A Hybrid Quantum-Classical Neural Network for Image Classification

Key Authors:

Arvind Ramachandra and Munish Singh

Abstract

Despite significant advancements in technology, modern machine learning applications continue to face computational challenges, as processing large amounts of data through classical computation remains time intensive. In the coming years, the need to process a large amount of data from various sources needs to be captured by multiple technologies such as the internet of things (IoT), video camera, etc.

Even the world's fastest computers require days or even months to train a classical AI algorithm to make self-driving decisions for a car that requires a set of computationally intensive calculations on new data additions, thus creating complex relations within the data variables.

Quantum machine learning applications can perform highly accurate calculations faster than classical computing systems due to high computing efficiency thus impacting insight generation through large databases or factoring in large numbers. After studying multiple quantum algorithms, we have developed and tested a 6-qubits quantum circuit to perform hybrid quantum-classical machine learning on the IBM-Aer quantum simulator using PennyLane for binary/multiclass classification.

We are utilizing 4000+ images of the tumor dataset and 1000+ images of the metal corrosion dataset for the development of machine learning models in the quantum simulator.

1. OVERVIEW OF QUANTUM MACHINE LEARNING

The machine learning algorithm extracts information and then offers predictions informed by the newly sourced data samples. In contrast with other mathematical techniques, these algorithms construct and update their predictive model based upon known data (the training dataset). Machine learning techniques can apply to different tasks, including: spam filtering, image processing, wide societal impact, image recognition, signal processing.

Myriad advancements in quantum information processing within recent years illustrate that some quantum algorithms can provide a speedup over their classical analogues. The application of quantum to the field of classical machine learning may produce similar results. Such a combination of quantum computing power and machine learning ideas would be a great boost to the quantum information science field and may introduce new practical solutions for current machine learning problems.

Quantum computing is an applied field of quantum mechanics combined with computer science, mathematics, and physics. The field harnesses the thesis of quantum mechanics to expand the horizon and solve the challenges regarding computational capabilities that we face in machine learning.

Quantum computing has an efficiency advantage in multi-dimensional systems and multi-variable statistical analysis. Whereas classical systems with many degrees of freedom are difficult to model because of the curse of dimensionality, the quantum parallelism effect allows us to avoid this problem. Quantum computational resources are the key to solving a variety of problems with expansive dimensions (machine learning, for example). Quantum computing relies on properties

of quantum mechanics to compute problems that would be out of reach for classical computers. A quantum computer uses qubits, which are like regular bits in a computer with the added ability to be placed into a superposition and share entanglements with one another.

Classical computers perform deterministic classical operations or can emulate probabilistic processes using sampling methods. By harnessing superposition and entanglement, quantum computers can perform quantum operations that are difficult to emulate at scale with classical computers. Ideas for leveraging NISQ quantum computing include optimization, quantum simulation, cryptography, and machine learning.

The idea of using quantum mechanics for computations originates from simulating such systems. Feynman (1982) noted that simulating quantum systems on classical computers becomes unfeasible as soon as the system size increases, whereas quantum particles would not suffer from similar constraints. Deutsch (1985) generalized the idea; he noted that quantum computers are universal Turing machines and that quantum parallelism implies that the performance of certain probabilistic tasks is faster by any classical means. In this study, we will develop and test a hybrid quantum-classical machine learning model on medical images and a random image dataset.

2. QUIBIT AND QUANTUM STATE

The basic element of quantum computing is the quantum bit (qubit), which has two basic states, $|0\rangle$ and $|1\rangle$. The notation "$|\rangle$" is called the Dirac notation, and we use it often, as it is the regular notation for states in quantum mechanics.

$|0\rangle = [1, 0]$

$|1\rangle = [1, 0]$

It is also possible to form linear combinations of states, often called superpositions. A qubit $|\psi\rangle$, can be seen as a generalization of the classical bit, that allows the superposition of $|0\rangle$ and $|1\rangle$ in a state of

$$|\psi\rangle=\alpha|0\rangle +\beta|1\rangle \quad \dots (1)$$

where α and β are complex coefficients. For example, a qubit can be in the state

$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \quad \dots (2)$$

which, when measured, gives the result 0 fifty percent ($|\frac{1}{\sqrt{2}}|^2$) of the time and the result 1 fifty percent of the time. We will return often to this state, which is sometimes denoted as $|+\rangle$.

For example: Let us put equation (2) in equation (1); therefore, we can say that $\alpha= \frac{1}{\sqrt{2}}$, $\beta= \frac{1}{\sqrt{2}}$

$$|\psi\rangle=\frac{1}{\sqrt{2}}[1,0] + \frac{1}{\sqrt{2}}[0,1]$$

$$|\psi\rangle= [\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$$

$|\psi\rangle= [\alpha, \beta]$.

The measurement of a qubit in a superposition state means that it will collapse to one of its basic states, although it is not possible to determine which one before measuring it. However, the $|\alpha|^2$ is the probability of having the result 0 for the

measurement or the $|\beta|^2$ is the probability of having the result 1 for the measurement. Therefore,

$|\alpha|^2 + |\beta|^2 = 1$, … (3)

since the addition of the probabilities will result in 1.

Equation (3) can be written as

$|\psi\rangle = e^{i\gamma}(\cos\theta/2\,|0\rangle + e^{i\psi}\sin\theta/2\,|1\rangle)$ … (4)

where θ, γ, and ψ are real numbers. Since $e^{i\gamma}$ in the front has no observable effect, we can rewrite the equation (4) as

$|\psi\rangle = \cos\theta/2\,|0\rangle + e^{i\psi}\sin\theta/2\,|1\rangle$ … (5)

The θ and ψ define a point on the unit's three-dimensional sphere, often referred to as a Bloch sphere.

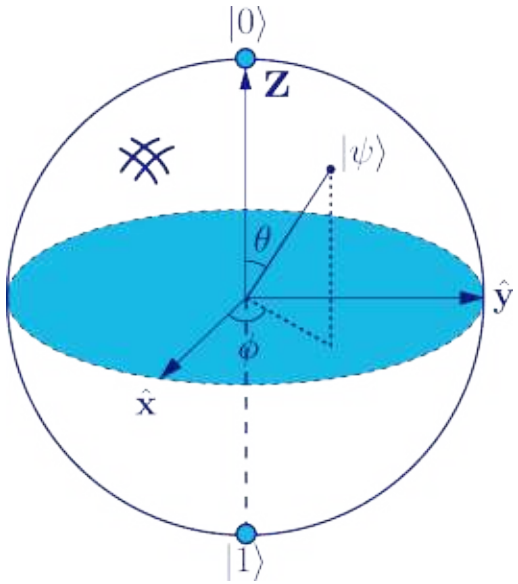


Figure 1. Bloch sphere representation of a qubit

Similarly, this can be represented for multiple qubits. For example, let us take a 2-qubit system that has four computational basis states, denoted as $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$.

$|\psi\rangle=\alpha 00|00\rangle + \alpha 01|01\rangle + \alpha 10|10\rangle + \alpha 11|11\rangle$ ... (4)

The Bell state, also referred to as an EPR pair, is an important two-qubit state, which can be represented in the following states:

a) $|\psi^{+}\rangle=|00\rangle + |11\rangle/ \sqrt{2}$

b) $|\psi^{-}\rangle=|00\rangle - |11\rangle/ \sqrt{2}$

c) $|\psi^{+}\rangle=|01\rangle + |10\rangle/ \sqrt{2}$

d) $|\psi^{-}\rangle=|01\rangle - |10\rangle/ \sqrt{2}$

If you want to implement the Bell state $|\psi^{+}\rangle=|01\rangle + |10\rangle/ \sqrt{2}$ in python using Qiskit, the circuit will look like this:

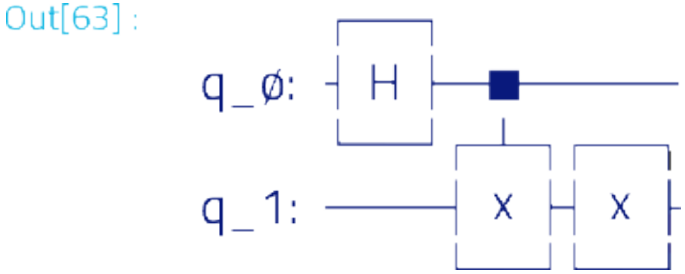


Figure 2. Bell state representation for $|\psi^{+}\rangle$

This creates the following state vector:

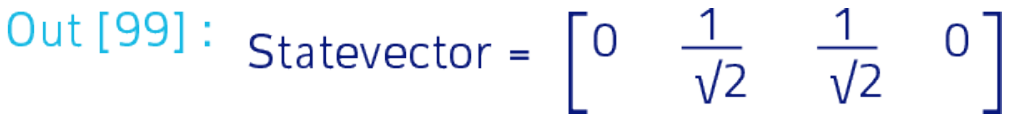


Figure 3. The output for the Bell state representation for $|\psi^{+}\rangle$

The operations with qubits are carried out by a unitary transformation "*U*". When "*U*" is applied to a superposition state, the result is another superposition state, obtained as the superposition of the corresponding basis vectors. This is an appealing characteristic of unitary transformations, which is called quantum parallelism, because it can be employed to evaluate the

different values of a function f(x) for a given input x at the same time, although this parallelism may not be immediately useful since the direct measurement on the output generally gives only f(x) for one value of x.

Let $|y\rangle$ be in the superposition state $|y\rangle = \alpha|0\rangle+\beta|1\rangle$; the unitary transformation U_y may be defined as:

$$U_y: |y,0\rangle \rightarrow |y, f(y)\rangle \quad (1)$$

where $|y,0\rangle$ is the joint state with the first qubit in $|y\rangle$ and the second qubit in $|0\rangle$, and $|y, f(y)\rangle$ stands for the corresponding joint output state.

Therefore:

$$U_y: |y,0\rangle \rightarrow \alpha|0, f(0)\rangle + \beta|1, f(1)\rangle \quad (2)$$

that contains simultaneous information for f(0) and f(1), i.e., two different values of f(x). This process is known as an oracle or quantum black box, can process quantum superposition states with an exponential speed-up compared to classical inputs. The idea can be extended to an n-qubit system:

$$|\theta\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes |\psi_3\rangle \otimes |\psi_4\rangle \otimes \ldots \otimes |\psi_n\rangle \quad (3)$$

where $\otimes$ is the tensor product. The system as shown in Eq. (3) can simultaneously process states, but only one of them could be accessible by means of direct measurement.

QC and ML have converged towards a new discipline, Quantum Machine Learning (QML); QML can bring together the concepts from both fields to build an enhanced solution by either improving ML algorithms, quantum experiments, or both.

3. QUANTUM GATES

To build the above equations in qubit and quantum states, we plan to use a quantum circuit model for computation. The quantum gate is the basic building block for the quantum circuit on which qubits operate and functions similarly to the classical logic gate for conventional digital circuits.

Unlike classical logic gates, quantum gates perform reversible computation as no information can ever be erased; the input can always be recovered from the output. We have explored two different techniques to provide reversible circuit-based models capable of universal computation, which are as follows:

- A computer developed totally of billiard balls and mirrors, provides a beautiful concrete realization of the principles of reversible computation, and;
- A technique based on the Toffoli gate (a reversible logic gate), which can perform any computation that a classical computer can.

Hence, the quantum gates are represented by unitary matrices, which show that quantum gates in the circuits always have the number of inputs = the number of outputs. Some of the basic gates are as follows:

- Pauli-X: $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \equiv \sigma_X$ ….. It maps $|0\rangle$ to $|1\rangle$ and $|1\rangle$ to $|0\rangle$. It is equivalent to a NOT gate.
- Pauli-Y: $Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \equiv \sigma_Y$ ….. It maps $|0\rangle$ to $i|1\rangle$ and $|1\rangle$ to $-i|0\rangle$.
- Pauli-Z: $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \equiv \sigma_Z$ ….. Also called a phase-flip, it leaves the basis state $|0\rangle$ unchanged and maps $|1\rangle$ to $-|1\rangle$.
- Hadamard: H= $\frac{1}{\sqrt{2}}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ ….. It creates a superposition by mapping $|0\rangle$ to $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|1\rangle$ to $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$.

- Phase shift: $S = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 0 \\ 0 & e^{i\psi} \end{pmatrix}$ ….. It leaves the basis state $|0\rangle$ unchanged and maps $|1\rangle$ to $e^{i\psi}|1\rangle$.
- Controlled NOT: $CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$ …. It acts on 2 qubits and performs the NOT operation on the target qubit only when the control qubit is $|1\rangle$.
- SWAP: $SWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ …. It acts on 2 qubits and swaps these two qubits.

4. ALGORITHMS

4.1 DEUTSCH-JOZSA

The Deutsch-Jozsa problem was the first quantum algorithm that performed better than any classical algorithm. A hidden Boolean function, f, takes a string of bits as the input and returns either 0 or 1:

$f(\{x_0, x_1, x_2,...\}) \rightarrow 0$ or 1 , where x_n is 0 or 1

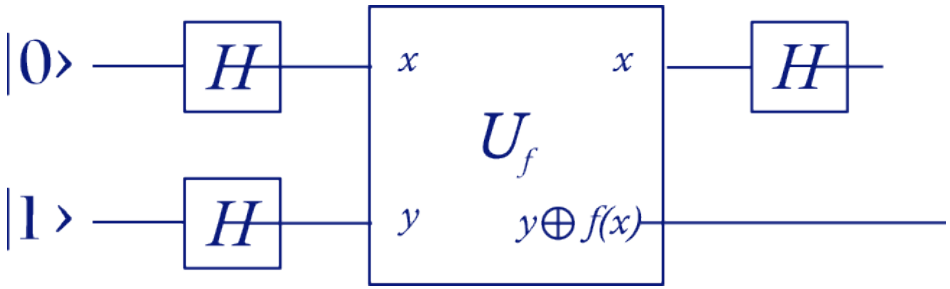


Figure 4. The circuit diagram for the Deutsch-Jozsa

The property of the given Boolean function is guaranteed to either be balanced or constant. A constant function returns all 0's or all 1's for any input, while a balanced function returns 0's for exactly half of all inputs and 1's for the other half. Our task is to determine whether the given function is balanced or constant.

Note that the Deutsch-Jozsa problem is an n-bit extension of the single-bit Deutsch problem.

Classical Solution:

For the case of deterministic classical algorithms, if *f* is balanced, then there are $2^{n/2}$ inputs x yielding f(x) = 0 and $2^{n/2}$ inputs x yielding f(x) = 1. Thus, in the worst case, an algorithm might be very unlucky and have its first $2^{n/2}$

queries return value f(x) = 0, only to have query $2^{n/2} + 1$. In this setting, the Deutsch-Jozsa algorithm yields an exponential improvement over classical algorithms, requiring just a single query to *f*.

Quantum Solution:

Inputs: (1) A black box U_f performs the transformation $|x\rangle|y\rangle \rightarrow |x\rangle|y \oplus f(x)\rangle$ for $x \in \{0, \ldots, 2n-1\}$ and $f(x) \in \{0, 1\}$. It is known that f(x) is either constant for all values of x or f(x) is balanced, that is, equal to 1 for exactly half of all the possible x and 0 for the other half.

Outputs: 0 if and only if f is constant.

Runtime: One evaluation of U_f. Always succeeds.

Procedure:

- $|0\rangle^{\otimes n}|1\rangle$ Initialize state.
- $\rightarrow \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2n-1}|x\rangle \left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$ Create superposition using Hadamard gates.
- $\rightarrow \frac{1}{\sqrt{2}}\sum_x(-1)^{f(x)}|x\rangle \left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$ Calculate function *f* using U_f.
- $\rightarrow \sum_z\sum_x\frac{(-1)^{x.z+f(x)}|z\rangle}{\sqrt{2^n}} \left[\frac{|0\rangle-|1\rangle}{\sqrt{2}}\right]$ Perform Hadamard transformation.
- $\rightarrow z$ Measure to obtain final output *z*.

4.2 BERNSTEIN-VAZIRANI (BV)

The Bernstein-Vazirani is an improvised version of the Deutsch-Jozsa algorithm. In the Deutsch-Jozsa algorithm, there is no scaling; the function of interest only acts on one bit. Therefore, we cannot scale the problem up to larger bits and see how the complexity scales.

The Bernstein-Vazirani (BV) algorithm presents us with a quantum algorithm whose complexity scales better than the best classical algorithms. Namely, the quantum algorithm will give us a linear speedup in the number of queries relative to the best classical algorithm. The complexity is in terms of the number of queries to an oracle, so the speedup is also in the terms of the number of queries.

Problem: Given an oracle access to f : {0, 1}n → {0, 1} and a promise that the function f(x) = s·x ($\sum_{i=1}^{n}$ si · xi) in F_2^n or f(x)=s·x (mod 2), where the algorithm is trying to find *s*, which is a secret string.

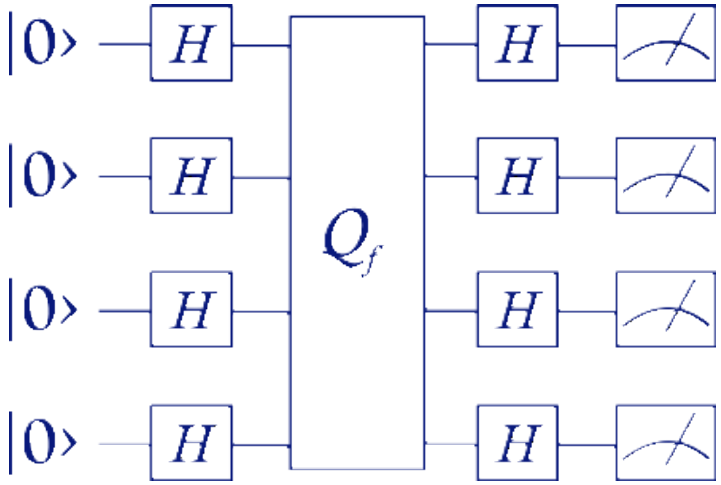


Figure 5. The circuit diagram for the Bernstein-Vazirani

The Classical Solution:

Classically, the oracle returns fs(x)=s·x mod2 given an input *x*. Thus, the hidden bit string *s* can be revealed by querying the oracle with the sequence of inputs, such as:

$$f(100...0_n) = s1$$

$$f(010...0_n) = s1$$

$$f(001...0_n) = s1$$

.

.

.

$$f(000...1_n) = s1\ ,$$

where each query reveals a different bit of *s* (the bit s_i). For example, with $x=100...0_n$ one can obtain the least significant bit of *s*, with x = 0100...0 we can find the next least significant, and so on. Therefore, we would need to call the function f_s (x) *n* times.

The Quantum Solution:

Using a quantum computer, we can solve this problem with total confidence after only one call to the function f(x)f(x). The quantum Bernstein-Vazirani algorithm to find the hidden bit string is very simple:

Procedure:

- $|0\rangle^{\otimes n}$ Initialize state to the $|0\rangle^{\otimes n}$ state, and output qubit to $|-\rangle$.
- $\rightarrow |+\rangle^{\otimes n} = \frac{1}{\sqrt{2}} \sum_{x\in\{0,1\}^n} |x\rangle$ Apply Hadamard gates to the input register.
- $\rightarrow \frac{1}{\sqrt{2^n}} \sum_x (-1)^{f(x)} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_x (-1)^{s\cdot x} |x\rangle = \frac{1}{2} (|0\rangle + (-1)^{s_1} |1\rangle) \otimes \frac{1}{2} (|0\rangle + (-1)^{s_2} |1\rangle) \otimes \ldots$ Apply Hadamard gates to the output register.
- $\rightarrow 0$ if $s_i = 0$ and 1 if $s_i = 1$ Measure to obtain final output *s*.

If the same problem statement is passed to both the Bernstein-Vazirani and the Deutsch-Jozsa algorithm with a similar circuit:

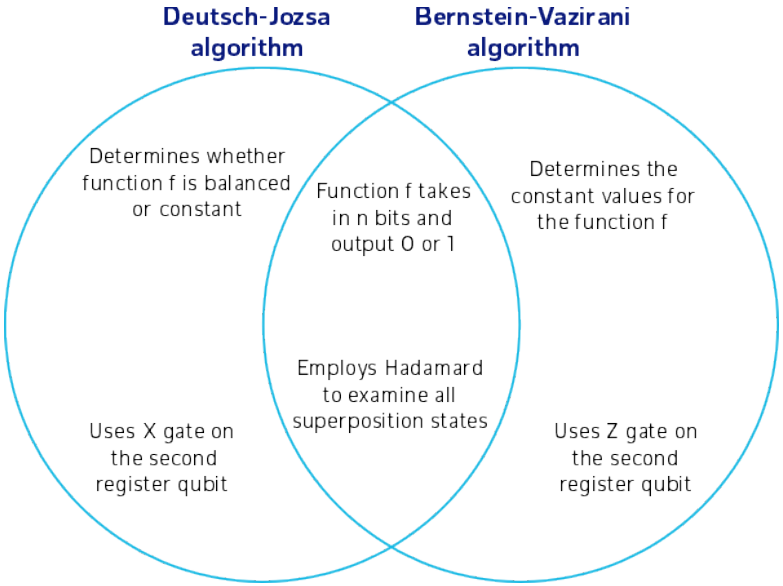


Figure 6. Represents the differences and similarities between the Deutsch-Jozsa and Bernstein-Vazirani algorithms

4.3 SIMON'S

Simon's algorithm was the first quantum algorithm to show an exponential speed-up against the best classical algorithm when solving a specific problem.

It has inspired the quantum algorithms based on the quantum Fourier transform and is utilized in the most famous quantum algorithm, Shor's factoring algorithm.

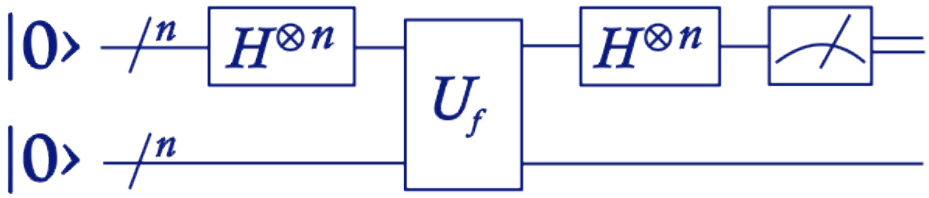


Figure 7. The circuit diagram for Simon's algorithm

Problem:

In Simon's problem, we are provided with a function from n bit strings to n bit strings,

$f : \{0, 1\}^n \rightarrow \{0, 1\}^n$,

such that for all $x,y \in \{0,1\}^n$.

For given x, y: $f(x) = f(y) \Leftrightarrow x = y \oplus b$.

Simon's problem is then, by querying f(x), to determine whether the function belongs to (i) $b = 0^n$ or to (ii) $b \neq 0^n$ when $\oplus$ denotes bitwise addition module two.

Classical Solution:

If we want to know what *b* is with 100% certainty for a given *f*, we must check up to $2^{n-1}+1$ inputs, where *n* is the number of bits in the input. This means checking just over half of all the possible inputs until we find two cases of the same output. Much like the Deutsch-Jozsa problem, if we get lucky, we could solve the problem within our first two tries. But if we happen to get an *f* that is one-to-one or get unlucky with an *f* that's two-to-one, then we're stuck with the full $2^{n-1}+1$. There are known algorithms that have a lower bound of $\Omega(2^{n/2})$, but the complexity grows exponentially with *n*.

Quantum Solution:

The circuit uses 2^n qubits arranged in two registers of *n* qubits each. Note that we're using a single line in the top register to represent *n* qubits and similarly in the bottom register.

Procedure:

- $|0\rangle^{\otimes n}|0\rangle^{\otimes n}$ Initialize state to the $|0\rangle^{\otimes n}$ state.
- $\rightarrow \frac{1}{\sqrt{2^n}}\sum_{x\epsilon\{0,1\}^n}|x\rangle|0\rangle^{\otimes n}$ Apply Hadamard gates to the input register.
- $\rightarrow \frac{1}{\sqrt{2^n}}\sum_{x\in\{0,1\}^n}|\mathrm{x}\rangle\,|\mathrm{f(x)}\rangle$ Apply query function Q_f.
- $\rightarrow \frac{1}{\sqrt{2}}(|\mathrm{x}\rangle+|\mathrm{y}\rangle)$ Output from the first register
- $\rightarrow \frac{1}{2^n}\sum_{z\in\{0,1\}^n}[(-1)^{x.z}+(-1)^{y.z}]\,|z\rangle$ If f(x) is two-to-one, the state $\frac{1}{\sqrt{2}}(|\mathrm{x}\rangle+|\mathrm{y}\rangle)$.
- $\rightarrow (-1)^{x.y}=(-1)^{y.z}$ Output from the first register

Using the criterion for Simon's problem, (f(x)=f(y)⟹x=y⊕s), we found that:

- x.y=y.z mod 2
- x.z=(x⊕s).z mod 2

- $x.z = x.z \oplus s.z \mod 2$
- $0 = s.z \mod 2$

Therefore, when we run the above quantum circuit each time, we get a bit string *z* which is orthogonal to the secret string *s*.

4.4 SHOR'S

Peter Shor released the quantum order-finding algorithm in a seminal paper in 1994 and noted that the problems with performing discrete logarithms and factoring can be reduced to order-finding. The final paper, including extended discussion and references, was published in 1997.

Shor's algorithm is an integer factorization quantum algorithm based upon the Fourier transformation. Today, the best-known classical algorithm requires super polynomial time to factor the product of two primes; the widely used cryptosystem, RSA (Rivest–Shamir–Adleman), relies on factoring being impossible for large enough integers. Communication security is heavily dependent on the application of RSA encryption. This method relies on the inaccessibility of large prime factors of a large composite number. In simple terms, the problem statement it solves is: the encrypter takes two (or more) large primes and multiplies them. The decrypter arduously tries to work backwards from the product to the factors.

A 193 decimal digital, which is known as RS-640, has taken approximately 30 2.2GHz-Opteron-CPU years, and over five months of calendar time, to solve.

Shor's algorithm consists of two parts:

- Performed on classical computers: The classical part reduces the factorization to a problem of finding the period of the function.

- Performed on quantum machines: The quantum part uses a quantum computer to find the period using the Quantum Fourier Transform (QFT).

We will focus on the second part performed on quantum computers. Shor's algorithm can solve the problem of period finding; since a factoring problem can be converted into a period finding problem in polynomial time, an efficient period finding algorithm can also be used to factor integers efficiently. The following steps need to be considered for the development of the algorithms:

- If *n* is even, prime, or a prime power, then exit.
- Pick a random integer x < n and compute gcd (x, n). If it is not 1, then we have achieved a factor of *n*.
- Quantum algorithm:

  Pick q as the smallest power of 2 with $n^2 \leq q < 2n^2$.

  Find period r of x^a mod n.

  The measurement gives us a variable c, which has the property $\frac{c}{q} \approx \frac{d}{r}$ where d ∈ N, where q is the dimension of a Hilbert space.

- To find d, r via continued fraction expansion algorithm. d and r are only determined if gcd(d, r) = 1 (reduced fraction).
- If r is odd, go back to 2. If $x^{\frac{r}{2}} \equiv -1$ mod n, go back to 2. Otherwise, the factors p, q = gcd($x^{\frac{r}{2}} \pm 1$, n).

From a cryptography perspective, Shor's algorithm is very important as it can factor large numbers much faster than classical algorithms (polynomial instead of exponential). NIST has initiated a process to solicit, evaluate, and

standardize one or more quantum-resistant public-key cryptographic algorithms.

4.5 VARIATIONAL QUANTUM LINEAR SOLVER (VQLS)

The VQLS algorithm is a variational quantum algorithm that utilizes a Variational Quantum Eigensolver (VQE) to solve systems of linear equations more efficiently than a classical computational algorithm. The output from the VQLS is similar to the Harrow-Hassidim-Lloyd (HHL) quantum linear-solver algorithm, except the HHL provides computation speedup that is advantageous over VQLS and requires a more robust fault-tolerant quantum hardware along with many qubits to perform computations.

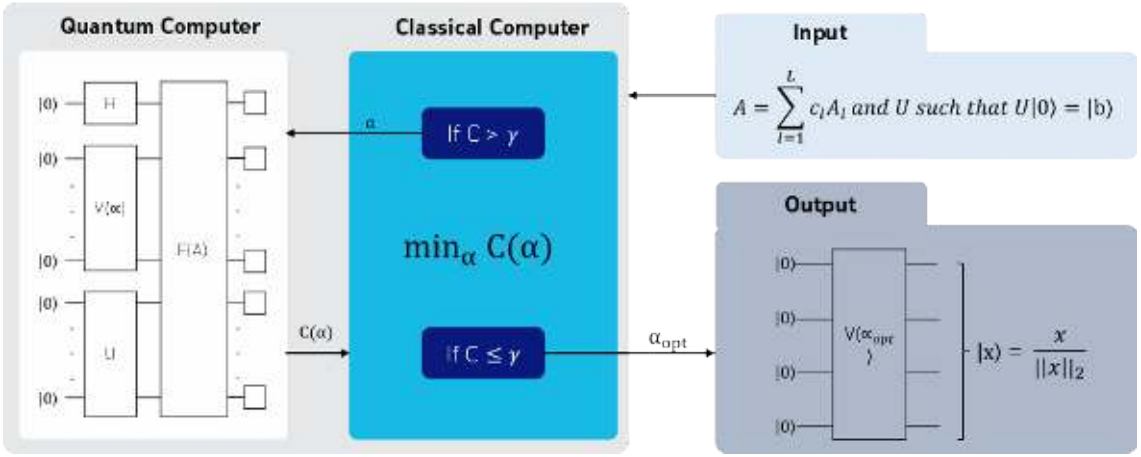


Figure 10. The top overview of the VQLS algorithm

VHQCAs can help reduce quantum circuit depth at the cost of additional classical optimization. Particularly, VHQCAs

can employ a short-depth quantum circuit to evaluate a cost function, which requires the parameters of a quantum gate sequence, and then leverage well-established classical optimizers to minimize this cost function.

As shown in Figure 10, the input A-matrix is a linear combination of unitary Al and a short-depth quantum circuit U which develop the state |b⟩, sent to the VQLS algorithm. The output of the VQLS algorithm provides a variationally quantum state of |x⟩ which can also be stated as a linear system: A|x⟩ = |b⟩. The parameters ∝ in the V(∝) are adjusted in the hybrid quantum-classical optimization loop until the cost C(∝) is reached below a user-specified threshold limit.

When it reaches the threshold limit, the loop is terminated and the resulting $\propto_{opt}$is passed to the sequencing gate V($\propto_{opt}$), which builds a state of |x⟩ = $\frac{x}{||x||_2}$ from which observable quantities can be computed. Furthermore, the final value of the cost C($\propto_{opt}$) provides an upper bound on the deviation of observables measured |x⟩ from observables measured on the exact solution.

4.6 HYBRID QUANTUM-CLASSICAL NEURAL NETWORK (HQCNN)

The hybrid quantum-classical neural network approach has the potential to solve today's computational problems in machine learning, which are not feasible to solve with classical computing. To develop an HQCNN architecture, the last layer of the classical layer can utilize a parameterized quantum circuit.

Parameterized quantum circuits (PQCs) offer a solid way to implement many algorithms and demonstrate quantum supremacy in the noisy intermediate-scale quantum (NISQ) era. The PQCs consist of fixed gates e.g., controlled NOTs (CNOTs), and adjustable gates, e.g., qubit rotations, that are dependent on the classical input vector.

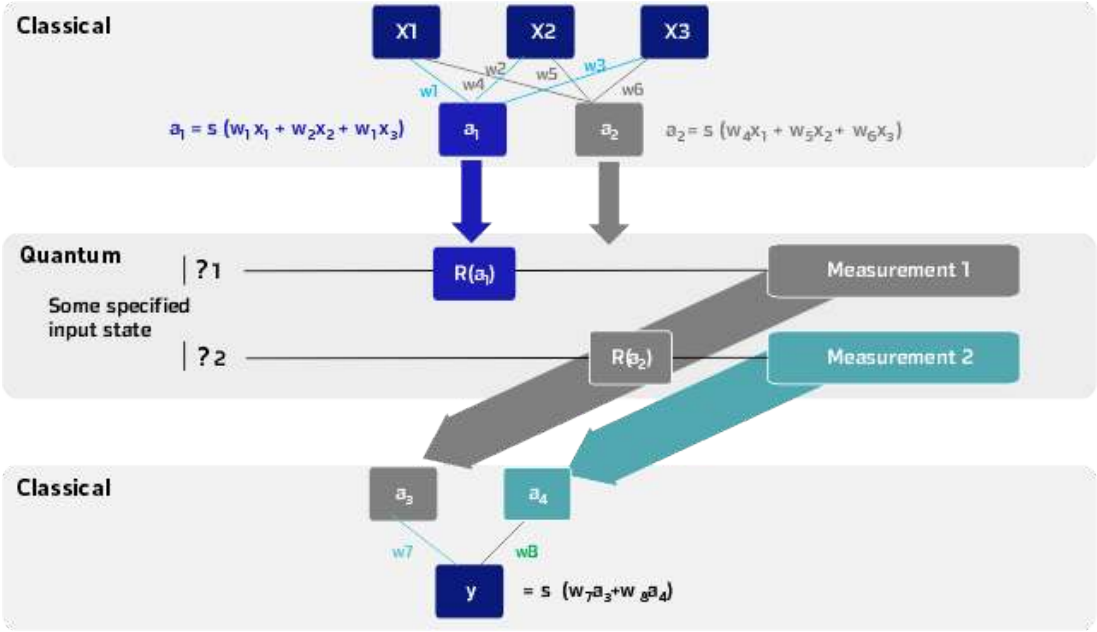


Figure 8. The PQCs architecture used to integrate the classical and quantum layers

As shown in Figure 8, the convolutional neural network (CNN) architecture consists of interconnected convolutional layers and pooling layers, ending with a fully connected layer. The output from the last layer of the classical layer is utilized as an input to the quantum layer using PQCs. The is a non-linear function and is the value of neuron i at each hidden layer. represents any rotation gate about an angle equal to and y is the final prediction value generated from the hybrid network.

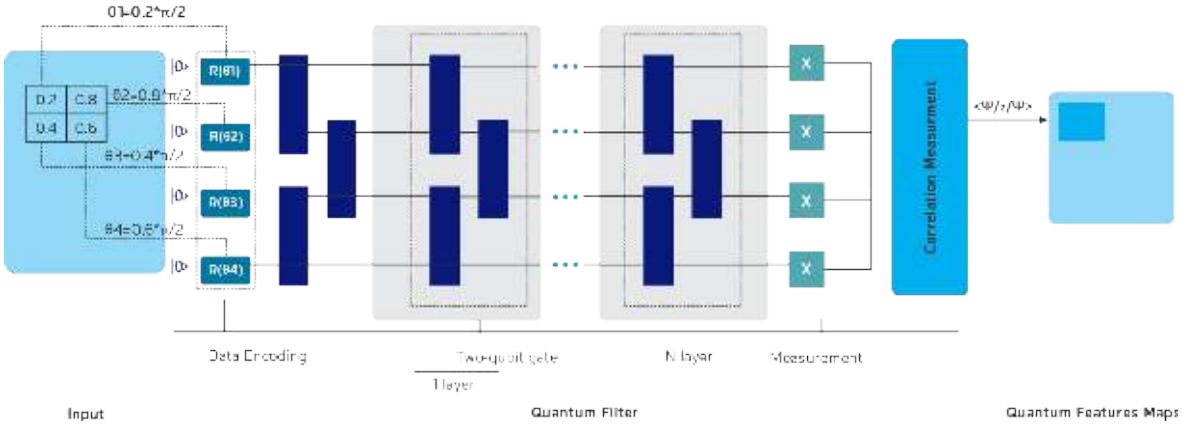


Figure 9. The HQCNN architecture

The primary objective of the classical convolutional layers is to extract the features from the inputs passing through these layers using a filter, which also requires a high computationally intensive step of CNN (shown in Figure 9, an HQCNN architecture where input is a 2x2 matrix which passes through a 6-filter quantum convolutional layers).

Each filter takes a 2x2 matrix and converts it into a separable 4-qubit quantum state then changes this state with a

PQCs. In the case of images encoded as a 3-dimensional matrix, the filter only works on the first two dimensions. At the end of the quantum filter, a correlational measurement is performed on the output of the quantum state and a scalar is found. On collecting all the scalar outputs, the final output of the quantum convolutional layer is a 3-dimensional matrix. A pooling layer is used to reduce the dimension of the data and it can be repeated to end with a fully connected layer.

To see the implementation results please check the circuit design and testing section.

5. DESIGN & TESTING

For development and testing, we are building our classification model by utilizing a PyTorch integration with a Qiskit-Aer simulator using PennyLane. In this development, we are employing the hybrid-classical neural network approach, which is utilized for the development of a classification task with a pretrained neural network architecture (ResNet-18) for classifying the images.

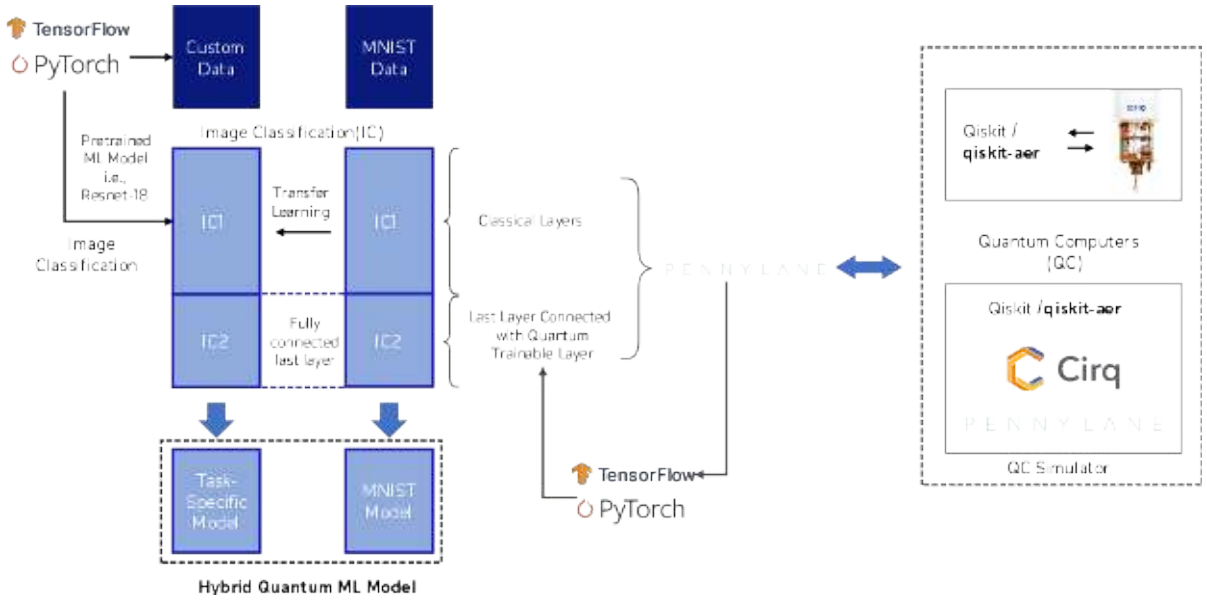


Figure 11. The architecture for the development of the hybrid quantum-classical ML

For study purposes, we are utilizing PyTorch integration with the Qiskit-Aer simulator using PennyLane to build our classification model. In this development, we are utilizing the hybrid-classical neural network approach, which

is used for the development of a classification task with a pre-trained neural network architecture (ResNet-18) for classifying the images. Additionally, there are various connectors available, such as Pennylane, which can help to connect the PyTorch/TensorFlow conveniently to multiple quantum simulators and real quantum computers at the backend.

As shown in Figure 12, we have developed a quantum circuit for a multi-class hybrid classical model, which utilizes 6 qubits for the multiclass classification.

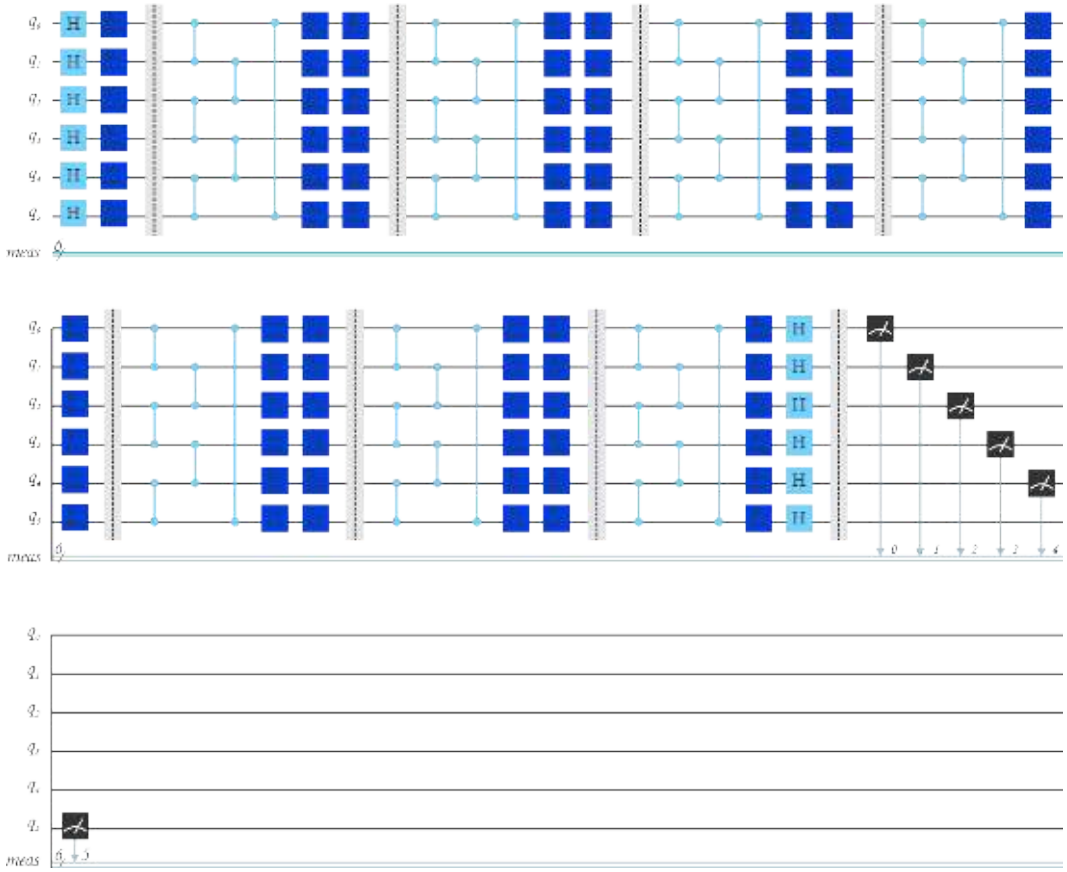


Figure 12. The quantum circuit for hybrid quantum-classical neural networks (HQCNN) for multi-class image classification

At the initial level, the 6-qubits are introduced to the Hamdard(H) gate to bring them to their superposition states. As the data embedding layer that takes inputs from the previous layer of the neural network, the output states from the H gate are processed with the Rotational Y (R_y) gate.

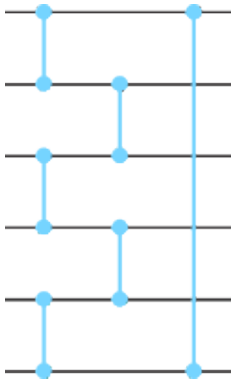


Figure 13. The entanglement layer created using Controlled Z(C_z) gate

The parametrized quantum circuit utilizes the Controlled Z(C_z) gate as an entanglement layer combined with Rotational Y (R_y) and Rotational Z (R_z) gate, which is repeated 6 times. The output from the last entanglement layer is rotated using the Rotational X(R_x) along with the H gate, and the result is measured in the Z-basis state from the quantum layer. To maintain the symmetry between the classical and quantum models' hyperparameters, we have utilized the cross-entropy loss function with Adam optimizer and a learning rate of 0.001 and 0.0004.

We have tested the above quantum circuit on two different datasets: brain tumor and corrosion. We have created a custom brain tumor dataset containing 4500+ images to develop a multi-class brain tumor classification model.

We have utilized transfer learning to replace the last layer for the resnet-18 with a quantum layer, which is then re-trained on the local quantum simulators for the development of the brain tumor model.

Table 1. Hyper-parameter values used for the classification model on different image datasets

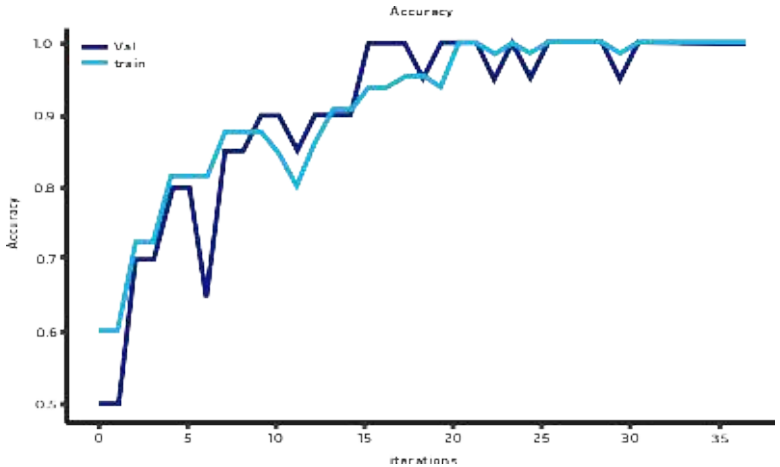
Parameters	**Metal Corrosion**	**Brain Tumor**
Quantum Depth	31	31
Number of Epochs	100	200
Batch Size	8	16
Learning Rate	0.0004	0.001
Max Accuracy Achieved During Iterations	100%	96.62%

We have tested another dataset of metal corrosion; we developed a custom metal corrosion dataset containing 1000+ images to develop a corrosion classification model.

The output from the quantum simulator for the two datasets are as follows:

Table 2. The output results from the metal corrosion dataset

Dataset	**Metal Corrosion**
Best Accuracy	100%

Test Results	
Accuracy Results	

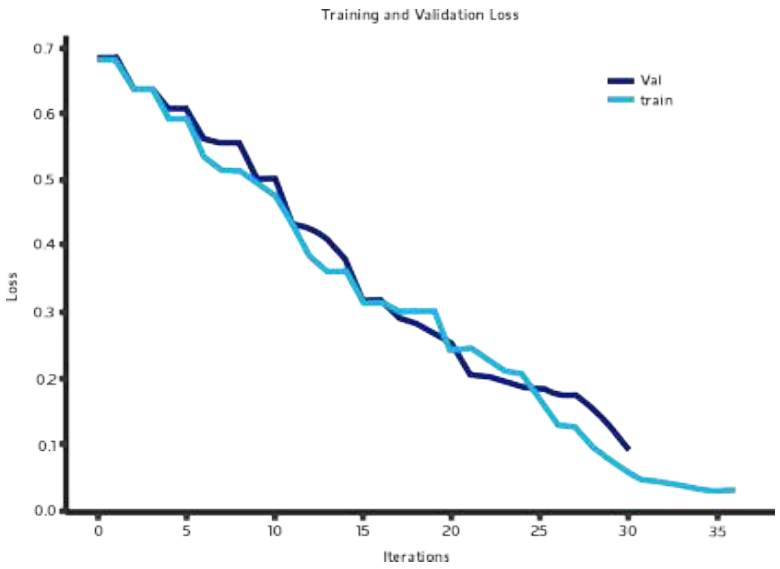
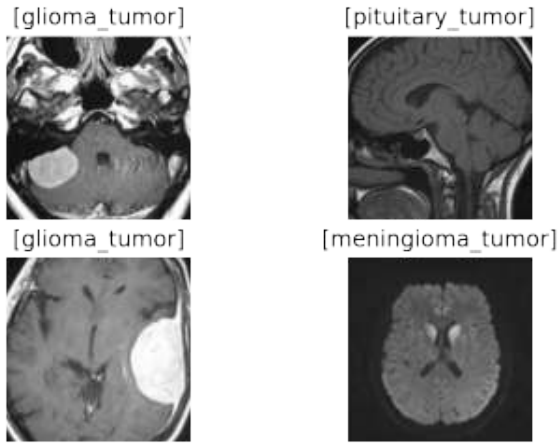
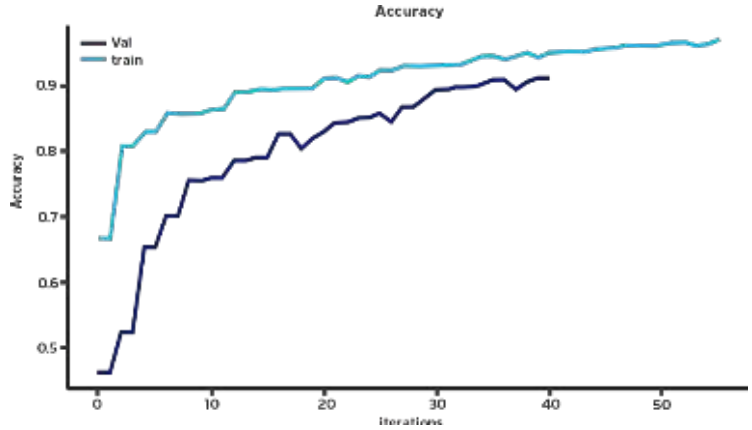
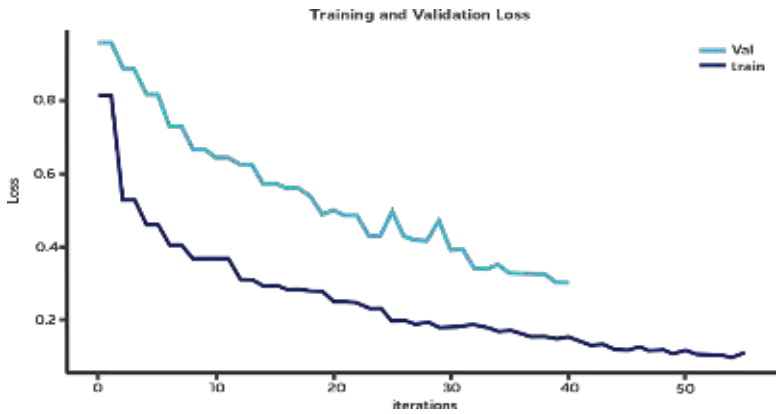
Loss Results	
Evaluation Metrics	precision recall f1-score corrosion 1.00 1.00 1.00 no_corrosion 1.00 1.00 1.00

Table 4. The output results from the brain tumor dataset

Dataset	**Brain Tumor** **(Multi-class)**
Best Accuracy	96.62%

Test Results	
Accuracy Results	

Loss Results	
Evaluation Metrics	

	precision	recall	f1-score
glioma_tumor	0.91	0.98	0.95
meningioma_tumor	0.88	1.00	0.94
pituitary_tumor	0.98	0.68	0.80

6. CONCLUSION

With the advancement in computational power and algorithmic advances, machine learning techniques are becoming more powerful in identifying patterns within the data. Today's quantum system can achieve the capability to produce a typical pattern that a classical system will not be able to produce efficiently, so it is sensible to propose that quantum machine learning may outperform classical machine learning tasks.

To develop this study on quantum machine learning, we have gone through various basic concepts of quantum computing which have helped us to develop the quantum circuit and explore various quantum algorithms for the development of the hybrid quantum-classical architecture. We have utilized PennyLane to integrate PyTorch with the IBM-Aer quantum simulator to develop a hybrid quantum-classical machine learning model.

During the test, we explored that the hybrid quantum machine learning model provided a high accuracy within a short span of time with a quantum depth of 31. Thus, indicating the superior properties of the quantum machine learning process. In the future, this study can be extended to other quantum machine learning algorithms, the impact of the quantum circuit depth and noise from the circuit which might have impact on the accuracy and training time of the model for multiple industries.

7. REFERENCES

1. Menard, Alexandre, et al. “A game plan for quantum computing.” *McKinsey Quarterly*, February 6, 2020.
2. Sandhu, Adarsh. “Quantum Computing: Technology with limitless possibilities looking for tough problems to solve.” *Springer Nature*, 2018.
    1. Adcock, Jeremy, et al. “Advances in quantum machine learning.” Quantum Engineering Centre for Doctoral Training, University of Bristol, December 10, 2015.
    2. TA-Swiss. Quantum Technologies, 2021
    3. Yanofsky, Noson S., and Mirco A. Mannucci. *Quantum Computing for Computer Scientists*. Cambridge University Press, August 2008.
    4. Tensorflow Quantum Computing, 2021. www.tensorflow.org
    5. José D. Martín-Guerrero, Lucas Lamata. "Quantum Machine Learning: A tutorial”, Neurocomputing, 2021.
    6. QuIC Seminar. “The Bernstein-Vazirani Algorithm.” Michigan State University.
    7. Watrous, John. “Lecture 6: Simon’s algorithm.” University of Waterloo, February 2, 2006.
    8. Grant Salton, Daniel Simon, and Cedric Lin. Exploring Simon’s Algorithm with Daniel Simon, 2021
    9. Erdi ACAR and Ihsan YILMAZ. COVID-19 detection on IBM quantum computer, 2020
    10. Junhua Liu,1, 2 Kwan Hui Lim. Hybrid Quantum-Classical Convolutional Neural Networks, 2021. arxiv.org
    11. Peter Young. The Bernstein-Vazirani Algorithm, 2019

12. Chih-Chieh Chen, Shiue-Yuan Shiau, Ming-Feng Wu & Yuh-Renn Wu. Hybrid classical-quantum linear solver using Noisy Intermediate-Scale Quantum machines, 2019
13. Andrea Mari, Thomas R. Bromley, Josh Izaac, Maria Schuld, Nathan Killoran. Transfer learning in hybrid classical-quantum neural networks, 2020
14. Ville Bergholm, Josh Izaac, Maria Schuld, Christian Gogolin. PennyLane: Automatic differentiation of hybrid quantum-classical computations, 2020
15. IBM Qiskit textbook. qiskit.org
16. Changyu Dong. Math in Network Security
17. Elisa Baumer, Jan-Grimo Sobez, Stefan Tessarini. Shor's Algorithm, 2015
18. Marcello Benedetti, Erika Lloyd, Stefan Sack, Mattia Fiorentini. Parameterized quantum circuits as machine learning models, 2019
19. Post-Quantum Cryptography, NIST
20. Carlos Bravo-Prieto, Ryan LaRose, M. Cerezo, Yigit Subasi, Lukasz Cincio, Patrick J. Coles. Variational Quantum Linear Solver, 2019
21. Andi Sama. Hello Tomorrow,I am a Hybrid Quantum Machine Learning, 2020
22. Aidan Pellow-Jarman. Near Term Algorithms for Linear Systems of Equations, 2021
23. Aidan Pellow-Jarman, Ilya Sinayskiy. A Comparison of Various Classical Optimizers for a Variational Quantum Linear Solver, 2021

24. Isaac Chuang and Michael Nielsen. Quantum Computation and Quantum Information, 2010

25. QuIC Seminar. The Bernstein-Vazirani Algorithm

8. GLOSSARY

$\|0\rangle$	Qubit state with vector output (1, 0)
$\|1\rangle$	Qubit state with vector output (0,1)
σ_x, σ_y and σ_z	Pauli sigma matrices
$\oplus$	Modulo two additions
U	Unitary operator or matrix
H	Hadamard gate
α, β, γ, and δ	Real-valued number
$\|\psi\rangle$	Superpositions
$\|+\rangle$	Positive state phase
$\|-\rangle$	Negative state phase
θ and ψ	Point on the unit's three-dimensional sphere
$\|\psi^+\rangle$	Positive phase state
$\|\psi^-\rangle$	Negative phase state
$\otimes$	Tensor product
I	Identity Matrix
$H^{\otimes n}$	Parallel action of n Hadamard gates
R_x	Rotational X gate
R_y	Rotational Y gate
R_z	Rotational Z gate
†	Hermitian conjugate