Continuous Variable Quantum Neural Networks

Brief Notes and Stuff

Sangeet. S

November 3, 2023

1 Basics of MLPs

A *multi-layer perceptron* (MLP) is a type of *feed-forward* neural network (NN). It contains *fully connected* layers in which every input node is connected to every output node. Every layer performs an affine transformation on the data followed by a non-linear *activation* function.

Given an input $\mathbf{x} \in \mathbb{R}^n$ for which the MLP outputs $\mathbf{y} \in \mathbb{R}^m$, a layer $\mathcal{L}i : \mathbb{R}^{n_i} \to \mathbb{R}^{m_i}$ is given by,

$$\mathbf{y} = f_{\boldsymbol{\theta}}(\mathbf{x}) = \mathcal{L}N \circ \mathcal{L}N - 1 \circ \cdots \circ \mathcal{L}1(\mathbf{x}), \tag{1}$$

$$\mathcal{L}i(\mathbf{x}) := \varphi\left(W_i(\theta_{W_i})\mathbf{x} + \mathbf{b}(\theta_{\mathbf{b}_i})\right), \tag{2}$$

where, $W_i \in \mathbb{M}_{m\times n}(\mathbb{R})$ and $\mathbf{b}_i \in \mathbb{R}^m$ are the weight matrix and bias vector associated to the layer $\mathcal{L}i$ $(\forall i : n_{i+1} = m_i)$, which are parameterised by θ_{W_i} and $\theta_{\mathbf{b}_i}$ respectively.

The ultimate goal is to tune the parameters $\boldsymbol{\theta}$ to get *accurate* results which we build towards by *training* our NN appropriately. Various universality theorems [1, 2, 4] guarantee that given *enough* free parameters, MLPs are capable of approximating *any continuous function on a closed and bounded subset of* $\mathbb{R}^n$ to arbitrary accuracy. Deeper networks have been shown to be more powerful and more efficient than shallower networks with the same number of parameters [5, 6, 7].

Unfortunately, these universality theorems do not say anything about how to actually find these approximations. The I/O relation of the function is inferred from the data. This is where the training comes into the picture. The training is typically done using a *stochastic gradient descent* algorithm that chooses the optimal model parameters $\boldsymbol{\theta}$ based on some preliminary *training data*, by estimating derivatives of these parameters and minimizing an appropriate *cost function*.

2 CVQC and Wigner Functions

A qumode state ρ can be represented by its so-called *Wigner transform* on a phase space. The Wigner transform $F(\mathbf{x}, \mathbf{p})$ of a state ρ is a *quasi-probability* distribution given by,

$$F(\mathbf{x}, \mathbf{p}) := \frac{1}{(2\pi)^n} \int_{\mathbb{R}^n} \mathrm{d}\mathbf{y}\, \mathrm{e}^{i\mathbf{p}\cdot\mathbf{y}} \left\langle \mathbf{x} - \frac{\mathbf{y}}{2} \middle| \rho \middle| \mathbf{x} + \frac{\mathbf{y}}{2} \right\rangle, \tag{3}$$

where, $(\mathbf{x}, \mathbf{p}) \in \Gamma \simeq \mathbb{R}^{2n}$. This can be extended to other linear operators $\hat{\mathcal{O}}$ on a Hilbert space to obtain their phase space representation, i.e,

$$\mathcal{O}(\mathbf{x}, \mathbf{p}) := \int_{\mathbb{R}^n} \mathrm{d}\mathbf{y}\, \mathrm{e}^{i\mathbf{p}\cdot\mathbf{y}} \left\langle \mathbf{x} - \frac{\mathbf{y}}{2} \middle| \hat{\mathcal{O}} \middle| \mathbf{x} + \frac{\mathbf{y}}{2} \right\rangle. \tag{4}$$

Given a complete ONB, such as an eigenbasis of the quadratures $\hat{\mathbf{x}}$ or $\hat{\mathbf{p}}$, we find that the expectation value of the linear operator $\hat{\mathcal{O}}$ with respect to a state ρ is given by,

$$\mathrm{Tr}\Big(\hat{\mathcal{O}}\rho\Big) = \int_{\Gamma} \mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{p}\,\mathcal{O}(\mathbf{x},\mathbf{p})F(\mathbf{x},\mathbf{p}). \tag{5}$$

In the context of quantum computation, a set of gates is universal if it can be used to build any unitary transformation through a polynomial-depth quantum circuit. In the CV case, a universal set is a set of *gaussian gates* - which correspond to affine maps on the phase space; with a single *non-gaussian gate*, which corresponds to a nonlinear map on the phase space. Gaussian transformations are so-named because they map the set of gaussian distributions in phase space to itself.

For an N-mode system, the most general gaussian transformation has the effect of a matrix $M \in \mathrm{Sp}_{2N}(\mathbb{R})$ on the phase space Γ such that,

$$\begin{pmatrix}\mathbf{x}\\ \mathbf{p}\end{pmatrix} = M\begin{pmatrix}\mathbf{x}\\ \mathbf{p}\end{pmatrix} + \begin{pmatrix}\boldsymbol{\alpha}_r\\ \boldsymbol{\alpha}_i\end{pmatrix}, \tag{6}$$

where, $\boldsymbol{\alpha} \in \mathbb{C}^N$, with $\boldsymbol{\alpha}_r = \mathrm{Re}\{\alpha\}$ and $\boldsymbol{\alpha}_i = \mathrm{Im}\{\alpha\}$. We can further (Euler) decompose M as,

$$M = \Theta_2\begin{pmatrix}\Sigma & 0\\ 0 & \Sigma^{-1}\end{pmatrix}\Theta_1, \tag{7}$$

where, $\Theta_{j\in[2]} \in \mathrm{O}_{2N}(\mathbb{R}) \cap \mathrm{Sp}_{2N}(\mathbb{R})$. Such a matrix can be written in the form,

$$\Theta_j = \begin{pmatrix}C & D\\ -D & C\end{pmatrix}, \quad \text{with,} \quad CD^T - DC^T = 0 \quad \text{and} \quad CC^T + DD^T = \mathbb{1}. \tag{8}$$

In particular, $C \in \mathrm{O}_N(\mathbb{R}) \implies C \oplus C \in \mathrm{O}_{2N}(\mathbb{R}) \cap \mathrm{Sp}_{2N}(\mathbb{R}) \simeq \mathrm{U}_N(\mathbb{C})$. Check out the 2-out-of-3 property for unitary groups for good measure. This isomorphism allows us to perform the transformations Θ_j via the unitary action of passive linear optics (how are they interferometers?).

A typical gaussian gate is the rotation gate which produces a transformation $R(\phi)$ given by,

$$R(\phi): \qquad \begin{pmatrix}x\\ p\end{pmatrix} \longmapsto \begin{pmatrix}\cos\phi & \sin\phi\\ -\sin\phi & \cos\phi\end{pmatrix}\begin{pmatrix}x\\ p\end{pmatrix}. \tag{9}$$

Together with a two-mode gaussian gate, the beam-splitter, which has the effect $BS(\theta) = R(-\theta) \oplus R(-\theta)$ on the phase space, we can generate the transformations Θ_j described by (8). Let $\mathcal{U}_j(\theta_j, \phi_j)$ be an N-mode unitary that has the effect of the transformation Θ_j on the phase space.

Using N local squeezing gates $S(r_j) = \mathrm{diag}\,(e^{-r_j}, e^{r_j})$, which are also gaussian, we can generate the matrix, $\mathrm{diag}\,\big(\Sigma, \Sigma^{-1}\big)$ on the phase space. With this, we are able to reproduce (7). To complete the picture given by (6), we need N local displacement gates $D(\alpha_j) := \exp\Big(\alpha_j \mathrm{a}^\dagger - \alpha_j^* \mathrm{a}\Big)$.

Common non-gaussian gates include the cubic phase gate $V(\gamma) := \exp\big(i\gamma\hat{x}^3/3\big)$ and the Kerr gate $K(\kappa) = \exp\big(i\kappa\hat{n}^2\big)$, where $\hat{n}$ is the number operator.

3 CV QNNs

The fully connected NN architecture can be used for designing variational circuits in the CV model. A fully connected quantum layer $\mathcal{L}$ is given by the gate sequence,

$$\mathcal{L} := \Phi \circ \mathcal{D} \circ \mathcal{U}_2 \circ \mathcal{S} \circ \mathcal{U}_1, \tag{10}$$

where, $\mathcal{U}_{i\in[2]}(\boldsymbol{\theta}, \boldsymbol{\phi})$ are N-mode interferometers, $\mathcal{S}(\mathbf{r}) := \bigotimes_{j\in[N]} S(r_j)$ and $\mathcal{D}(\boldsymbol{\alpha}) := \bigotimes_{j\in[N]} D(\alpha_j)$. Also, $\Phi \equiv \Phi(\boldsymbol{\lambda}) = \bigotimes_{j\in[N]} \varphi(\lambda_j)$, where, $\varphi(\lambda_j)$ is some local non-gaussian gate parametrised by λ_j.

The set of gate parameters $\{\boldsymbol{\theta}, \boldsymbol{\phi}, \mathbf{r}, \boldsymbol{\alpha}, \boldsymbol{\lambda}\}$ are the free parameters of the QNN. From the previous section, it is clear that the gaussian sub-layer $\mathcal{D} \circ \mathcal{U}_2 \circ \mathcal{S} \circ \mathcal{U}_1$ is sufficient to parameterise any unitary affine transformation on N qumodes (6) in the phase space picture.

Progressive layers in an MLP structure for a QNN can become smaller via measurements on qumodes or by tracing them out. The output of the NN is determined by a measurement on the final layer's output qumode(s). It would probably be a good idea to add a schematic to show how these things look. Different choices of measurements (homodyne, heterodyne, photon-count) may be better suited depending on the task.

3.1 Embedding a classical NN

QNNs reduce to the special case of classical MLPs when we don't create superposition and entanglement. The QNN architecture described above is able to accept classical inputs $\mathbf{x} \in \mathbb{R}^N$ by encoding it via $\mathcal{D}(\mathbf{x}) |0\rangle = |\mathbf{x}\rangle \equiv |x_1\rangle \otimes \cdots \otimes |x_N\rangle$ for the first layer. To obtain the output, we may perform an ideal homodyne measurement, which projects onto the eigenstates $|x_j\rangle$.

To embed the classical MLP, we must ensure that there isn't any mixing between the x and p variables, while reproducing (2) within this encoding. In other words, our layers must execute,

$$|\mathbf{x}\rangle \longmapsto |\varphi(W\mathbf{x} + \mathbf{b})\rangle \,. \tag{11}$$

To proceed, we can break the calculation into the following steps -

1. For simplicity, we assume W is full rank and perform an SVD on W to obtain $O_2 \Sigma O_1$.

2. We must restrict $\mathcal{U}_{i\in[2]}$ such that $\Theta_{i\in[2]}$ from (7) is block-diagonal. In other words, we must restrict $\mathcal{U}_i$ to the action of a phase-less beam-splitter $BS(\theta)$. For real entries, the unitarity of U_i reduces to orthogonality. Then for a real coherent state $|\mathbf{x}\rangle$, it can be shown (Appendix A [3]) that, $\mathcal{U}_i |\mathbf{x}\rangle = |C_i \mathbf{x}\rangle$, where, C_i is orthogonal. We choose $\mathcal{U}_i$ such that $C_i = O_i$.

3. A tensor product of local squeezing gates gives an element-wise scaling of the vector, i.e,

$$\mathcal{S}(\mathbf{r}) = \mathrm{e}^{\frac{1}{2}\sum_i r_i} |\Sigma \mathbf{x}\rangle \,, \tag{12}$$

   where, $\Sigma = \mathrm{diag}\left(\{e^{-r_i}\}\right)$. The prefactor won't matter cos the states are not normalisable.

4. A bias translation of $\mathbf{b} \in \mathbb{R}^N$ can be achieved by displacing the state, i.e,

$$\mathcal{D}(\mathbf{b}) |\mathbf{x}\rangle = |\mathbf{x} + \mathbf{b}\rangle \,. \tag{13}$$

5. By following the above steps we can successfully generate our affine transformation, i.e,

$$(\mathcal{D} \circ \mathcal{U}_2 \circ \mathcal{S} \circ \mathcal{U}_1) |\mathbf{x}\rangle = |(O_2 \Sigma O_1)\mathbf{x} + \mathbf{b}\rangle . \tag{14}$$

6. Now, we append an ancillary qumode j' in the vacuum state $|0\rangle_{j'}$ to the j^{th} qumode $|x\rangle_j$ which is the state $|x_j\rangle$. In other words, $|x\rangle_j \mapsto |x\rangle_j |0\rangle_{j'}$. Let us apply the two-mode unitary, $\mathcal{V}(\varphi) := \exp\big(i\varphi(\hat{x}_j) \otimes \hat{p}_{j'}\big)$, where $\varphi(\hat{x}_j)$ is a Taylor polynomial of $\hat{x}_j$ on the state,

$$\begin{aligned} \exp\big(i\varphi(\hat{x}_j) \otimes \hat{p}_{j'}\big) |x\rangle_j |0\rangle_{j'} &= \exp\big(i\varphi(x_j)\hat{p}_{j'}\big) |x\rangle_j |0\rangle_{j'} \\ &= D_{j'}(\varphi(x_j)) |x\rangle_j |0\rangle_{j'} = |x\rangle_j |\varphi(x_j)\rangle_{j'} \end{aligned} \tag{15}$$

   where we have used that D acts as a translation operator when coherent states are real. After swapping the modes and tracing out the ancilla we have, $|x_j\rangle \mapsto |\varphi(x_j)\rangle$. How would you decompose a SWAP gate in terms of the gaussian gates being used?

The above steps put together show that a single layer of a classical MLP is a special case of that of a fully connected QNN layer.

3.2 Other Architectures

A *convolutional neural network* (CNN) that is typically used for computer vision and image recognition problems may also be realised. Since the task of detecting an object is largely independent of where the object appears in an image, the network must exhibit translation equivariance. As a result, the weight matrix W is set to be a convolution. To implement this in a quantum model, we may consider a Hamiltonian $H(\hat{\mathbf{x}}, \hat{\mathbf{p}})$ that generates the gaussian unitary, $\exp(-iHt)$. Enforcing translation symmetry is then equivalent to setting H to be translation invariant. In phase space, such a gaussian is given by the symplectic matrix,

$$M = \begin{pmatrix} M_{\mathbf{xx}} & M_{\mathbf{xp}} \\ M_{\mathbf{px}} & M_{\mathbf{pp}} \end{pmatrix}, \tag{16}$$

where, for $\mathbf{u}, \mathbf{v} \in \{\mathbf{x}, \mathbf{p}\}$, we have $[M_{\mathbf{uv}}]_{ij} = [M_{\mathbf{uv}}]_{i+1,j+1}$. Derivation in Appendix B [3].

A *recurrent neural network* (RNN) is widely used for problems involving sequences such as time series analysis, natural language processing, etc. As the name suggests, an RNN takes two inputs at every iteration, $\mathbf{x}^{(t)}$ from a source/model and $\mathbf{h}^{(t)}$ which comes from the same network ata previous time step. An output $\mathbf{y}^{(t)}$ is (optionally) returned after the input is processed by the NN. RNNs have a time translation symmetry in the sense that the weights and biases remain the same in every layer. RNNs are well-suited for optical implementations as one can connect the output modes back to the input using optical fibres.

A *residual neural network* (resnet) uses a modified network topology via short-cut connections to add the output of a layer to its input. In other words, if a layer $\mathcal{L}$ were to execute a function f, then the resnet maps $\mathbf{x} \mapsto \mathbf{x} + f(\mathbf{x})$. In order to do this in with a QNN, we may use a two-mode CX gate after (15) to obtain the residual transformation,

$$|x\rangle |\varphi(x)\rangle \xmapsto{CX} |x\rangle |x + \varphi(x)\rangle . \tag{17}$$

Some schematics here would also be nice, but maybe a slightly more elaborate one as opposed to the one in [3].

4 Training QNNs

Given a noisy training dataset, we consider the simple case of training a QNN to reproduce the action of a function $f(x)$ on one-dimensional inputs x. Given an input $D(x)\,|0\rangle$ and output $|\psi_x\rangle$, the goal is to produce ψ_x such that $\langle\psi_x|\hat{x}|\psi_x\rangle = f(x)$. As such, we define our loss function L as,

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(f(x_i) - \langle\psi_{x_i}|\hat{x}_i|\psi_{x_i}\rangle\right)^2, \tag{18}$$

which is the mean squared error (MSE) between the outputs and desired outcomes. We use supervised learning (why is this *supervised* and what does that mean?) to train the NN, where the training and test data are tuples $(x_i, f(x_i))$ for x_i values chosen uniformly in some random interval. We can include noisy data by considering functions of the form $f(x) = f(x) + \Delta f$ where Δf is drawn from a normal distribution with zero mean and standard deviation ϵ.

4.1 Overfitting

In an ideal scenario, the QNN produces smooth outputs that do not overfit the data [3]. CV QNNs are adept at smoothing as a result of Hölder's inequality cited source forgets the absolute value on the LHS, i.e,

$$|\,\mathrm{Tr}\{(\rho-\sigma)\mathcal{O}\}| \leq \|\rho-\sigma\|_1 \|\mathcal{O}\|_\infty, \tag{19}$$

for any two states ρ, σ and an operator $\mathcal{O}$. In other words, the expectation values of an observable for states that are close to each other cannot differ significantly.

Why exactly is this an incorrect criterion for smoothing? What we require is that the outputs produce smooth fits for noisy encoded data. Is the inequality itself incorrect? But that can't be right? Reasons -

- The inequality follows from $|Tr(A^\dagger B)| \leq \|A\|_p \|B\|_q$ where $1/p + 1/q = 1$. Here we have $p = 1$ and $q = \infty$. Also, $A = \rho - \sigma$ and $B = X$.
- Now, $2 - 2\sqrt{F(\rho,\sigma)} \leq \|\rho-\sigma\|_1 = 2T(\rho,\sigma) \leq 2\sqrt{1-F(\rho,\sigma)}$, where F and T denote the fidelity and trace distance. Also, $\rho \to \sigma \implies F(\rho,\sigma) \to 1 \implies \|\rho-\sigma\|_1 \to 0$.
- States close to each other cannot be orthogonal since $F \to 0$ for such states.
- If this were not the case and similar states let to different results then we would be overfitting the data since our training doesn't generalise to all the similar data points.
- One thing to check would be if such similar data when encoded lead to quantum states that are close in the prescription described.

4.2 Device Imperfections

A significant source of errors in photonic quantum circuits is due to photon losses, which can be modelled using a lossy bosonic channel with a loss parameter η - where, $\eta = 0\%$ represents perfect transmission. The lossy channel acts at the end of each layer, ensuring that the effect of photon loss increases with circuit depth.

4.3 Penalties and Regularisation

For numerical simulations, each qumode must be truncated to a given cutoff dimension in the infinite-dimensional Hilbert space of Fock states. As a result, it is possible for the gate parameters to reach values such that the outputs have significant support outside of the truncated Hilbert space during training. In the simulation, this results in unnormalised output states and unreliable computations. To address this issue, we add a penalty to the loss function that penalizes unnormalised quantum states.

5 Power of QNNs

6 project

Goal is to get a feeling to which extend quantum mechanics actually matters for these proposed QCVNNs.
For example, by doing this:

- We take the simplest case put forward as evidence of the architecture working, namely the non-linear function fitting examples.
- First task is to re-produce Figure 5. This could be done using their source code, or a fresh re-implementation. (In any case, it would be nice if cut-off dimension were a variable).
- If using their implementation: Have to be able to read out intermediate quantum states.
- Now could use various ways of tracking quantumness. One would be to estimate negativity of Wigner functions throughout the evolution.
- Another, maybe more interesting approach: Just substitute the QCVNN by a non-linear classica CVNN, e.g. in the following way:
- All but the non-linear gate have direct classical interpretations. A fair comparison would be to sample a phase space point from the vacuum gaussian distribution in step one and track its evolution. For the Kerr gate, need to come up with a natural one-parameter analogue. E.g. rotation with angular velocity proportional to radius. Ideally, one would just swap out the quantum code with the clasiscal analogue in the same outer framework used by them. Then one can compare fitting accuracy.
- Outlook: One could also use a Monte-Carlo based framework. But maybe not as a first step.

7 Training for 1D Regression

7.1 QNN

Encoding layer performs $\mathcal{D}(x)\,|0\rangle$ for the entire batch of points. The Quantum Layers perform the required transformations and return a full batch of states from which a Decoding Layer extracts the batch of expectation values.

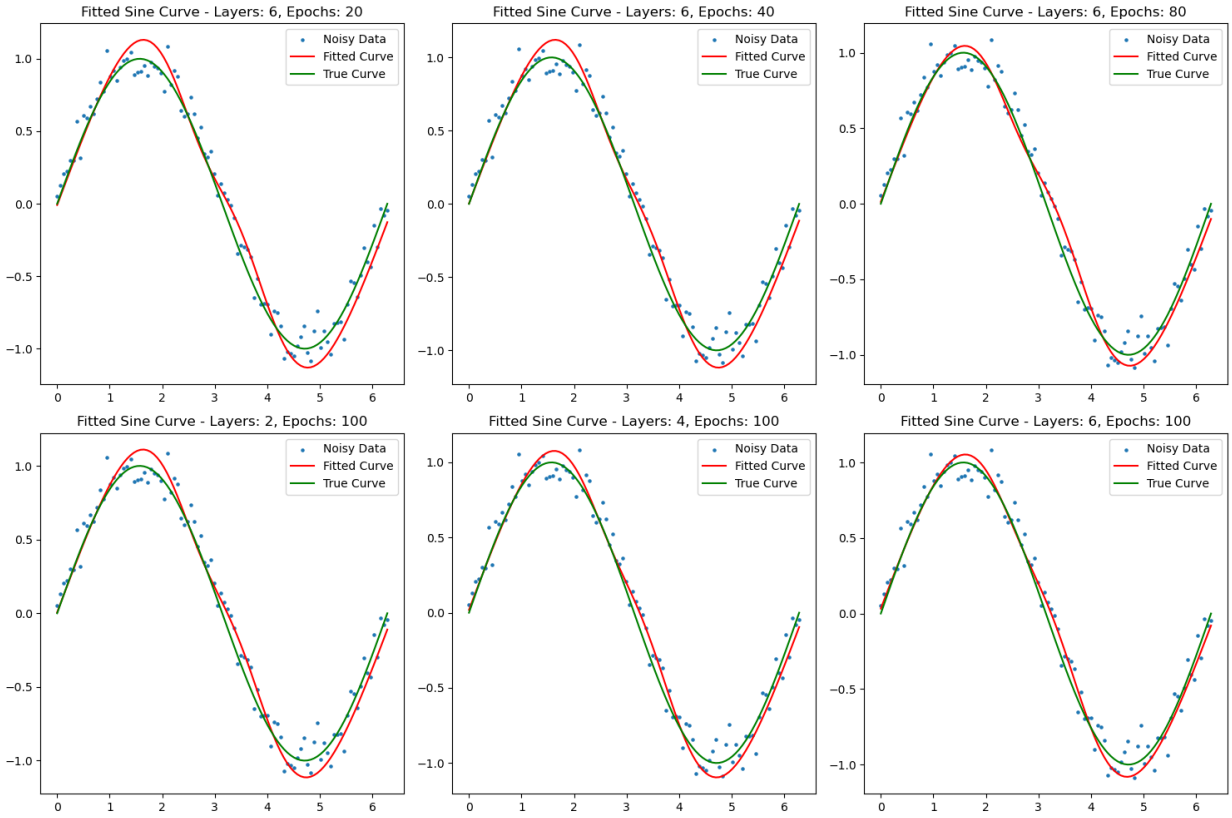

Figure 1: Fitting a Noisy Sinusoid (Noise level = 0.1)

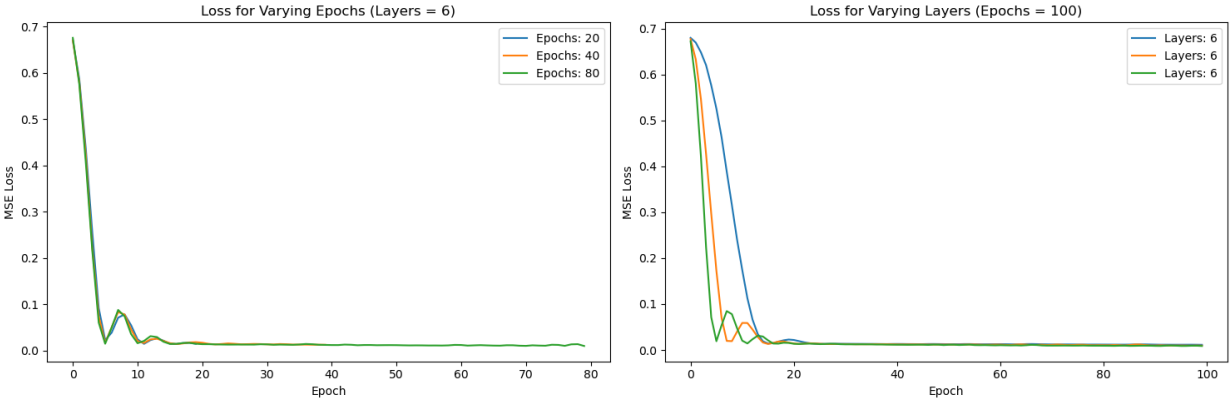

Figure 2: MSE Loss

7.2 Analogous Classical NN

Kerr-like transformation achieved by conditional rotation with angular velocity proportional to radius (value of the coordinate for 1D). The trainable parameter κ is the proportionality constant.

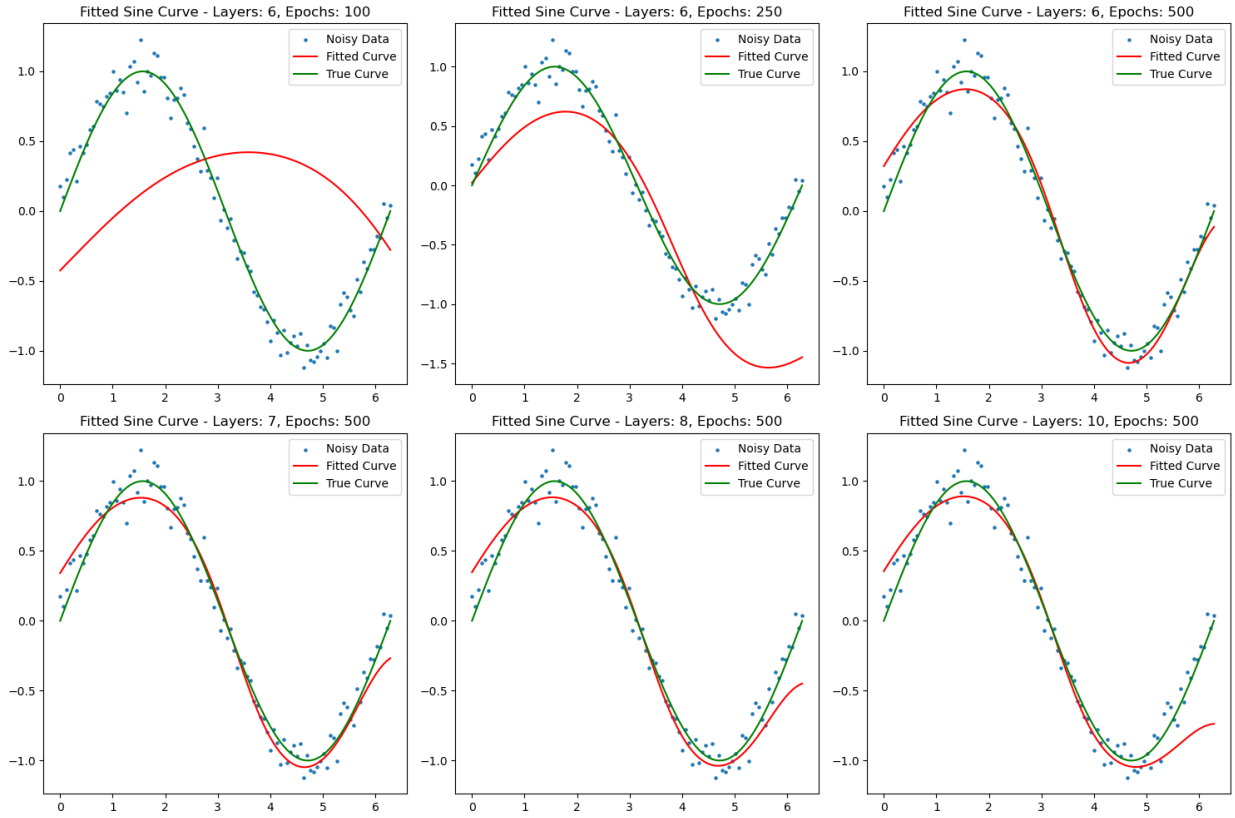

Figure 3: Fitting a Noisy Sinusoid (Noise level = 0.1)

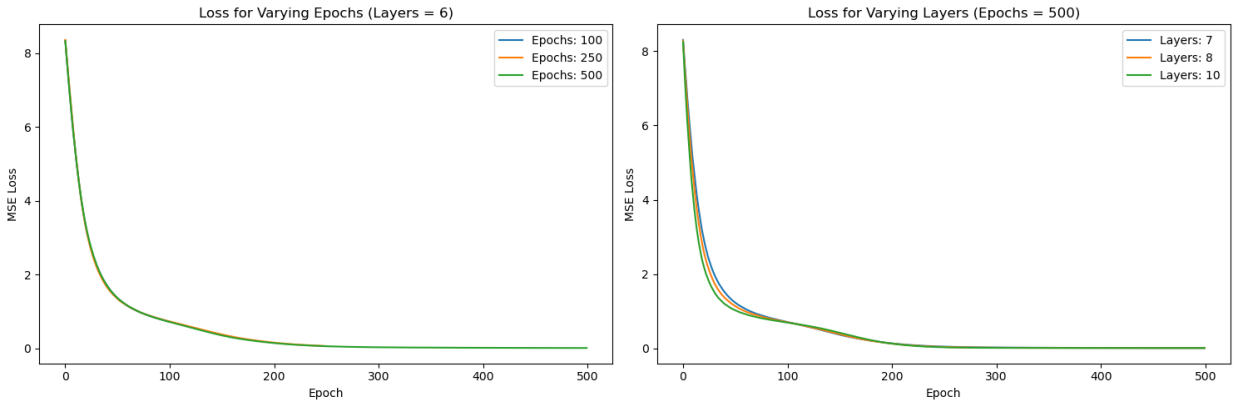

Figure 4: MSE Loss

7.3 Training and Validation MSE Comparision

Choosing the best models for both scenarios to compare.

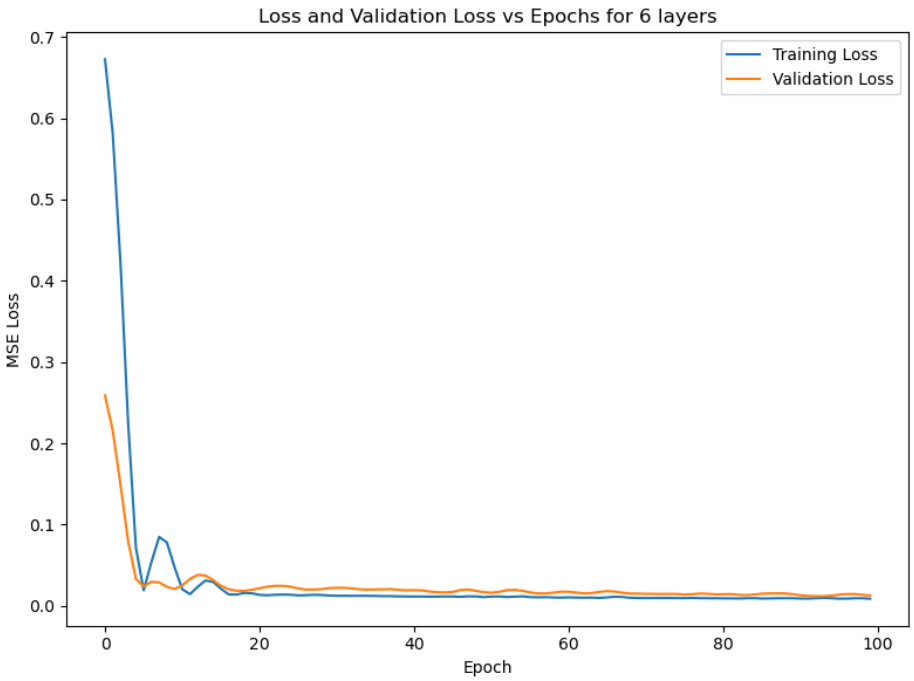

Figure 5: MSE Loss with 6 Layers and 100 epochs (QNN)

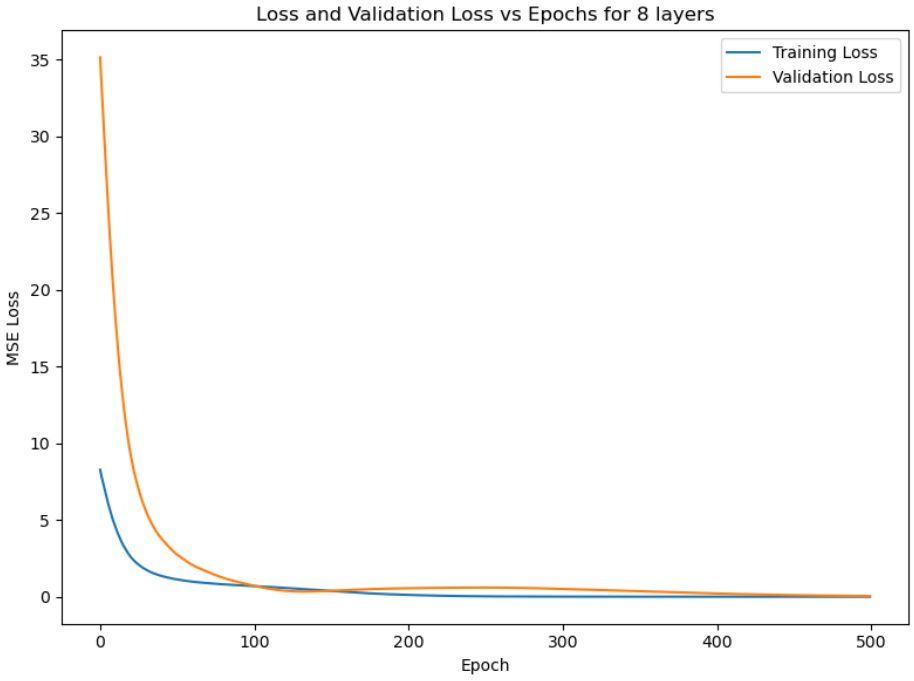

Figure 6: MSE Loss with 8 Layers and 500 epochs (classical NN)

Bibliography

[1] George V. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314, 1989.

[2] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[3] Nathan Killoran, Thomas R. Bromley, Juan Miguel Arrazola, Maria Schuld, Nicolás Quesada, and Seth Lloyd. Continuous-variable quantum neural networks. *Phys. Rev. Res.*, 1:033063, Oct 2019.

[4] Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

[5] Henry W. Lin and Max Tegmark. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168:1223–1247, 2016.

[6] W. Maass, G. Schnitger, and E. D. Sontag. *A Comparison of the Computational Power of Sigmoid and Boolean Threshold Circuits*, pages 127–151. Springer US, Boston, MA, 1994.

[7] Guido Montúfar. Universal approximation depth and errors of narrow belief networks with discrete units. *Neural Computation*, 26:1386–1407, 2013.