Quantum Neural Networks
Towards an era of Quantum-Assisted Machine Learning

Utkarsh, Shweta Sahoo & Harjinder Singh
Center for Computational Natural Sciences and Bioinformatics,
International Institute of Information Technology, Hyderabad
utkarsh.azad@research.iiit.ac.in

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Abstract
In this work, we investigate a hybrid quantum-classical (HQC) approach to envisage a quantum neural network using quantum parameterized circuits (QPCs). To simulate these quantum circuits, we make use of three different libraries based on the task at hand: Pennylane by Xanadu [3], pyQuil by Rigetti [4] and Qiskit by IBM [5]. Our analysis shows that given enough labelled training data points $\{x; f(x)\}$, a QPC with sufficient qubits and depth can be trained using classical optimization routines to perform both regression and classification tasks.

Introduction

Fault-tolerant universal quantum computers still appear to be more than a decade away. However, consistent growth in the field of quantum technologies over the years has lead to the development of Noisy Intermediate-Scale Quantum (NISQ) devices. The computational capabilities of these devices are considerably restricted due to limited connectivity, short coherence time, poor qubit quality and minimal error-correction. A particular class of useful algorithms that can be executed on these devices are variational algorithms which use the following hybrid approach: Prepare a highly entangled quantum state using a limited depth quantum processor, then apply a classical optimization routine on the gate parameters to converge to that quantum state for which the objective function is minimized.

Availability of massive amounts of data in the natural sciences has enabled the use of machine learning (ML) and artificial intelligence (AI) for tasks ranging from molecular discovery to prediction of new genes. Despite the excitement and promising results, the requirement of extensive computational resources to train ML models largely restricts their current applicability. Various proposals have been made to overcome this bottleneck, and one of them is to use quantum computers to achieve significant speedups. In the same spirit, we showcase a quantum-assisted approach in which a QPC is used as a machine learning model and optimized using a scheme similar to back-propagation in feed-forward neural networks. In general, such a circuit (model) can then be trained for a given labelled dataset $\{x, f(x)\}$ on current generation NISQ devices to perform classification and regression tasks by learning a function $f(x)$.

Methodology

1. Encode the classical information, i.e., given input data $\{x_i\}$, unitary input gate U_{x_i}, and teacher $f(x_i)$, we encode $\{x_i\}$ into some quantum state $|\psi_{in}(x_i)\rangle$ by applying U_{x_i} to the initialized qubits $|0\rangle^{\otimes N}$.
2. Mimick $f(\vec{x})$ using a θ-parameterized unitary to generate the following output state
$$|\psi_{out}(x_i, \theta)\rangle = U(\theta)\,|\psi_{in}(x_i)\rangle \tag{1}$$
3. Train the quantum circuit using expectation measures of the following observables: $\{B_j\} \in \{I, X, Y, Z\}^{\otimes N}$

Encoding

We can represent the state of any qubit k by:
$$\rho_k = \frac{I + \vec{m}.\vec{\sigma}}{2} \tag{2}$$
Here, $\vec{m}$ is the bloch vector and $\vec{\sigma}$ is the pauli vector. We encode our classical input data $\vec{x}$ in the components of $\vec{m}$ using standard rotation gates: $\{R_x(\theta_1), R_y(\theta_2), R_z(\theta_3)\}$ and then represent the state of an N-qubit register in the orthogonal pauli basis as:
$$\rho_{en}(\vec{x}) = \frac{1}{2^N} \sum_{i_1,i_2,\ldots,i_n} a_{i_1,i_2,\ldots,i_n}(\vec{x})\left[\sigma_{i_1} \otimes \sigma_{i_2} \otimes \ldots \otimes \sigma_{i_n}\right] \tag{3}$$
Here $i_1, i_2, \ldots, i_n \in \{0,1,2,3\}$, $\sigma_{i_j} \in \{I, X, Y, Z\}$ and the 4^N real coefficients $a_{i_1,i_2,\ldots,i_n}$ are characterized by some polynomial in the classical input data $\vec{x}$.

θ-Parameterized Unitary

After doing the state preparation of our classical input data $\vec{x}$, we need to find a unitary U that evolves the encoded state into the target state that corresponds to $f(\vec{x})$. Such a unitary can be realized by using the following two unitary blocks:

- U_{ent} – Entangler unitary consisting of a series of two-qubit C-NOT gates acting between each pair of qubits. We can measure the unitary's entangling capability by using the Meyer-Wallach entanglement measure [1]:
$$Ent = \frac{1}{N|S|} \sum_{\theta_i \in S} \sum_{k=1}^{N} 2(1 - Tr[\rho_k^2]) \tag{4}$$
where ρ_k is the one-qubit reduced density matrix, $S = \{\theta_i\}$ is the set of sampled circuit parameter vectors.
- $U_{\{\vec{\theta}\}}$ – θ-Parameterized unitary corresponding to the gate operation: $R_z(\theta)R_y(\phi)R_z(\lambda)$ acting on each qubit. We can measure its expressibility using the following measure [1]:
$$Expr = D_{KL}(\hat{P_{PQC}}(F;\theta)||P_{Haar}(F)) \tag{5}$$
where where $\hat{P}_{PQC}(F;\theta)$ is the estimated probability distribution of fidelities resulting from sampling states from a PQC.

The unitary $U = \{U_{ent}, U_{\{\vec{\theta}\}}\}$ can be thought of as the single layer of a neural network. Thus, applying multiple unitary blocks should be equivalent to using a multi-layer neural network.

Parameterized Expectation Measurements

Given the density matrix ρ, the expectation value of any Hermitian operator $\hat{M}$ is easily obtained by obtaining $Tr(\hat{M}\rho)$. In the pauli basis, any hermitian operator $\hat{M}$ can be expressed as $\sum_{i_1,i_2,\ldots,i_n} = b_{i_1,i_2,\ldots,i_n}[\sigma_{i_1,i_2,\ldots,i_n}]$. Hence, $\langle\hat{M}\rangle$ will be equal to $\sum_{i_1,i_2,\ldots,i_n} a_{i_1,i_2,\ldots,i_n} b_{i_1,i_2,\ldots,i_n}$.

When measurement of the k^{th} qubit is performed along the σ_l direction, the coefficients $a_{\ldots i_k \ldots}$ are set to zero for $\hat{\sigma}_{i_l} \perp \hat{\sigma}_j$. In our case, since the coefficient a_{i_j} describing the state ρ_{en} is characterized by some polynomial in classical input data $\vec{x}$, from (3) we can see that the same will happen for $\langle\hat{M}\rangle$. Therefore, it is sufficient to perform a single qubit measurement in the computational basis on qubit 0 to train our QPC. This output can be represented by some function F, and then for a chosen observable B_j we have the following output:
$$y_i = y(x_i, \theta) \equiv F(f\langle B_j(x_i, \theta)\rangle)) \tag{6}$$
We use this to tune our QPC parameters iteratively by minimizing the following loss function:
$$L_{f,y} = L_{(f(x_i), y(x_i, \theta))} \tag{7}$$

Results

We used our proposed model to learn functions $f(x)$ such as x^3, $\sin x$, and e^x. For doing this we used a QPC with number of qubits (N) and depth (D) to be 5. We made use of 25 training points for each of these functions with MSE loss, and the bloch vector used to encode them into the quantum circuit was:
$$\vec{m}(x) = x^3\hat{i} + x\sqrt{1 - x^4}\hat{j} + \sqrt{1 - x^2}\hat{k} \tag{8}$$
To apply gradient descent we calculated the gradient of the expectation value of an observable $\langle B(\theta)\rangle$ with respect to the circuit parameters $\vec{\theta}$ using the following relation [2]:
$$\frac{\partial\langle B\rangle}{\partial\theta_j} = \frac{1}{2}Tr[BU_{l:j+1}U_j(\frac{\pi}{2})\rho_j U_j^\dagger(\frac{\pi}{2})U_{l:j+1}^\dagger] - \frac{1}{2}Tr[BU_{l:j+1}U_j(\frac{-\pi}{2})\rho_j U_j^\dagger(\frac{-\pi}{2})U_{l:j+1}^\dagger] \tag{9}$$

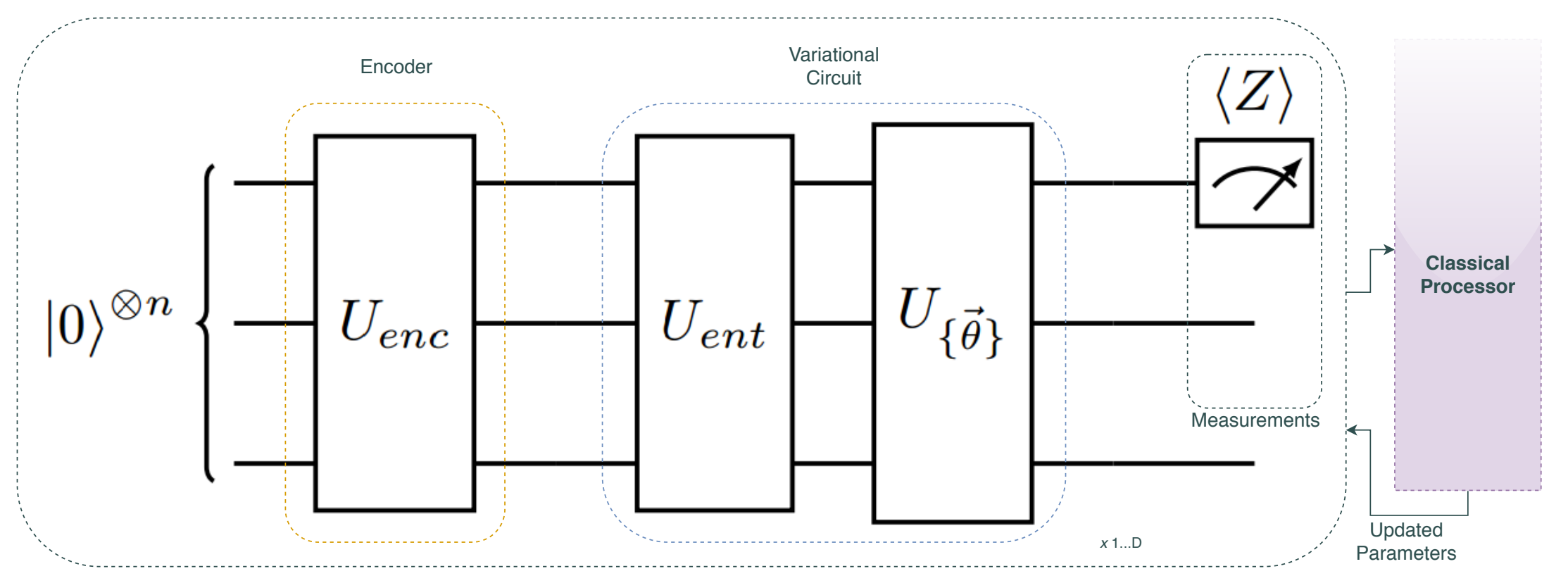

Figure 1: Overview of the quantum parameterized circuit

Number of circuit layers (L)	Expr (D_{KL})
1	0.24
2	0.053
3	0.020
4	0.007
5	0.005

Table 1: Expressibility values (or KL divergences) for the circuit with n=4 qubits. A lower value of *Expr* corresponds to better expressibility.

Number of circuit layers (L)	Ent $\langle Q\rangle$
1	0.43
2	0.62
3	0.70
4	0.75
5	0.80

Table 2: Entangling capability values for the circuit with n=4 qubits. Range: [0,1] where, Ent increases from 0 to 1.

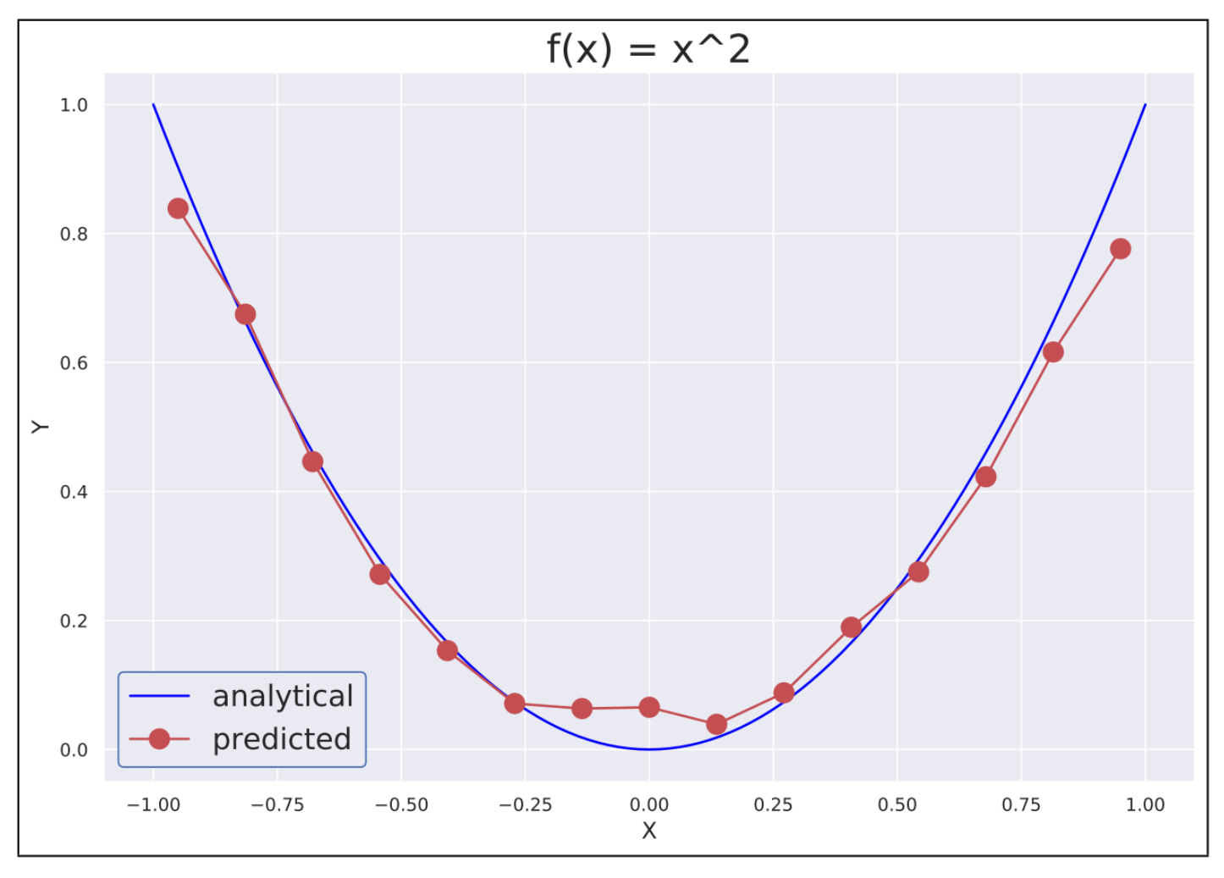

Figure 2: Plot for $f(x) = x^2$

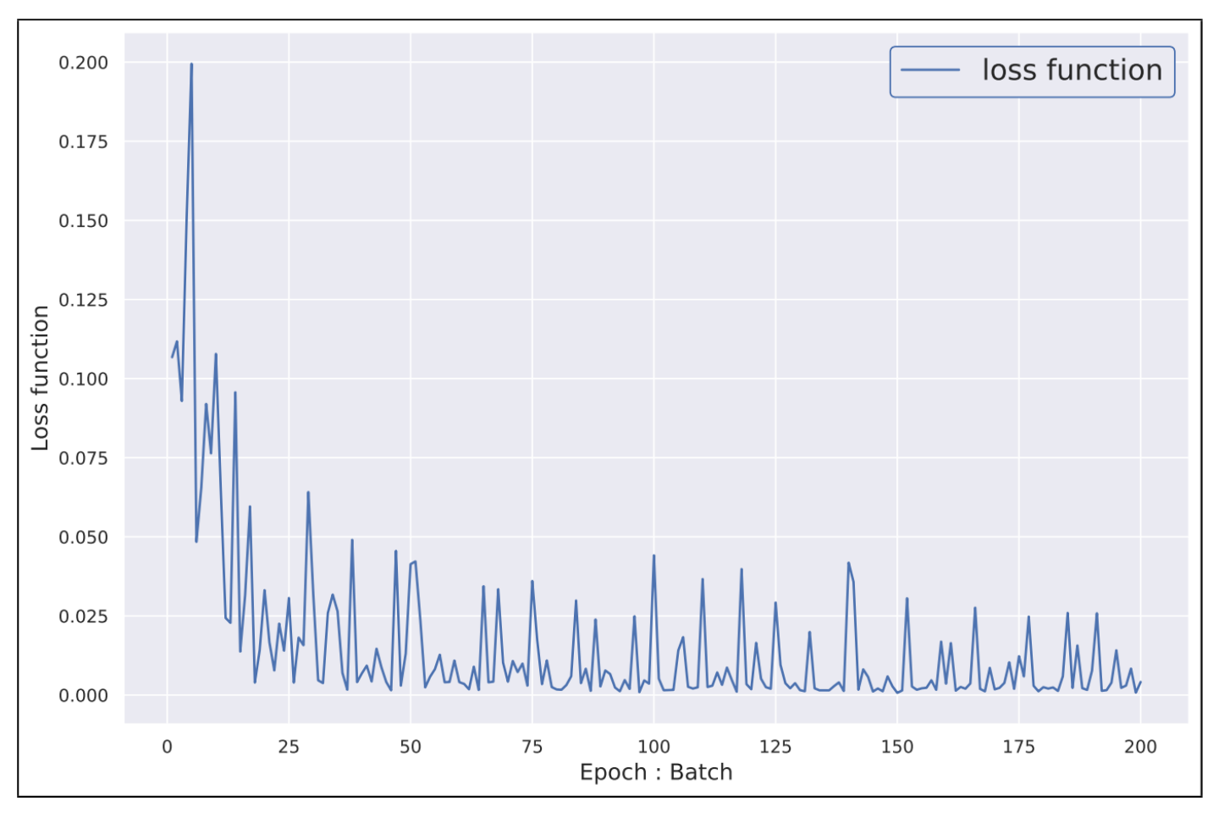

Figure 4: Loss function variation w.r.t. batch iterations for $f(x) = x^2$

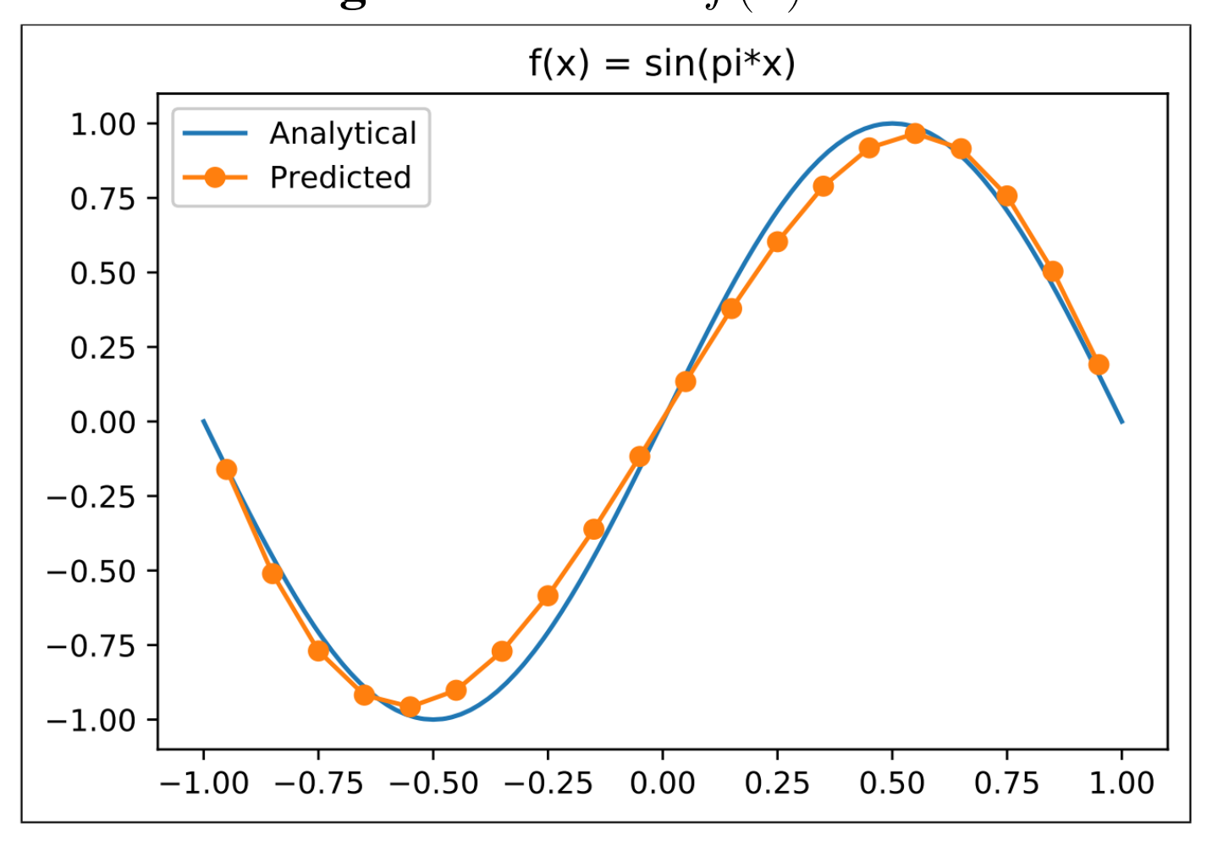

Figure 3: Plot for $f(x) = \sin \pi x$

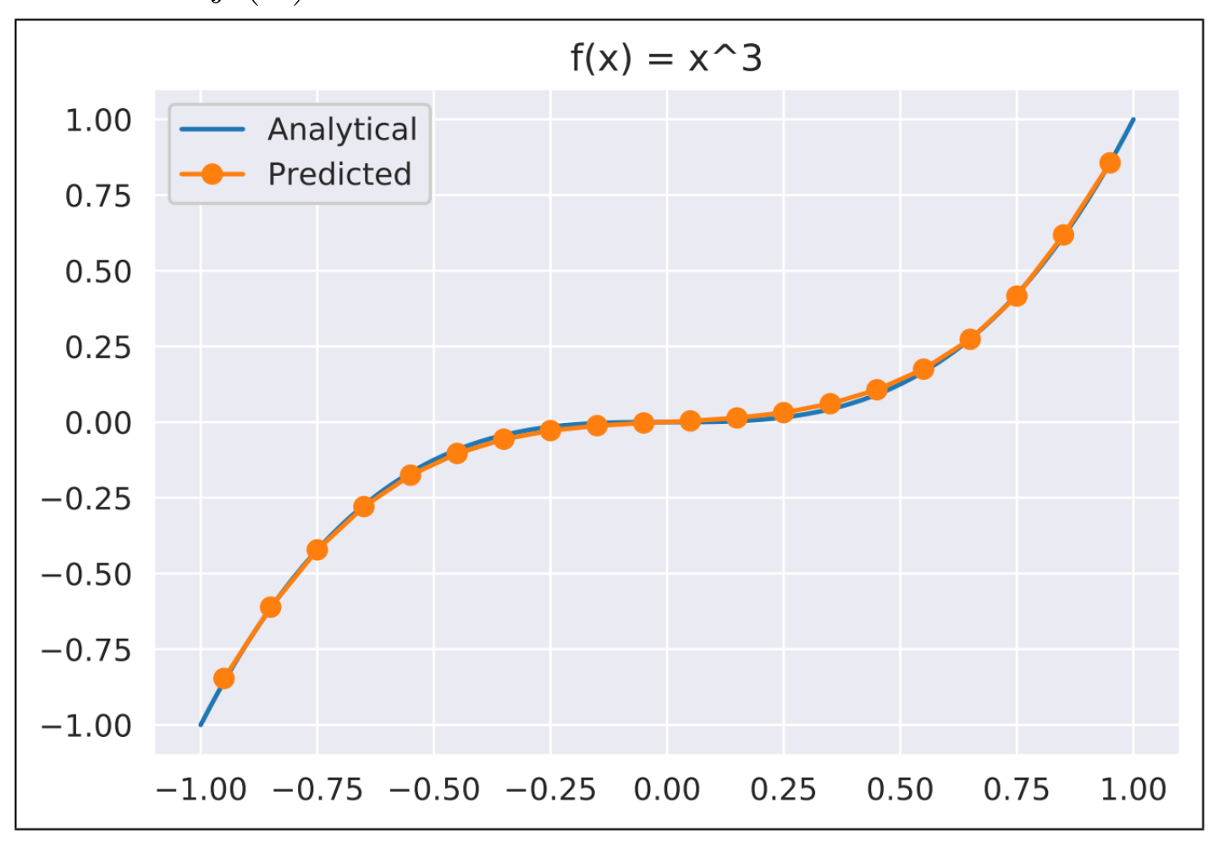

Figure 5: Plot for $f(x) = x^3$

Future Work

In an effort to improve hybrid quantum-classical algorithms like these, one needs to develop principled approaches to design parameterized quantum circuits. In this spirit, we aim to come up with one such approach that will aid in designing (and selecting) a problem-based ansatz in terms of accuracy, scalability, and implementability on near-term quantum devices.

References

1. Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms, Sukun Sim, Peter Johnson, Aspuru-Guzik, arXiv:1905.10876 (2019)
2. Quantum Circuit Learning, K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii Phys. Rev. A 98, 032309, (2018)
3. PennyLane: Automatic differentiation of hybrid quantum-classical computations, Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin, Carsten Blank, Keri McKiernan, and Nathan Killoran. arXiv:1811.04968 (2018)
4. A Practical Quantum Instruction Set Architecture, R. Smith, M. J. Curtis and W. J. Zeng, arXiv:1608.03355 (2016),
5. Qiskit: An Open-source Framework for Quantum Computing, 10.5281/zenodo.2562110 (2019)