14]. The displacement gate and the Kerr gate in the CV model allow for the bias addition and non-linear activation components of classical neural networks of the form $L(x) = \phi(Wx + b)$ to be directly implemented in quantum as $L(|x\rangle) = |\phi(Wx + b)\rangle$. The classifiers are composed of a classical feed-forward neural network, a quantum data encoding circuit, and a CV quantum neural network circuit. On a truncated MNIST dataset of 600 samples, a 4-qumode hybrid classifier achieves 100% training accuracy.

Keywords: quantum computing, quantum machine learning, quantum neural networks, continuous variable quantum computing, photonic quantum computing, classical quantum hybrid model, quantum MNIST classification

Dated: February 2022

1. INTRODUCTION

Unlike the original assumption that quantum computers would replace classical computers, quantum processing units (QPUs) are emerging as task-specific special purpose processing units much like Graphics Processing Units. The currently available QPUs are called near-term quantum devices because they are not yet fully fault-tolerant and are characterized by shallow and short quantum circuits. Nonetheless, the availability of these devices allows for active research on quantum algorithms specific for these devices, especially in quantum chemistry, Gaussian boson sampling, graph optimization, and quantum machine learning.

The QPUs are based on two different theoretical models of quantum computing: the discrete variable (qubit-based) model and the continuous variable (CV) model [38]. The discrete variable model is an extension of the computational space from $\{0, 1\}$ to a complex projective Hilbert space $\mathbb{C}P^1$, which is the surface of a projective sphere. The computational basis is composed of two elements $\{|0\rangle, |1\rangle\}$ [21]. The CV model is an extension of the computational space to an infinite dimensional Hilbert space whose computational basis is infinite, i.e., $\{|0\rangle, |1\rangle, \ldots, |n\rangle, \ldots\}$ [38]. The quantum state of an information channel (qumode) is represented by an infinite complex projective linear combination of these basis states. In practice, it is approximated by by a finite linear combination using only a finite basis $\{|0\rangle, |1\rangle, \ldots, |n\rangle$. The number of basis states used for the approximation is called cutoff dimension. On an $m-$qumode system with cutoff dimension n, the computational state space is of dimension n^m. By varying cutoff dimension and the number of qumodes used for computation, we can control the dimension of the computational space.

Implementing machine learning algorithms on quantum computers is an active areaa of research. Quantum machine learning algorithms can be implemented on variational quantum circuits with parametrized quantum gates [28]. A quantum circuit is a collection of quantum gates and the change of states induced by the circuit on the initial quantum state is considered quantum computation. The results of quantum computing extracted via measurement are incorporated into optimization and parameter update computations performed on classical circuits [28].

Quantum neural networks (QNN), a subset of quantum machine learning, follow the same architectural frame work: QNN on a QPU and optimization on a CPU. The main components of classical neural networks described as $L(x) =$

$\phi(Wx+b)$ are the affine transformation $Wx+b$ and the non-linear activation function $\phi(\cdot)$. In the qubit model, all the available unitary gates are linear in nature hence a direct implementation of bias addition and non-linear activation function is not feasible. In the CV model, the displacement gate implements bias addition and the Kerr gate, non-linear activation function, hence a direct translation of $L(x) = Wx+b$ to its quantum version $L(|x\rangle) = |\phi(Wx+b)\rangle$ is naturally embedded in the model.

The proposed CV MNIST classifiers are built on Xanadu's X8 simulator, which simulates quantum computation on an $8-$qumode photonic quantum computer. The ability to control the size of output vectors based on the number of qumodes and the notion of cutoff dimension allows for producing one-hot encoded labels of MNIST dataset. Different architectures on $2-$qumodes, $3-$qumodes, up to $8-$qumodes are introduced. They are classical-quantum hybrid networks with classical feed-forward neural networks, quantum data encoding circuits, and CV quantum neural network circuits according to the CV binary classifier proposed by Killoran et al [14]. The quantum machine learning software library, Pennylane [5], is used for the quantum circuit and Tensorflow Keras is used for the classical circuit and optimization. The classifiers achieve above 95% training accuracy.

This paper is organized in the following manner: In Section 2, the CV model of quantum computing is discussed, especially the infinite dimensionality of the computation state space and how the notion of of cutoff dimension allows us to determine the dimension of the actual computational state. In Section 3, we examine how CV quantum neural networks naturally implement classical neural networks. In Section 3.2, the architecture of the binary classifier in ”Continuous variable quantum neural networks” is examined [14]. In Section 4, the structure of the 8 hybrid classifiers and the experimental results are presented.

2. CV QUANTUM COMPUTING

The quantum state space of the CV model is an infinite dimensional Hilbert space, which is approximated by a finite dimensional space for practical information processing purposes. Under phase space representation of the model, the computational basis states are Gaussian states as opposed to single points. Therefore the quantum gates taking these Gaussian states to Gaussian states offer richer arrays of transformations than just unitary (linear) transformations as in the qubit model.

The CV model is based on the wave-like property of nature. It uses the quantum state of the electromagnetic field of bosonic modes (qumodes) as information carriers [7]. Its physical implementation is done using linear optics, containing multiple qumodes [15]. Each qumode contains a certain number of photons probablistically, which can be manipulated with quantum gates.

A photon is a form of electromagnetic radiation. The position wave function $\Psi(x)$ describes the light wave electromagnetic strength of the qumode depending on the number of photons present. It is a complex valued function on real valued variables: $\Psi : \mathbb{R} \to \mathbb{C} : x \mapsto \alpha$, where x is the position(s) of the photons. The wave function with more than one photon displays the constructive and destructive interactions between the light waves of the photons.

Phase space representation of the quantum state of a linear optical system describes the state using the position x and momentum p variables of the photons in the given qumode system. The quasi-probablity distribution of x and p are plotted on the $xp-$plane, using the Wigner function. It is given by

$$W(x,p) = \frac{p}{h} = \frac{1}{h}\int_{-\infty}^{\infty} e^{-\frac{ipy}{\hbar}}\Psi\left(x+\frac{y}{2}\right)\Psi^*\left(x-\frac{y}{2}\right)dy$$

where $h = 6.62607015 \times 10^{-34}$ is the Plank constant and $\hbar = 6.582119569 \times 10^{-16}$ the reduced Plank constant [20].

This Wigner function is applied to the position wave function $\Psi_k(x)$, where k denotes the number of photons present in a qumode, to create Fock basis, also known as number basis. The image of the Wigner functions $W_0(x_0,p_0), W_1(x_1,p_1), \ldots, W_5(x_5,p_5)$ where x_k and p_k represent the position and momentum variables with k photons present in the system is shown in Figure 1.

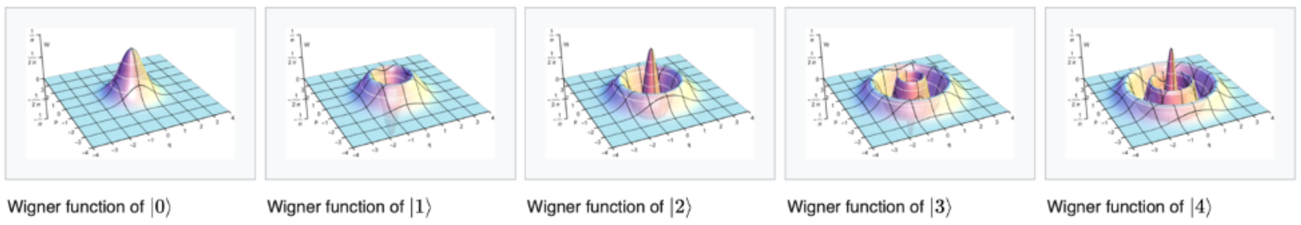

Figure 1. Fock basis: Gaussian states image source

They are used as computational basis states expressing the quantum state of a system. The quantum state $|\psi\rangle$ of a qumode is expressed as a superposition of Fock basis states:

$$|\psi\rangle = c_0|0\rangle + c_1|1\rangle + \ldots + c_n|n\rangle + \ldots, \sum_{k=0}^{\infty} \|c_k\|^2 = 1$$

where c_k is the probability amplitude of basis $|k\rangle$.

In practice, there are not going to be an infinite number of photons physically present in a qumode. We approximate $|\psi\rangle$ with $|\hat{\psi}\rangle$ by cutting off the trailing terms. The number of Fock basis states we use to approximate the true state $|\psi\rangle$ is called "cutoff dimension". Let n be cutoff dimension. Then the approximating state $|\hat{\psi}\rangle$ is in superposition of $|0\rangle, |1\rangle, \ldots, |n-1\rangle$, i.e.,

$$|\psi\rangle = c_0|0\rangle + c_1|1\rangle + \ldots + c_{n-1}|n-1\rangle \text{ where} \sum_{k=0}^{n-1} \|c_k\|^2 = 1.$$

In vector representation, the state is expressed as a column vector of size n =cutoff dimension, with the k^{th} entry being c_k.

A multi-qumode system is represented by the tensor product of individual qumode states: $|\psi_0\rangle \otimes |\psi_1\rangle \otimes \ldots \otimes |\psi_{m-1}\rangle$, where m is the number of qumodes. When these states are approximated by cutoff dimension n, the computational basis of the resulting quantum state is of size n^m.

$$|\psi_0\rangle \otimes |\psi_1\rangle \otimes \ldots \otimes |\psi_{m-1}\rangle = \\ d_0|00\ldots0\rangle + d_1|00\ldots1\rangle + d_{n^m-1}|n-1, n-1, \ldots, n-1\rangle$$

Fock basis states constitute the eigenstates of the number operator $\hat{n} := \hat{a}^\dagger \hat{a}$ where $\hat{a}^\dagger$ is called the constructor and $\hat{a}$ is called the annihilator [38]. Their matrix representation is given by

$$\hat{a}^\dagger = \begin{bmatrix} 0 & 0 & 0 & \ldots & 0 & 0 \\ \sqrt{1} & 0 & 0 & \ldots & 0 & 0 \\ 0 & \sqrt{2} & 0 & \ldots & 0 & 0 \\ 0 & 0 & \sqrt{3} & \ldots & 0 & 0 \\ \vdots & & \ldots & \ddots & & \vdots \\ 0 & 0 & 0 & \ldots & \sqrt{n-1} & 0 \end{bmatrix} \text{ and}$$

$$\hat{a} = \begin{bmatrix} 0 & \sqrt{1} & 0 & 0 & \ldots & 0 \\ 0 & 0 & \sqrt{2} & 0 & \ldots & 0 \\ 0 & 0 & 0 & \sqrt{3} & \ldots & 0 \\ \vdots & & \ldots & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & \ldots & \sqrt{n-1} \\ 0 & 0 & 0 & 0 & \ldots & 0 \end{bmatrix}$$

Note that the constructor $\hat{a}^\dagger|k\rangle = \sqrt{k+1}|k+1\rangle$ for $k \geq 0$ indeed constructs the next level Fock basis state and the annihilator $\hat{a}|0\rangle = 0$, $\hat{a}|k\rangle = \sqrt{k}|k-1\rangle$ for $k \geq 1$ annihilates the the current level Fock basis state to the lower.

The product of $\hat{a}^\dagger$ and $\hat{a}$ returns a matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 2 & 0 & \ldots & 0 \\ \vdots & & \ldots & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & n-2 & 0 \\ 0 & 0 & 0 & 0 & 0 & n-1 \end{bmatrix}$$

which we denote as "number operator" $\hat{n}$. The Fock basis states $|k\rangle$ are indeed eigenstates of the number operator as shown in $\hat{n}|k\rangle = k|k\rangle$ where $k \in \{0, 1, \ldots, n-1\}$ for cutoff dimension= n.

Standard Gaussian gates in the CV model are of the form $U = e^{-iHt}$ where H represents the Hamiltonian of the system, describing the total energy as the sum of kinetic and potential energy. In the finite quantum state space of a qumode approximated by cutoff dimension n, this infinite sum representing a quantum gate is also approximated by a finite matrix exponential. The matrix exponential e^{-iHt} is of the form

$$\hat{U} = \sum_{k=0}^{n-1} \frac{(-iHt)^k}{k!} = I - iHt + \frac{(-iHt)^2}{2!} + \ldots + \frac{-iHt^{n-1}}{(n-1)!}$$

Some of the standard Gaussian gates taking Gaussian states to Gaussian states are listed below.

Squeezer with parameter $z : S(z) = exp\left(\frac{z^*\hat{a}^2 + z\hat{a}^{\dagger 2}}{2}\right)$

Rotation with parameter $\phi : R(\phi) = exp\left(i\phi\hat{a}^\dagger\hat{a}\right)$

Displacement with parameter $\alpha : D(\alpha) = exp\left(\alpha\hat{a}^\dagger - \alpha^*\hat{a}\right)$

Another Gaussian gate, the beamsplitter, is a two-qumode gate with parameters θ and ϕ:

$$B(\theta, \phi) = exp\left(\theta\left(e^{i\phi}\hat{a}\hat{b}^\dagger + e^{-i\phi}\hat{a}^\dagger\hat{b}\right)\right)$$

where $\hat{b}^\dagger$ and $\hat{b}$ are constructor and annihilator of the 2nd qumode respectively.

Measurement is done via counting the number of photons present in each qumode with a photon detector. Xanadu's Pennylane offers a suite of measurement methods as outlined in Table 1. The size of the output vectors are affected by the number of qumodes used for computation and the notion of cutoff dimension, denoted by n. Currently available QPU by Xanadu, $X8$, offers 8 qumodes.

Measurement method	output size per qumode	output size for $m-$qumodes
expectation value	1	m
variance	1	m
probability	n	n^m

The measurement methods used in the proposed MNIST classifiers are the expectation value method and the probability memthod.

The expectation value method produces a single real-valued output. Let $|\psi\rangle = \begin{bmatrix}\psi_0 \\ \psi_1\end{bmatrix}$ be the quantum state of a multi-qumode system after desired quantum computational operations are performed. The expectation value measurement method returns $\langle\psi|A|\psi\rangle$ where the operator A is usually the Pauli$-X$, Pauli$-Y$, or Pauli$-Z$ gate. The expectation value of the Pauli$-X$ matrix is

$$\begin{aligned}\langle\psi|X|\psi\rangle &= \begin{bmatrix}\psi_0^* & \psi_1^*\end{bmatrix}\begin{bmatrix}0 & 1 \\ 1 & 0\end{bmatrix}\begin{bmatrix}\psi_0 \\ \psi_1\end{bmatrix} \\ &= \begin{bmatrix}\psi_0^* & \psi_1^*\end{bmatrix}\begin{bmatrix}\psi_1 \\ \psi_0\end{bmatrix} \\ &= \psi_0^*\psi_1 + \psi_1^*\psi_0 \\ &= 2\left(Re(\psi_0)Re(\psi_1) + Im(\psi_0)Im(\psi_1)\right) \in \mathbb{R},\end{aligned}$$

which is a real number. In a multi-qumode system with m qumodes, we can get a vector of size m by getting the expectation value for each qumode

The probabilities method returns the probability of each computation basis state. Suppose an $m-$qumode system has cutoff dimension n for each qumode. Then, there are n^m computational basis states, hence we get a vector of size n^m.

3. QUANTUM NEURAL NETWORKS

Machine learning is a way of extracting hidden patterns from data by learning a set of optimal parameters for a mathematical expression that most closely match the data. The mathematical expression used for pattern extraction is called a machine learning algorithm. An algorithm with an optimal set of parameters, learned via training, is called a model. With near-term devices available on cloud, execution of quantum machine learning (QML) algorithms on quantum computers or simulators is now feasible.

QML algorithms are built with variational circuits i.e., parametrized circuits, composed of quantum gates whose actions are defined by parameters. Training is the process of ”learning” optimal parameters of the gates which produce as accurate inferences as possible for new data samples. The measurement results from a variational circuit run on a QPU are sent to a CPU for parameter optimization, i.e., computation of objective function, gradients, and new parameters. The updated parameters are fed back to the quantum circuit to adjust the parameterized gates for next iteration. The illustration of the process is shown in the figure.

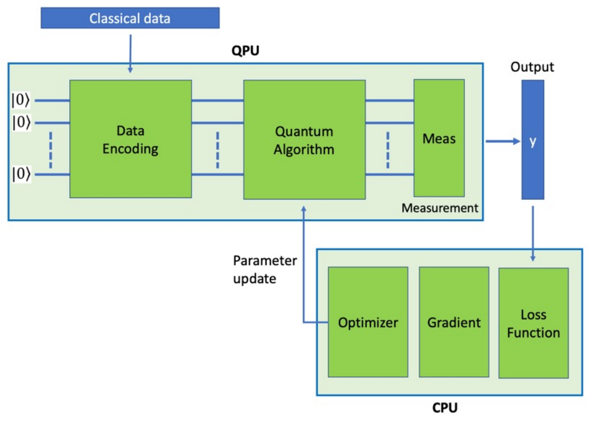

Figure 2. Variational quantum circuit parameter update

Google and Xanadu offer Python based software packages specifically for quantum machine learning: Tensorflow Quantum (Google) and Pennylane (Xanadu) [5].

Neural network is one of the subsets of machine learning algorithms, defined by a stack of layers, each composed of an affine transformation $Wx+b$ and a non-linear activation function $\phi(\cdot)$. Each layer of a neural network can be mathematically described as $L(x) = \phi(Wx+b)$. The output from one layer is then fed as input into the subsequent layer and the entire network is a composition of different layers: $L(x) = L_m \circ L_{n-1} \circ \ldots L_1(x)$. The entries of the matrix W and the bias vector b for each layer are learned as parameters through an iterative training process given an objective function. The goal is to find an optimal set of parameters $\{W_1, W_2, \ldots, W_m, b_1, b_2, \ldots, b_m\}$ for a network of m layers.

In quantum neural networks, the objective is to implement the classical mathematical expression $L(x) = \phi(Wx+b)$ as a quantum state $L(|x\rangle) = |\phi(Wx+b)\rangle$. In converting classical neural networks into quantum circuits, the key components are:

- data encoding: $x \to |\psi(x)\rangle$
- affine transformation $W|\psi(x)\rangle + |b\rangle$
- non-linear activation function $\phi(|\cdot\rangle)$

In the qubit model, all available unitary gates are linear.

Hence a direct way of implementing the bias addition component and the non-linear activation function component of classical neural networks into quantum is absent in the model.

In the CV model, however, the displacement gate and the Kerr gate allow for a direct translation from classical to quantum.

3.1 Continuous variable QNN

Naturally embedded in the CV model are quantum gates to directly implement the expression $L(|x\rangle) = |\phi(Wx+b)\rangle$.

The affine transformation $Wx+b$ is implemented by the composition $D \circ U_2 \circ S \circ U_1$, where U_k denotes the k^{th} interferometer, S a set of m squeezers, D a set of m displacement gates. The activation function $\phi(|\cdot\rangle)$ is implemented by a set of Kerr gates, which are non-linear. The composition $\phi \circ D \circ U_2 \circ S \circ U_1$ acting on a quantum state $|x\rangle$ gives us the desired state $L(|x\rangle) = |\phi(Wx+b)\rangle$. The schematic of the circuit is shown below.

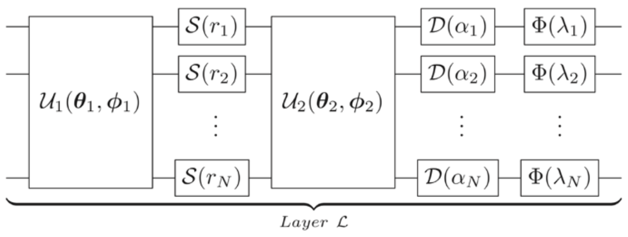

Figure 3. CV quantum neural network architecture [14]

The interferometer U_k on an $m-$qumode system is composed of $m-1$ beamsplitters and m rotation gates as shown in the figure.

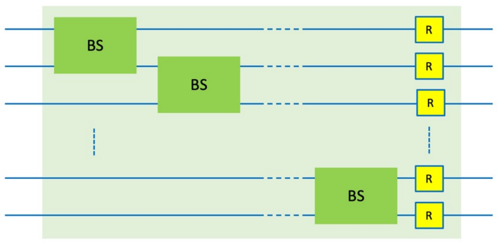

Figure 4. Make-up of an interferometer

The action of a phaseless interferometer U_k on the quantum state $|x\rangle = \otimes_{k=1}^{m} |x_k\rangle$ has an effect of an orthogonal matrix acting on $|x\rangle$ [14]. Orthogonal matrices are just unitary matrices with real entries, inducing length-preserving rotations. Then the transpose of an orthogonal matrix represents the reverse rotation of the original matrix, thus orthogonal. Then the composition $U_2 \circ S \circ U_1$ can be considered as the composition $O_2 \circ S \circ \left(O_1^T\right)^T$, where O_2 and O_1^T are orthogonal.

Let W be the linear transformation matrix we want to implement with a quantum circuit. Any matrix W can be factorized using singular value decomposition (SVD) as $W = U\Sigma V^*$, where U and V are orthogonal and Σ is diagonal [43]. The parameterized squeezer $S(r_k)$ acts on the quantum state $|x_k\rangle$ of each k^{th} qumode as $S(r_k)|x_k\rangle = \sqrt{e^{-r_k}}|e^{-r_k}x_k\rangle$. Collectively they have an effect of a diagonal matrix $S = S(r_1) \otimes S(r_1) \otimes \ldots \otimes S(r_m)$ acting on $|x\rangle = \otimes_{i=1}^{m} |x_k\rangle$. The composition $U_2 \circ S \circ U_1$ implements a quantum version of the linear transformation matrix W [14].

The bias addition is realized with displacement gates D. The displacement gate has an effect

$$D(\alpha_k)|\psi_k\rangle = |\psi_k + \sqrt{2}\alpha_k\rangle$$

for each k^{th} qumode. Then $D(\alpha)|\psi\rangle = |\psi + \sqrt{2}\alpha\rangle$ collectively for $\alpha^T = [\alpha_1, \alpha_2, \ldots, \alpha_m]$. For some desired bias b, let $\alpha = \frac{b}{\sqrt{2}}$, then the collection of displacement gates implements the bias addition. The composition $D \circ U_2 \circ S \circ U_1$ acting on the quantum state $|x\rangle$ gives us the affine transformation

$$D \circ U_2 \circ S \circ U_1 |x\rangle = |O_2 \Sigma O_1 x + b\rangle = |Wx+b\rangle .$$

The non-linear activation function $\phi(\cdot)$ is realized with Kerr gates. The Kerr gate, parameterized by the parameter κ, is a non-linear transformation gate. Let n be the cutoff dimension and m the number of qumodes. For the quantum state $|\psi\rangle$ of one qumode, which is a superposition of n Fock basis states, the Kerr gate with parameter κ has an effect

$$K(\kappa)|\psi\rangle = \begin{bmatrix} e^{i\kappa 0^2} & 0 & \ldots & 0 \\ 0 & e^{i\kappa 1^2} & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & e^{i\kappa(n-1)^2} \end{bmatrix} \begin{bmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{n-1} \end{bmatrix}$$
$$= \begin{bmatrix} \psi_0 \\ e^{i\kappa 1^2}\psi_1 \\ \vdots \\ e^{i\kappa(n-1)^2}\psi_{n-1} \end{bmatrix},$$

which is non-linear.

Together, the circuit $L = \Phi \circ D \circ U_2 \circ S \circ U_1$ gives us a quantum version $L(|x\rangle) = |\Phi(Wx+b)\rangle$ of a classical neural network $L(x) = \Phi(Wx+b)$.

3.2 CV Binary Classifier

The binary classifier outlined in ”Continuous-variable quantum neural networks” is a classical and quantum hybrid network [14].

The dataset used contains 284,806 genuine and fraudulent credit card transactions with 29 features, out of which only 492 are fraudulent. The dataset is truncated to 10 features as per the paper and 1,968 samples with 1:3 ratio of fraudulent vs. genuine.

The proposed classical-quantum hybrid model has a classical neural network taking input vectors of size 10 and outputting vectors of size 14, quantum encoding circuit, and a 2-qumode quantum neural network which outputs vectors of size 2. We can regard the output vectors as one-hot encoding of binary classification of fraudulent vs. genuine. The architecture of the hybrid network is

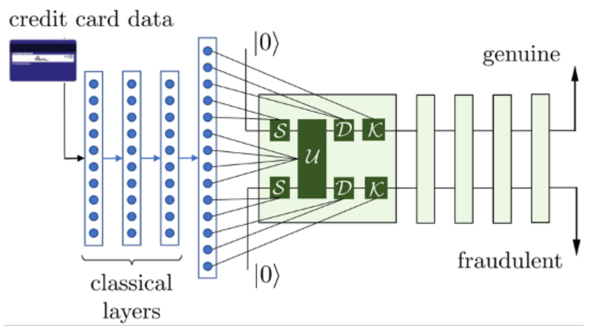

Figure 5. Binary hybrid classifier circuit [14]

The data flow of the circuit is

- Classical network: 2 hidden layers with 10 neurons, each using Exponential Linear Units (ELU) as activation function. Output layer with 14 neurons.
- Data encoding: Output vector from the classical network is converted to a quantum state by the circuit - squeezers, interferometer, displacement gates, and Kerr gates
- Quantum network: 4 layers of QNN. Each layer is composed of interferometer 1, squeezers, interferometer 2, displacment gates, and Kerr gates.
- Measurement: The expectation value of the Pauli$-X$ gate $\langle\phi_k|X|\phi_k\rangle$ is evaluated for each qumode state$|\phi_k\rangle$ for the k^{th} qumode.

The experiment yields 97% training accuracy.

Code: Keras-Pennylane implementation

4. CV MNIST CLASSIFIERS

The multi-classifier models presented in this section are inspired by the CV binary classifier architecture described in the section above [14]. They are run on Xanadu's $X8$ photonic quantum simulator, made up of $8-$qumodes. There are two models classifying 8 classes: digits $0,1,\ldots,7$ and five models classifying 10 classes: digits $0,1,\ldots,9$. The measurement outputs from these models are interpreted as one-hot encoded predictions of image labels.

The classifiers on 8 classes are:

num of qumodes	cutoff dim	measurement method	output size
3	2	probability	$2^3 = 8$
8	2	expectation value	$1 \times 8 = 8$

A classical neural network is used as a pre-processing step to reduce the original image matrices of size $28 \times 28 = 784$ to vectors of lower length that the data encoding circuit can accommodate. For the data encoding circuit, squeezers, an interferometer, displacement gates, and Kerr gates are used. For the quantum neural network, the quantum circuit implementing $L(|x\rangle) = |\phi(Wx+b)\rangle$ as in the binary classifier is used.

The 10-class classifiers are built on $2,3,\ldots,6-$qumodes. The label $k \in \{0,1,\ldots,9\}$ of an image matrix, when converted into a one-hot encoded vector, becomes a vector of length 10 with all zeros but the k^{th} entry as 1. The cutoff dimensions are selected so that the output vectors exceed 10 in length. For each classifier, a different number of zeros are padded to the one-hot encoded labels to match the output size of the circuit. The cutoff dimension used for each classifier and the size of output vectors are shown in the table.

num of qumodes	cutoff dim	output size	number of padding 0's
2	4	$4^2 = 16$	6
3	3	$3^3 = 27$	17
4	2	$2^4 = 16$	6
5	2	$2^5 = 32$	22
6	2	$2^6 = 64$	54

The architecture in Figure 6 depicts the data flow of image matrix $\rightarrow$ classical layers $\rightarrow$ reduced output vectors $\rightarrow$ data encoding $\rightarrow$ QNN $\rightarrow$ measurement $\rightarrow$ output vectors as one-hot encoded labels.

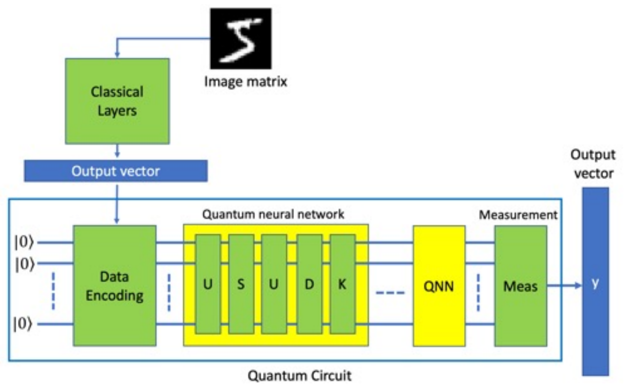

Figure 6. MNIST multi-classifier architecture

The boxes U, S, D, and K represent interferometers, a set of m squeezers, a set of m displacement gates, and a set of m Kerr gates respectively, where m =the number of qumodes.

4.1 Classical layers

Classical feed forward neural networks are used for pre-processing of data images to reduce the size of $28 \times 28 = 784$ to smaller size vectors fitting the number of parameters available for data encoding. The image matrices are flattened to vectors of size $28 \times 28 = 784$ and then reduced to vectors of smaller sizes through Keras dense layer operations with activation function "ELU". The output vectors are then encoded as quantum states by the data encoding quantum circuit.

4.2 Data encoding

The output vectors from the classical neural network are in classical states. The quantum data encoding circuit converts classical states into quantum states. The quantum gates used for data encoding are squeezers, interferometer, displacement gates, and Kerr gates. The entries of a classical vector are used as the parameters of these parameterized quantum gates.

An interferometer is composed of beamsplitters on pairs of adjacent qumodes and rotation matrices. Squeezers $S(z)$ and displacement gates $D(\alpha)$ can be considered either one parameter gates or two parameter gates when we view the parameters $z, \alpha \in \mathbb{C}$ in Euler formula $z = a + bi = r(\cos\phi + i\sin\phi) = re^{i\phi}$. For the purpose of data encoding, we use them as two-parameter gates.

The number of parameters that these gates can accommodate for $m-$qumode circuits are $8m-2$, where m is the number of qumodes.

squeezers	BS	rotation	displacement	Kerr
$2m$	$2(m-1)$	m	$2m$	m

Based on these values, the size of the classical network output vectors was determined.

4.3 Quantum circuit

The QNN circuit implements a quantum version of the classical neural network $L(x) = \phi(Wx+b)$ as

$$L(|x\rangle) = |\phi(Wx+b)\rangle = \phi \circ D \circ U_2 \circ S \circ U_i\,|x\rangle$$

where $|x\rangle$ is a quantum data encoded state of the original data vector x, U_k interferometers, S squeezers, D displacement gates, and $\phi(\cdot)$ Kerr gates respectively. The number of parameters per a collection of gates on $m-$qumodes are

U_k	S	D	$\phi(\cdot)$	total
$2(m-1)+m$	m	m	m	$2\cdot 2(m-1)+5m$ $=9m-4$

On the 8-qumode classifier, 2 layers of QNN are applied. On the rest of classifiers, 4 layers are applied. The number of parameters for the varying number of qumodes are shown in the table.

num of qumodes	per layer	total
2	14	56
3	23	92
4	32	128
5	41	164
6	50	200
8	68	136

4.4 Measurement

Pennylane offers three different measurement methods for CV-based computations: expectation value, variance, and probabilities. The expectation value and variance methods yield a single-valued result for each qumode. The probability method yields vectors of size n^m, where n =cutoff dimension and m =the number of qumodes.

For the eight-qumode model, the expectation value method was used. The result is a vector of length $8 : [\langle\psi_0|X|\psi_0\rangle, \langle\psi_1|X|\psi_1\rangle, \ldots, \langle\psi_7|X|\psi_7\rangle]$ where $\langle\psi_k|X|\psi_k\rangle$ represents the expectation value measurement result of the k^{th} qumode. It is then translated as a one-hot encoded label vector of the corresponding image.

For the rest of the models, the probability method was used.

4.5 Parameter update

With the Pennylane Tensorflow plug-in functionality, the quantum circuit is converted into a Keras layer and added to the classical layers. Then Keras' built-in loss functions and optimizers can be used for parameter update. For most of the models, Categorical Crossentropy is used for loss function and Stochastic Gradient Descent is used for optimizer. For the 8-qumode classifier, the Mean Squared Error loss function performed better. The updated parameters are then used as the parameters of the quantum gates for the subsequent iteration of training.

4.6 Experimetal results

The $4-$qumode classifier yielded the best result of 100% training accuracy on 600 data samples. Accuracy comparison with the qubit-based binary classifiers using Tensorflow Quantum and Qiskit is listed in the table below.

	Tensorflow Q	Qiskit - PyTorch	Pennylane - Keras
Number of classes	2	2	10
Number of samples	10,338	100	600
Training accuracy	89.92%	100%	100%

All the classifiers tested achieve above 95% training accuracy. For the 8-qumode classifier, 300 samples and 2 layers of QNN were used with 50 epochs. For the rest of the classifiers, 600 samples and 4 layers of QNN were used. With the 4-qumode classifier, training accuracy of 100% is achieved in 70 epochs.

The loss and accuracy for the models with varying number of qumodes are listed below.

num of qumodes	learning rate	training loss	training accuracy
2	0.02	0.4966	97.14%
3*	0.05	0.0563	99.90%
3	0.05	0.4567	98.04%
4	0.03	0.1199	100.00%
5	0.02	0.2057	98.40%
6	0.02	0.2055	98.22%
8	0.1	0.0199	99.77%

The second line 3* indicates an $8-$class classifier.

All of these classifiers followed the typical loss and accuracy graphs depicted in Figure 7. The validation accuracy is not as high as the training accuracy, indicating "over-fitting". Further experiments with different hyperparameters or the use of regularizer are called for.

Code: Keras-Pennylane implementation

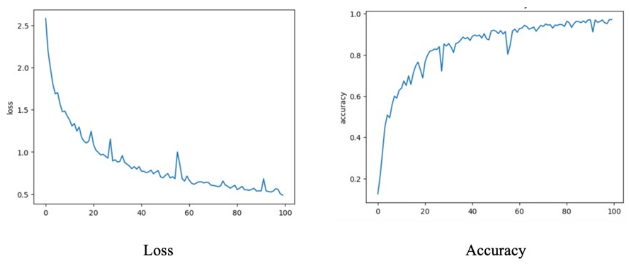

Figure 7. MNIST multi classifier experimental results

5. CONCLUSION

In this pape, classical and CV quantum hybrid classifiers using different number of qumodes and cutoff dimensions were examined. The quantum gates available in the CV model allow a natural implementation of a quantum neural network layer $L(|x\rangle) = |\phi(Wx+b)\rangle$. The flexibility of different measurement methods to output vectors of different lengths allows the networks to produce results that are interpreted as one-hot encoded labels of MNIST image dataset.

There is a limitation in encoding classical data into quantum states on near-term devices due to the number of qumodes, which is currently eight on Xanadu's X8 photonic QPU. Classical networks were used to reduce the number of entries in the image matrices for quantum data encoding. Although the role of the classical network is for pre-processing, the majority of parameters that are learned through the training process is on the classical network side. One way of encoding all the entries of an image matrix would be iterating through smaller sections of the image, which naturally segues to convolutional operations. A combination of quantum convolutional network layers and quantum neural network layers is a way of implementing a purely quantum network.

In implementing machine learning algorithms in quantum, the CV model offers many advantages over the qubit-based model. The quantum computational state space of the CV model is infinite while it is finite in the qubit-based model. The ability to define the dimension of the CV quantum computational space in approximating the original infinite computational space avails users of added freedom and flexibility in experimenting quantum algorithms. Photonic quantum computers, which are a version of physical implementation of the CV model can be easily incorportated into the current computing infrastructure because of their operability at room temperature.

References

[1] Scott Aaronson and Alex Arkhipov, (2011) *The computational complexity of linear optics*, Proceedings of the 43rd annual ACM Symposium on Theory of Computing.

[2] Scott Aaronson and Lijie Chen (2017) *Complexity-Theoretic Foundations of Quantum Supremacy Experiments*, Proceedings of the 32nd Computational Complexity Conference, No. 22, Pages 1–67.

[3] C. Adami, N. Cerf, (1999) *Quantum Computation with Linear Optics*, Quantum Computing and Quantum Communications. Lecture Notes in Computer Science, vol 1509. Springer.

[4] J. Arrazola et al., (2021) *Quantum circuits with many photons on a programmable nanophotonic chip*, Nature volume 591, pages 54–60.

[5] Ville Bergholm et al., (2018) *PennyLane: Automatic differentiation of hybrid quantum-classical computations*, arXiv:1811.04968

[6] Jacob Biamonte, Peter Wittek, Nicola Pancotti, P. Rebentrost, Nathan Wiebe, Seth Lloyd, (2017) *Quantum Machine Learning*, Nature 549.

[7] Samuel Braunstein and Peter van Loock, (2005) *Quantum information with continuous variables*, Reviews of Modern Physics 77, 513.

[8] William Case, (2008) *Wigner functions and Weyl functions for pedestrians*, American Journal of Physics 76, 937.

[9] Siddhartha Das, George Siopsis, and Christian Weedbrook, (2018) *Continuous-variable quantum Gaussian process regression and quantum singular value decomposition of non-sparse low rank matrices*, Physical Review A 97, 022315

[10] Abhinav Deshpande et al., (2021) *Quantum computational supremacy using Gaussian boson sampling*, arXiv:2102.12474.

[11] Edward Farhi and Hartmut Neven, (2018), *Classification with Quantum Neural Networks on Near Term Processors*, arXiv:1802.06002v2.

[12] Alessandro Ferraro, Stefano Olivares, Matteo G. A. Paris, (2005) *Gaussian states in continuous variable quantum information*, Bibliopolis.

[13] Craig Hamilton, Regina Kruse, Linda Sansoni, Sonja Barkhofen, Christine Silberhorn, Igor Jex, (2017) *Gaussian boson sampling*, Physic Review Letters.

[14] Nathan Killoran, Thomas Bromley, Juan Miguel Arrazola, Maria Schuld, Nicolás Quesada, and Seth Lloyd (2019) *Continuous variable quantum neural networks*, Physical Review Research 1, 033063.

[15] E. Knill, R. Laflamme, and G. J. Milburn (2001) *A scheme for efficient quantum computation with linear optics*, Nature volume 409, pages 46–52.

[16] Dave Krebs, (2007) *Introduction to Kernel Methods*, cs.pitt.edu.

[17] Seth Lloyd and Samuel Braunstein, (1999) *Quantum computation over continuous variables*, Physics Review Letters 82, 1784.

[18] Seth Lloyd, Maria Schuld, Aroosa Ijaz, Josh Izaac, and Nathan Killoran, (2020), *Quantum embeddings for machine learning*, arXiv:2001.03622.

[19] M. Menotti et al., (2018), *Nonlinear coupling of linearly uncoupled resonators*, Physics Review Letters.

[20] Jonas Neergaard-Nielsen, (2008) *Generation of single photons and Schrodinger kitten states of light*, PhD Thesis, Technical University of Denmark.

[21] Michael Nielsen and Isaac Chuang, (2010), *Quantum computation and quantum information*, Cambridge University Press.

[22] K. Parthasarathy, (2010), *What is a Gaussian State?*, Communications of Statistical Analysis, Vol 4, Num 2.

[23] Anatoli Polkovnikov, (2010), *Phase space representation of quantum dynamics*, Annals of Physics.

[24] J. Preskill, (2018), *Quantum Computing in the NISQ era and beyond*, Quantum, 2:79.

[25] Patrick Rebentrost et al, (2019), *Quantum gradient descent and Newton's method for constrained polynomial optimization*, New Journal of Physics 21 073023.

[26] Maria Schuld, (2021), *Supervised quantum machine learning models are kernel methods*, arXiv:2101.11020v2.

[27] Maria Schuld, Ville Bergholm, Christian Gogolm, Josh Izaac, and Nathan Killoran, (2018), *Evaluating analytic gradients on quantum hardware*, Physical Review A.

[28] Maria Schuld, Alex Bocharou, Krista Svore, and Nathan Wiebe, (2018), *Circuit-centric quantum classifiers*, Physical Review A.

[29] Maria Schuld, Mark Fingerhuth, and Francesco Petruccione, (2017), *Implementing a distance-based classifier with a quantum interference circuit*, Europhysics Letters, Volume 119, Number 6.

[30] Maria Schuld and Nathan Killoran, (2019) *Quantum machine learning in feature Hilbert spaces*, Physical Review Letters 122, 040504.

[31] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer, (2021), *Effect of data encoding on the expressive power of variational quantum machine learning models*, Physical Review A 103, 032430.

[32] Yichen Shen, Nicholas C. Harris, Scott Skirlo, Mihika Prabhu, Tom Baehr-Jones, Michael Hochberg, Xin Sun, Shijie Zhao, Hugo Larochelle, Dirk Englund, and Marin Soljačić, (2017), *Deep learning with coherent nanophotonic circuits*, Nature Photonics.

[33] Daiqin Su, Ish Dhand, Lukas G. Helt, Zachary Vernon, and Kamil Brádler. (2019), *Hybrid spatiotemporal architectures for universal linear optics*, Physical Review A 99, 062301.

[34] Alexander Tait, Mitchell Nahmias, Bhavin Shastri, and Paul Prucnal, (2014), *Broadcast and weight: an integrated network for scalable photonic spike processing*, Journal of Lightwave Technology.

[35] K.Tan, M. Menotti, Z. Vernon, J. E. Sipe, M. Liscidini, and B. Morrison1, (2019), *Stimulated four-wave mixing in linearly uncoupled resonators*, Optics Letters.

[36] Ilan Tzitrin, Takaya Matsuura, Rafael Alexander, Guillaume Dauphinais, J. Eli Bourassa, Krishna Sabapathy, Nicolas Menicucci, and Ish Dhand, (2021), *Fault-tolerant quantum computation with static linear optics*, PRX Quantum, Physics Review Journal.

[37] V. Vaidya et al., (2019), *Broadband quadrature-squeezed vacuum and nonclassical photon number correlation from a nanophotonic device*, Science Advances.

[38] Christian Weedbrook, Stefano Pirandola, Raúl García-Patrón, Nicolas J. Cerf, Timothy C. Ralph, Jeffrey H. Shapiro, and Seth Lloyd, (2012), *Gaussian Quantum Information*, Reviews of Modern Physics 84, 621.

[39] Peter Wittek, (2014), *Quantum Machine Learning: What Quantum Computing Means to Data Mining*, Elsevier Insights, Academic Press.

[40] Y. Zhang, M. Menotti, K. Tan, V. Vaidya, D. Mahler, L. Helt, L. Zatti, M. Liscidini, B. Morrison, and Z. Vernon, (2021), *Squeezed light from a nanophotonic molecule*, Nature Communications.

[41] Han-Sen Zhong et al., (2020), *Quantum computational advantage using photons*, Science 10.1126.

[42] (2019), *Quantum Computing: Progress and Prospects*, Consensus Study Report, The National Academic Press.

[43] 18.06SC Linear Algebra Fall, (2011), *Singular Value Decomposition*, MIT OpenCourseWare