Final Report

Name of the Student	Bartu Yaman
Name of the Advisor	Osman Barış Malcıoğlu
Project Title	Quantum State Encoding With Hybrid Neural Networks

Contents

1 INTRODUCTION

Can we find new physics through continuous variable quantum encoding? The intersection of the field of machine learning and continuous variable quantum information gave rise to this fascinating question.

It is well know by now that if a quantum system consists of individually addressable energy eigenstates, such as the internal levels in trapped atoms or the polarization states of electron spins [6], the qubit is represented as a superposition of a two level system. However, there are also quantum systems that do not have evenly-spaced energy levels, such as optical modes [2]. The latter type is usually referred to as continuous variable (CV) systems, where there is no trivial representation of a qubit. The large Hilbert space of each degree of freedom also provides flexibility for encoding the qumodes [7].

On the classical computing side, the recent advancements formed the new field of deep learning [9]. This new field, for example, also proposed exciting methods that take physical laws into account while training called "physics informed neural networks" which can integrate bot data and mathematical model into itself.

The sub-branch of deep learning we use, however, are autoencoders which are already widely used in different fields of physics for solving nonlinear boundary value problems [3], anomaly detection in quantum many-body systems [5] or realizing phenomena from smaller datasets [1] just to name a few. Especially for classical data processing, using an autoencoder, which was introuced in 1988 [12] to address the problem of "backpropagation without a teacher", as a tool for dimension reduction can reduce the overhead in terms of required resources, which is an important direction in mitigating the limiting factor for nearly all of the computational applications currently present.

The crucial point that ties the CV quantum information to deep learning is the fact that the fundamental computational units in deep learning are continuous vectors and tensors which are transformed in high-dimensional spaces, instead of conventional bit registers usually found in digital computing [4]. Even though these computations are being approximated by digital computers right now, specialized hardware that is analog in nature are being engineered as we speak [10].

The ability of using nonclassical effects like superposition, interference and entanglement may possibly prove to be advantageous over classical computing models, which makes exploring the field of quantum computation a worthwhile process. Following from the same though process, exploration of hybrid neural network schemes may provide interesting results [11]. Thanks to its naturally continuous architecture, the CV model solves one of the biggest problems of qubit-based systems which is the fact that they are not exactly continuous, since their measurement outputs are usually discrete. Recently, the first steps were taken to use the CV model for machine learning problems, showing how they can be primitively implemented [8].

In this work, we improve the existing quantum-classical autoencoder architecture [4], propose two new training methods to explore the capabilities of displacement encoding and create a roadmap for the further development of the quantum-classical autoencoder and its applications in computational and experimental physics.

2 METHOD

2.1 Continuous Variable Neural Networks

In the continuous variable model, information is carried in the quantum states of bosonic modes, or qumodes which are the 'wires' of a CV quantum circuit. Since the conjugate variables x and p are treated equally in the CV model, the state of a single qumode is encoded with these two variables (x, p). Quasiprobability distributions can be used to represent qumode states as real-valued functions in the phase space picture, specifically the Wigner quasiprobability distribution is used for representation in this work. Qumode states can also be represented by Fock states, which are the eigenstates of the photon number operator $\hat{n}$ in the Hilbert space formulation. Both the phase space and Hilbert space formulations yield the same results and they use different quantum gates than regular qubits. The Gaussian and non-Gaussian gates used in this work are:

- **Squeezing gate**, that scales $\hat{x}$ to $e^{-r}\hat{x}_\phi$ and $\hat{p}$ to $e^{r}\hat{p}_\phi$.

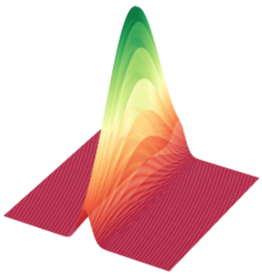

Figure 1: $\mathcal{S}$ gate applied to a vacuum state.

- **Displacement gate**, that shifts $\hat{x}$ by $Re(\alpha)$ and $\hat{p}$ by $Im(\alpha)$ where α is the input.

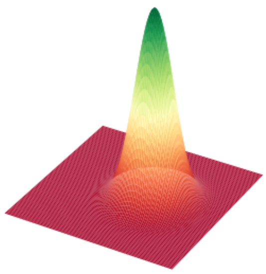

Figure 2: $\mathcal{D}$ gate applied to a vacuum state.

- **Rotation gate**, that shifts $\hat{x}$ to $\hat{x}cos\phi - \hat{p}sin\phi$ and $\hat{p}$ to $\hat{p}cos\phi + \hat{x}sin\phi$.

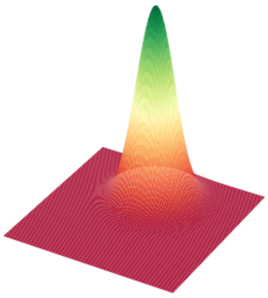

Figure 3: $\mathcal{R}$ gate applied to a vacuum state.

- **Kerr gate**, which rotates the states with $e^{i\phi k\hat{n}^2}$, thus intoduces nonlinearity.

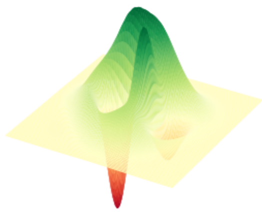

Figure 4: $\mathcal{K}$ gate applied to a vacuum state.

- **Cubic phase gate**, which shifts $\hat{p}$ by $\gamma\hat{x}^2$ where γ is a definable parameter but does not transform $\hat{x}$ in the phase space.

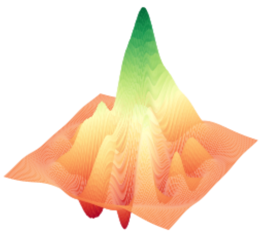

Figure 5: $\mathcal{V}$ gate applied to a vacuum state.

These gates make up the universal gate set that is used to create the quantum neural network layers. A single quantum layer $\mathcal{L}$ can be defined as:

$$\mathcal{L} := \Phi \circ \mathcal{D} \circ \mathcal{U}_2 \circ \mathcal{S} \circ \mathcal{U}_1 \tag{1}$$

where $\mathcal{U}_i = \mathcal{U}_i(\theta, \phi)$ are general linear optical interferometers that contain beam-splitter and rotation gates, $D = \otimes_{i=1}^{N} D(\alpha_i)$ and $\mathcal{S} = \otimes_{i=1}^{N} S(r_i)$ are displacement and squeezing gates and $\Phi = \Phi(\lambda)$ is the Kerr gate. The gate variables $(\theta, \phi, r, \alpha, \lambda)$ form the free parameters of the network, allowing the layers to be trained.

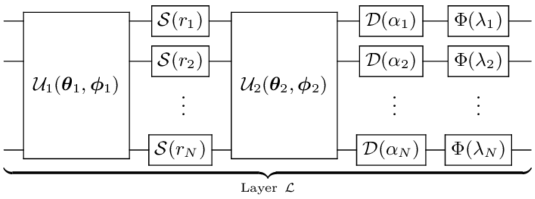

Figure 6: A quantum layer constructed from the universal gate set represented in the circuit form.

The proposed sequence of Gaussian gates $\mathcal{D} \circ \mathcal{U}_2 \circ \mathcal{S} \circ \mathcal{U}_1$ are able to parameterize every possible unitary affine transformation on any number of qumodes. In the phase space picture, these Gaussian transformations has the effect

$$\begin{bmatrix} x \\ p \end{bmatrix} \rightarrow M \begin{bmatrix} x \\ p \end{bmatrix} + \begin{bmatrix} \alpha_r \\ \alpha_i \end{bmatrix} \tag{2}$$

This sequence thus has the role of a 'fully connected' matrix transformation. Adding the non-Gaussian Kerr gate Φ adds universality to the transformation that still keeps the quantum circuit unitary. Combining all of these transformations, we get

$$\mathcal{L}(z) = \Phi(Mz + \alpha) \tag{3}$$

Notice that this function has exactly the same form as the classical neural network layer function

$$\mathcal{L}(x) = \phi(Wx + b) \tag{4}$$

Therefore as it is the case with classical neural networks layers, quantum layers can be stacked on top of each other to form deep networks.

This architecture allows the encoding of classical data as well, which is the heart of this work. The classical data can be encoded to the quantum feature space by applying a displacement $\mathcal{D}(x)$ to the vacuum state to obtain $\mathcal{D}(x) |0\rangle$. This means that the displacement operator $\mathcal{D}(x)$ should be the first layer of the network to encode the classical data to quantum feature space, then allowing the subsequent layers to use the output of the previous layers as input.

2.2 The Quantum Classical Autoencoder

In light of this continuous variable formulation of neural networks, it is possible to construct an autoencoder that uses both classical and quantum layers. Conventional autoencoders are made up from two parts, an encoder and a decoder. Their objective is to learn the input data perfectly to output the same data, essentially performing an identity operation on the input. Similarly for the quantum-classical autoencoder, the goal is to find a continuous phase space encoding of the Fock states that were given as classical inputs. This can be achieved by encoiding the inputs states to a latent space and decoding them back to Fock state form by displacing vacuum states using the latent space variables.

The general form of the quantum-classical autoencoder consists of a classical encoder and a quantum decoder. The classical encoder accepts any real linear combinations in the $|0\rangle, |1\rangle, |2\rangle$ subspace as inputs and they are followed by six hidden layers with dimension five, using the exponential linear unit as the activation function. The quantum decoder accepts displaced vacuum states as inputs and they are followed by a total of 25 quantum layers with controllable variables $(\theta, \phi, r, \alpha, \lambda)$. Output values of the classical encoder are used as inputs to the displacement operator $\mathcal{D}(r, \phi)$, bridging the gap between classical and quantum layers by encoding the classical data to the quantum feature space. An illustration of the described hybrid autoencoder is below.

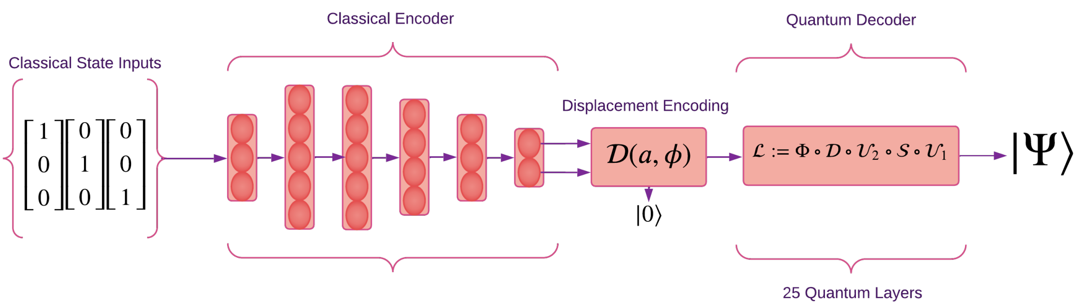

Figure 7: Illustration of the general quantum-classical autoencoder architecture.

To train the hybrid autoencoder, one-hot encoded vector inputs $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$ representing the first three Fock states $|0\rangle$, $|1\rangle$ and $|2\rangle$ are fed to the classical encoder. As described earlier, for in input state $|i\rangle$, suppose that the classical encoder outputs the vector (x_i, p_i) which is used to displace the vacuum in one mode, i.e. carrying out the operation $\mathcal{D}(\alpha_i)|0\rangle$ with $\alpha_i = (x_i, p_i)$. Then the final output of the quantum decoder is the state $|\Psi_i\rangle = \mathcal{V}\mathcal{D}(\alpha_i)|0\rangle$ where $\mathcal{V}$ is the product of unitary transformations resulting from the quantum layers.

Defining the normalized projection

$$|\psi_i\rangle = \frac{\Pi_3 |\Psi_i\rangle}{||\Pi_3 \Psi_i||} \tag{5}$$

onto the subspace of the first three Fock states, with Π_3 being the projector $\Pi_N = \Sigma_{i,j=0}^{N-1} |i\rangle \langle i| \otimes |j\rangle \langle j|$. Since the objective is to train the network so that $|\psi_i\rangle$ is close to $|i\rangle$ where closeness is measured using the fidelity $| \langle i|\psi_i\rangle |^2$, we can define a cost function

$$C = \Sigma_{i=0}^{2}(| \langle i|\psi_i\rangle |^2 - 1)^2 + \gamma P(|\Psi_i\rangle) \tag{6}$$

with $\gamma = 100$ for the regularization parameter, $P(|\Psi_i\rangle) = \Sigma_i(| \langle \psi_{x_i}| \Pi_{\mathcal{H}} |\psi_{x_i}\rangle |^2 - 1)^2$ for the penalty function. Additionally, the displacements in the input phase space can be constrained to a circle of radius $|\alpha| = 1.5$ to make sure the encoding is as compact as possible.

In the following two sub-sections, we propose two training models for implementing the quantum-classical autoencoder to encode quantum states.

2.3 The Split Training Model

The first model we propose is the 'Split Training Model' or STM for short. In STM, the classical encoder is trained using a classical decoder first, establishing the weights of the classical encoder in such a way to create a latent space that can fully represent the input data. Then, the classical encoder is separated from the classical decoder and connected to the quantum decoder. Finally, the gate variables $(\theta, \phi, r, \alpha, \lambda)$ of the quantum layers are trained using the latent space created by the already-trained classical encoder. This operation allows the classical encoder to produce a latent space that can best describe the input data, that is then used to displace the vacuum states in the input layer of the quantum layers. An illustration of this method is shown below.

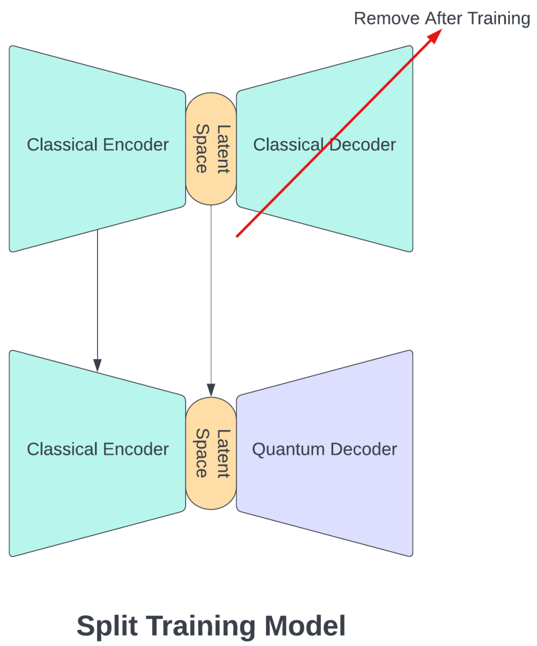

Figure 8: Illustration of the proposed Split Training Model.

This model can be considered as the conventional way of training and using the hybrid autoencoder for its use in encoding problems.

2.4 The Self Consistent Training Model

The second model we propose is the 'Self Consistent Training Model' or SCTM for short. In SCTM, the classical encoder and the quantum decoder is trained together, forcing the statevector output of the quantum decoder to be reused as an input for the classical encoder. This unconventional type of training allows the further experimentation with the limits of the quantum-classical autoencoder which can improve our understanding of the system and its capabilities.

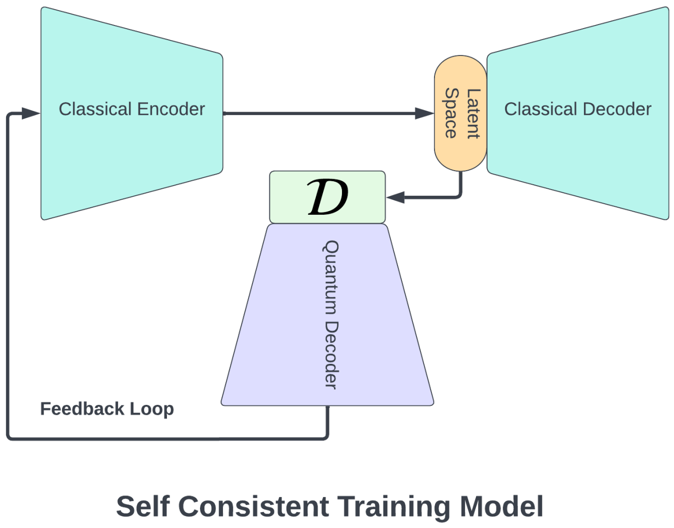

Figure 9: Illustration of the proposed Self Consistent Training Model.

The unconventional structure of SCTM will be used to compare its results with STM, highlighting the parameters and decisions that are more impactful to the performance of the hybrid autoencoder.

The SCTM produced some interesting results that may allow us to use it for experimental applications and they will be discussed at the end of this report.

3 RESULTS AND DISCUSSION

3.1 Increasing The Number Of Fock States To Be Encoded

The First Three Fock States

The training using the Split Training Model required 25 quantum layers in the quantum decoder, a learning rate of 0.05, 13 hidden layers in the classical encoder and 150 epochs with the Adam optimizer.

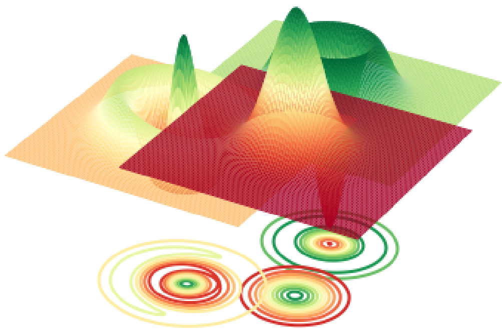

Figure 10: Continuous Encoding Of The First Three Fock States with STM

The training using the Self Consistent Training Model required 25 quantum layers in the quantum decoder, a learning rate of 0.05, 13 hidden layers in the classical encoder and 150 epochs with the Adam optimizer.

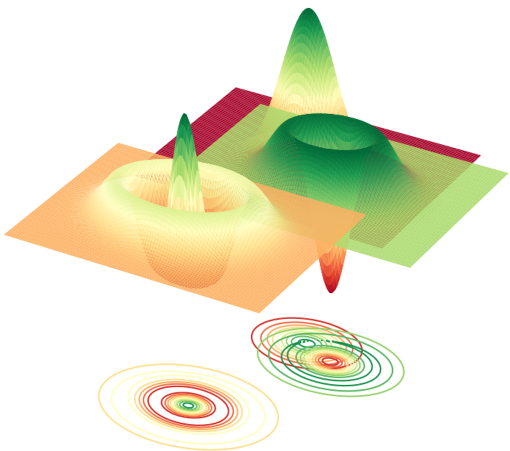

Figure 11: Continuous Encoding Of The First Three Fock States with SCTM

The First Four Fock States

The training using the Split Training Model required 80 quantum layers in the quantum decoder, a learning rate of 0.01, 13 hidden layers in the classical encoder and 200 epochs with the Adam optimizer.

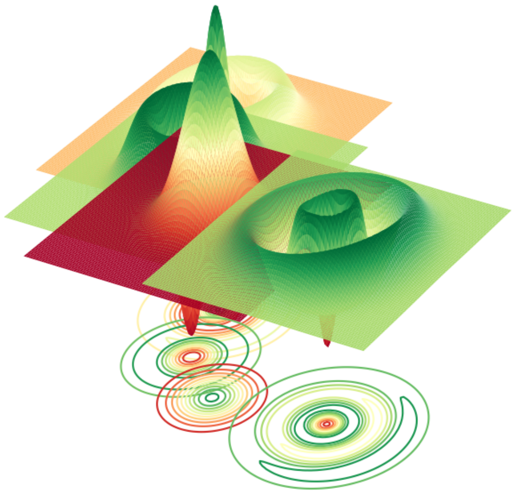

Figure 12: Continuous Encoding Of The First Four Fock States with STM

The training using the Self Consistent Training Model required 25 quantum layers in the quantum decoder, a learning rate of 0.05, 13 hidden layers in the classical encoder and 150 epochs with the Adam optimizer.

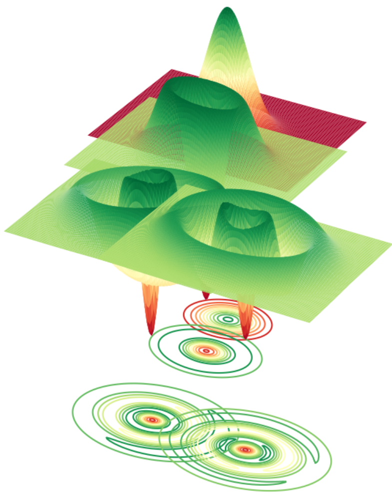

Figure 13: Continuous Encoding Of The First Four Fock States with SCTM

The First Five Fock States

The training using the Split Training Model required 150 quantum layers in the quantum decoder, a learning rate of 0.01, 13 hidden layers in the classical encoder and 400 epochs with the Adam optimizer.

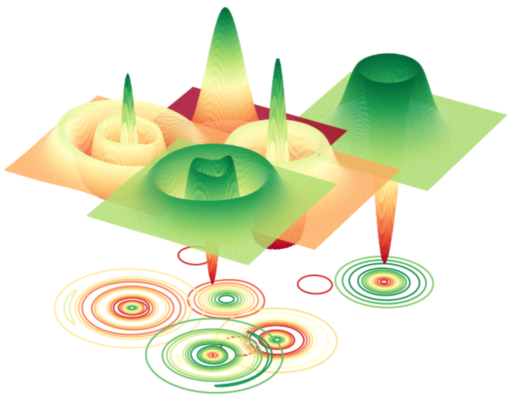

Figure 14: Continuous Encoding Of The First Five Fock States with STM

The training using the Self Consistent Training Model required 25 quantum layers in the quantum decoder, a learning rate of 0.05, 13 hidden layers in the classical encoder and 150 epochs with the Adam optimizer.

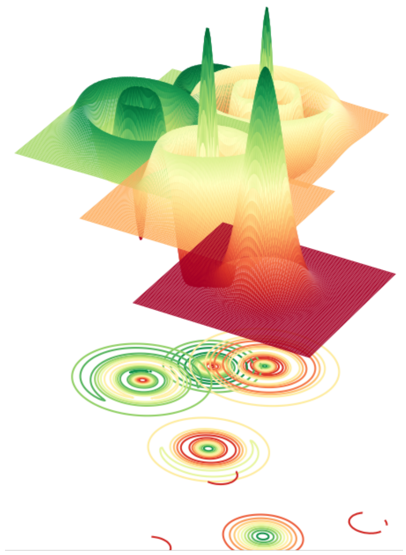

Figure 15: Continuous Encoding Of The First Five Fock States with SCTM

3.2 Optimizer Selection

We have experimented with four different optimizers to see how they effected the training results keeping the number of classical and quantum layers, learning rate and number of epochs. The training fidelity results are presented below:

To quickly go over the most well known ones of the used optimizers, SGD or Stochastic Gradient Descent is a spacific case of the vanilla Gradient Descent algorithm which subtracts the gradient multiplied by the learning rate from the weights and Adam is one of, if not the most popular optimizer in the machine learning circles, combining the individual weights from the RMSProp optimizer and the weighted average of momentum.

- Training fidelity results where the Adam optimizer was used.

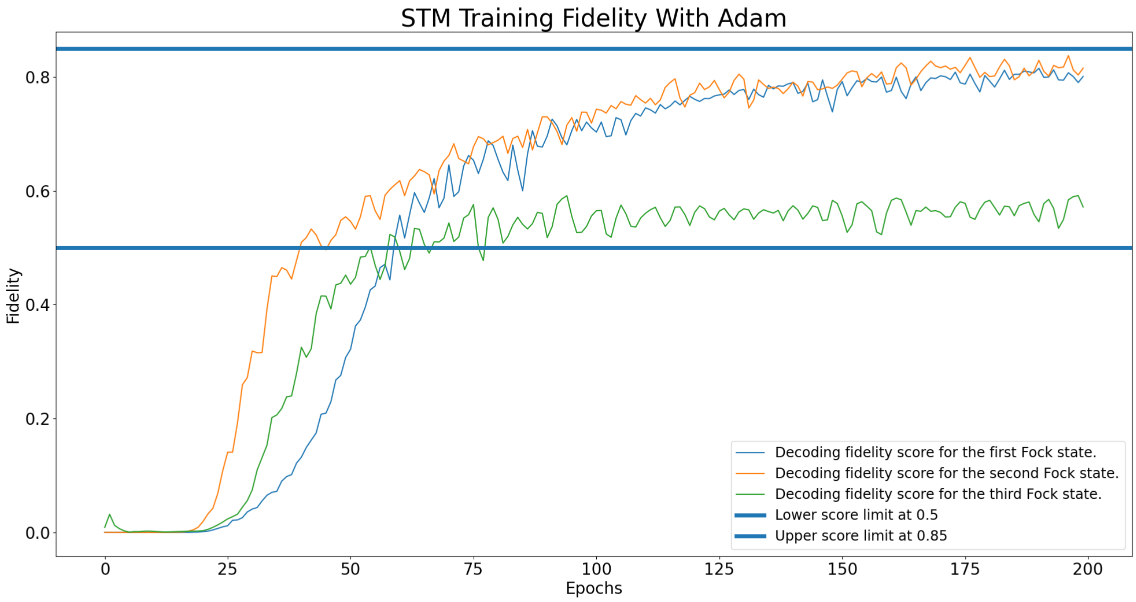

Figure 16: Training Fidelity Of The First Three Fock States with Adam

- Training fidelity results where the NAdam optimizer was used.

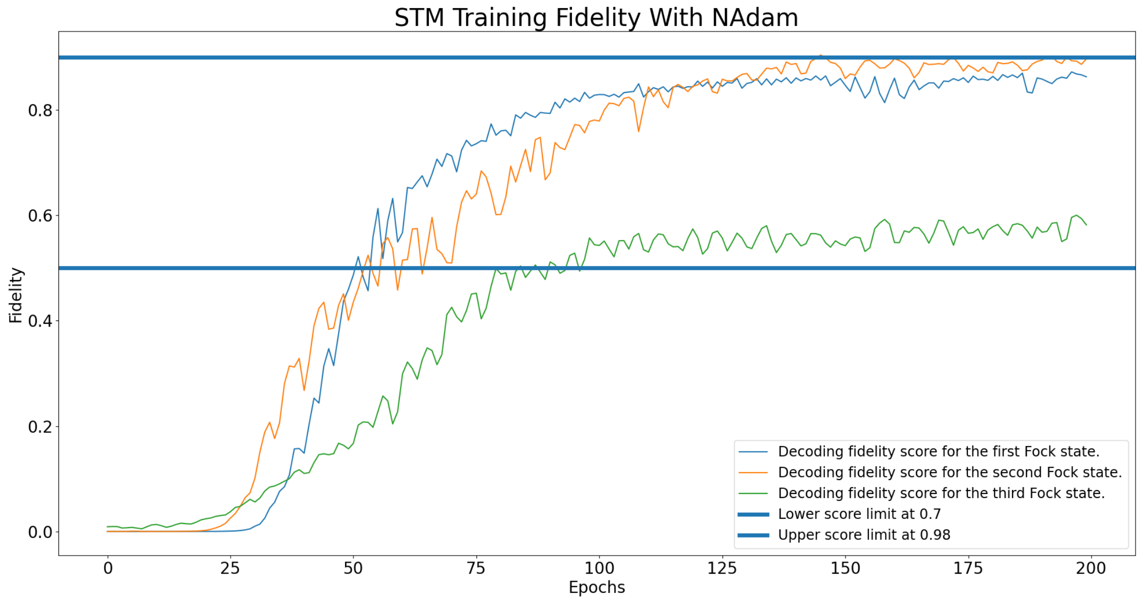

Figure 17: Training Fidelity Of The First Three Fock States with NAdam

- Training fidelity results where the RMSProp optimizer was used.

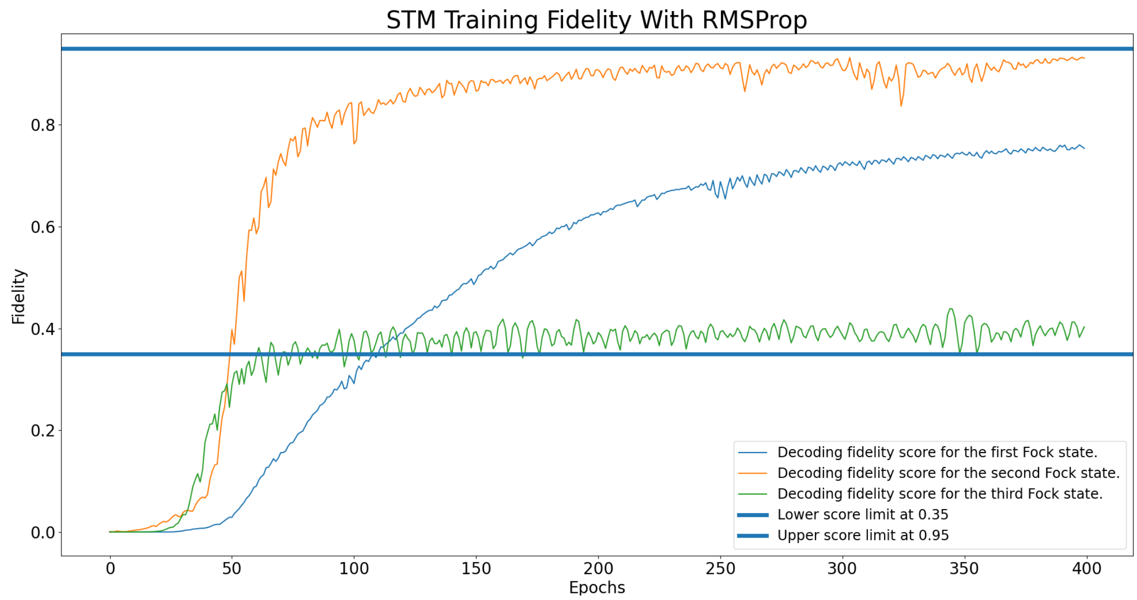

Figure 18: Training Fidelity Of The First Three Fock States with RMSProp

- Training fidelity results where the SGD optimizer was used.

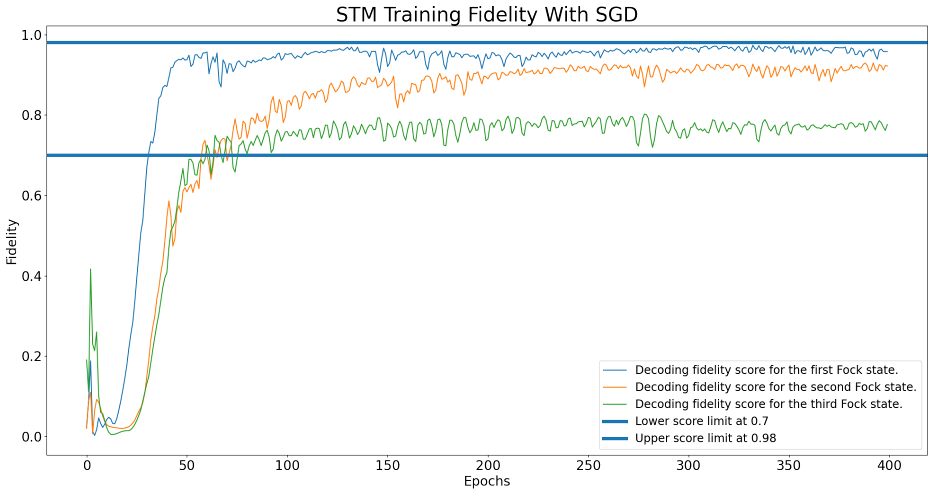

Figure 19: Training Fidelity Of The First Three Fock States with SGD

It is clear that the Stochastic Gradient Descent optimizer yielded better results for the hybrid autoencoder structure.

3.3 Nonlinear Gate Selection

We have experimented with two different nonlinear gates, Kerr gate and cubic phase gate that were mentioned in Section 2. Keeping the number of classical and quantum layers, learning rate and number of epochs constant; the results are presented below.

- Results using the Cubic phase gate as the nonlinear gate in the quantum layer.

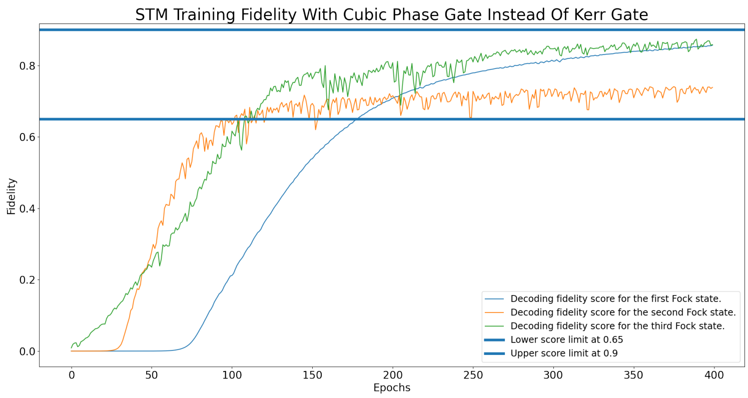

Figure 20: Training Fidelity Of STM Using The Cubic phase Gate For Nonlinearity

- Results using the Kerr gate as the nonlinear gate in the quantum layer.

Figure 21: Training Fidelity Of STM Using The Kerr Gate For Nonlinearity

4 CONCLUSION

To sum up, we have used a quantum-classical autoencoder structure that employs displacement encoding between the classical and quantum layers with two different training models to find a reliable and successful method to perform quantum state encoding; tuning training parameters, architectural parameters and comparing the two different training methods in the process.

For starters, it should be noted that encoding a vector of size three, four or five to a vector of size two is a very easy and trivial task for a powerful machine learning framework like Tensorflow. Because of that, the training accuracy of the classical autoencoder always reaches 100% immediately every time without exception. Unfortunately however, this doesn't mean that the encoding is even close to flawless every time. The reason for this is the fact that we are training the autoencoder one pure state vector representation at a time. This means that the entire input space is not available to the cost function of the classical autoencoder. Also, looking into the latent space revealed that increasing the number of hidden layers of the classical encoder allowed it to span the latent space much better, performing a more balanced encoding. Therefore to perform better encoding, we can increase the number of hidden layers which probably won't increase the computational overhead too much anyway since the input data size is too small and we can use the whole inputs space to train the classical autoencoder and then perform individual state training for the quantum decoder.

The encoding results presented in Section 3.1 clearly show that the quantum-classical autoencoder used in this work is capable of encoding the first five Fock states $\{|0\rangle, |1\rangle, |2\rangle, |3\rangle, |4\rangle\}$ into the optical phase space with varying success. The encoding is carried out by a polar coordinate transformation in the optical phase space that is decided by the output of the classical encoder, which is $[a, \phi]$ where a is the length of the transformation and ϕ is the angle of the transformation. Because of that, the quantum layers have no effect on the encoding process, that is, unless The Self Consistent Training Model is used which introduces a feedback loop into the model, influencing the judgement of the classical encoder. This may be the reason why the encoding results of the SCTM look considerably worse than the encoding results of the Split Training Model. Increasing the size of the input space puts a strain on the hybrid autoencoder model, producing worse results directly correlated to the size of the input space. Even though STM produced better encoding overall, it required more and more layers as the input space size increased. However, even though SCTM produced worse encoding results, it could perform the encoding every single time without failing. Thus we can come to the conclusion that, SCTM offers more reliable encoding while producing inferior results to those of STM. This result may lead us to think of a new method that combines the STM and SCTM methods to get both accurate and reliable results.

Among the four different optimizers (Adam, NAdam, RMSProp and SGD) investigated, the Stochastic Gradient Descent optimizer performed the best, having the highest training fidelity scores during the training of the quantum layers. Among the two different nonlinear gates investigated, the cubic phase gate seemed to perform better than Kerr gate but this parameter needs more investigation to come to a solid conclusion. It would be appropriate to note here that using the SGD optimizer with the cubic phase gate as the nonlinear gate produced the best fidelity results during the training of the quantum layers.

The main point we should focus on while discussing the results of this project should be the fact that the SCTM was able to perform successful quantum state encoding without increasing the number of quantum layers. So far, we have only worked with simulators up to this point, but since we want to implement these methods to real physical hardware as well, we need to worry about the number of quantum layers that need to be executed. Using real quantum computers, increasing the number of quantum layers would increase decoherence and thus errors since increasing the number of quantum layers means that the number of gates that need to be executed has increased.

In the near future, we plan on continuing the work on this project to implement the proposed training methods to real quantum computers. This means that we need to find a way to eliminate the need for a statevector or a wavefunction to calculate the cost, since the cost function we used in this project requires a statevector or a wavefunction so it can only be used with simulators. Going through the simulator path, there are a lot more parameters to explore in this hybrid autoencoder architecture which can yield interesting results. The simulator method shouldn't be underestimated since it allows us to study these systems easily and teaches us a lot about hybrid networks.

References

[1] Sulaiman Alvi, Christian Bauer, and Benjamin Nachman. Quantum anomaly detection for collider physics. 2022.

[2] Samuel L. Braunstein and Peter van Loock. Quantum information with continuous variables. *Reviews of Modern Physics*, 77(2):513–577, jun 2005.

[3] Steven Brunton Craig Gin, Daniel Shea and Nathan Kutz. Deepgreen: deep learning of green's functions for nonlinear boundary value problems. 118(8), nov 2021.

[4] Nathan Killoran, Thomas R. Bromley, Juan Miguel Arrazola, Maria Schuld, Nicolá s Quesada, and Seth Lloyd. Continuous-variable quantum neural networks. *Physical Review Research*, 1(3), oct 2019.

[5] Korbinian Kottmann, Friederike Metz, Joana Fraxanet, and Niccolò Baldelli. Variational quantum anomaly detection: Unsupervised mapping of phase diagrams on a physical quantum computer. *Phys. Rev. Research*, 3:043184, Dec 2021.

[6] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien. Quantum computers. *Nature*, 464(7285):45–53, mar 2010.

[7] Hoi-Kwan Lau and Martin B. Plenio. Universal quantum computing with arbitrary continuous-variable encoding. *Physical Review Letters*, 117(10), aug 2016.

[8] Hoi-Kwan Lau, Raphael Pooser, George Siopsis, and Christian Weedbrook. Quantum machine learning over infinite dimensions. *Physical Review Letters*, 118(8), feb 2017.

[9] Bengio Y. Hinton LeCun, Y. Deep learning. *Nature*, 521, feb 2015.

[10] Mitchell A. Nahmias, Hsuan-Tung Peng, Thomas Ferreira de Lima, Chaoran Huang, Alexander N. Tait, Bhavin J. Shastri, and Paul R. Prucnal. A laser spiking neuron in a photonic integrated circuit, 2020.

[11] Jonathan Romero, Jonathan P Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, aug 2017.

[12] D.E. RUMELHART, G.E. HINTON, and R.J. WILLIAMS. Learning internal representations by error propagation. pages 399–421, 1988.