A COMPARATIVE STUDY OF IMAGE CLASSIFICATION WITH QUANTUM NEURAL NETWORKS

[1]NADINE MATONDO-MVULA, [2]HEPSEEBA KODE, [3]LAIALI ALMAZAYDEH, [4]KHALED ELLEITHY

[1,2,4]Department of Computer Science and Engineering, University of Bridgeport, CT, USA
[3]Department of Software Engineering, Al-Hussein Bin Talal University, Ma'an, Jordan
E-mail: [1]nmatondo@my.bridgeport.edu, [2]hkode@my.bridgeport.edu, [3]laiali.almazaydeh@ahu.edu.jo, [4]elleithy@bridgeport.edu

Abstract - This paper addresses aspects related to quantum machine-based classification such as data encoding into quantum states, Barren plateau, robustness, and effectiveness with developing a classical artificial neural versus fully and hybrid quantum neural network approach. The two approaches were compared with a classical artificial neural network using deep learning approaches concerning information processing speed, stability, and reliability. Moreover, fully and hybrid quantum approaches were tested on quantum simulators, considering the current limitations of quantum hardware. The models were trained on an image dataset describing the diagnosis of acute lymphoblastic leukemia. The test accuracy reached 98% for a classical artificial neural network approach, 99% for the hybrid quantum approach, and 64% for the fully quantum approach.

Keywords - Quantum Computing, Quantum Machine Noisy Intermediate-Scale Quantum, Parameterized Quantum Circuits, Quantum Neural Networks, Variational Quantum Classifier.

I. INTRODUCTION

In recent years, large technology companies such as Google, IBM, and Microsoft have invested in quantum computing research to apply quantum properties such as entanglement and superposition to solve complex problems more efficiently than with classical computers. Further, promising results in quantum computing have been proven to be successful in the machine learning field, where learning efficiency and runtime can be improved [1]. Another aspect of the increased interest is the ability to implement fast linear algebra, of which the complexity grows exponentially with the number of qubits [2]. Consequently, the research subfield Quantum Machine Learning (QML), the convergence between quantum computing and machine learning [3], has arisen to solve machine learning classification problems with quantum computers.

With the growing research field, existent approaches have been made in supervised and unsupervised learning, using principal component analysis (PCA), support vector machine (SVM), k-nearest neighbor, etc., considering that the data must be encoded into quantum states. For example, the Quantum-inspired machine learning approach uses quantum computing techniques to enhance classical machine learning methods. For instance, Helstrom

Quantum Centroid (HQC) is a supervised binary classifier that uses the Helstrom Measurements to measure the minimum error probability of in quantum states encoded vectors conductive to predict the class [4]. Another approach is the adaption of a machine learning algorithm in quantum computing. Quantum versions of the k-nearest neighbor method are solutions to retrieving the distance of two n-dimensional vectors through quantum measurement. The length of a vector is stored as a scalar product in the quantum state. Thus, the distance between quantum states can then be retrieved through the quantum swap test [5]. The last subfield of the quantum machine learning approach is hybrid quantum-classical machine learning. It combines quantum algorithms with classical algorithms to achieve higher performance and, as a result, to decrease learning costs [6]. One proposed solution is the usage of a variational quantum classifier with a single-level quantum system in the input layer to encode data in quantum states and then train all samples belonging to the same class at once. This method uses fewer training parameters and thus speeds up the training process, which is an advantage over classical supervised machine learning classifiers [7].

Quantum Machine Learning on Noisy Intermediate-Scale Quantum (NISQ) processors is the second generation of quantum machine learning [2]. It is a new promising trend and achievement to more powerful quantum technology involving entanglement between quantum states [8]. Further, these new approaches on quantum machine learning use algorithms based on parameterized quantum transformations, also known as Parameterized Quantum Circuits (PQCs), Variational Quantum Circuits (VQCs), or Quantum Neural Networks (QNNs). Moreover, PQC-based quantum machine learning algorithms are robust to noise [9] and only require shallow quantum circuits, which is highly desirable for noisy intermediate-scale quantum devices. Compared to classical neural networks, PQC-based quantum machine learning algorithms have two main advantages: stronger expressive power [10] and more computing power [11], originating from the superposition principle of quantum mechanics.

In general, the first step in developing QNNs is to translate classical data, which is usually represented as

a multi-dimensional array, into a quantum state. To learn from the training data, these algorithms are then optimized through black-box optimization heuristics or gradient-based methods concerning the cost function [2]. This process of optimization transformations is comparable to methods of classical deep learning, where back propagation is applied to adjust the learnable hyperparameters with the result to learn the decision boundary. Generally, an artificial neuron contains an inner product and an activation function. The inner product is the weighted sum of the inputs, whereas the activation function describes the transformation of the weighted sum of the inputs. In quantum computing, a neuron can be constructed by using an encoding method for the sample data, quantum circuits, and gates to perform the inner product and activation function approximate. There are many proposed architectures for QNNs that adapt linear and nonlinear functions with promising results. For example, Google's Quantum TensorFlow, PennyLane, and IBM's Qiskit are open-source frameworks for developing machine learning, including QNNs. For this paper, a pure and hybrid QNNs model for image classification was developed.

II. PROBLEM IDENTIFICATION

The design of QNNs image classification requires the following problems to be addressed to build a robust quantum machine-based classifier for real-world problems.

A. Encoding data into quantum states

The data must be encoded into quantum states to process classical data in quantum computers. Further, current quantum machines cannot load image data with high dimensions. The data must be preprocessed and converted into lower dimensions using techniques such as principal component analysis (PCA), linear least discriminant analysis (LDA), or other dimensionality reduction algorithm. It must be noted that such preprocessing may reduce the accuracy because of information loss [12]. This can be a major drawback when working with medical images. Researchers proposed many ways to encode data into QNNs. In [13], a circuit encodes an input vector into a state of an n qubit quantum system. After that, transformation is applied to encode the parameters using a parameterized unitary matrix with a set of parameters. Reference [14] analyzes the robustness of data encoding methods, including amplitude, angle, and dense encoding. According to [14], different encodings result in distinct learnable decision bounds, thus influencing the model's efficiency. An open question is which data encoding method to use concerning the dataset to aim quantum advantage in the model compared to classical methods. For the QNNs classifier, the top 3 promising data encoding methods for data images were analyzed and compared to find a suitable encoding method for the classifier.

B. Barren Plateaus

One of the drawbacks in training QNNs is the occurrence of Barren plateaus, also known as exponentially vanishing gradient problems, due to a lack of structure in the network architecture and randomly initialized parameters. This can lead to performance issues of gradient-based and gradient-free optimization problems [1]. Further, according to [15],[16], Barren plateaus appear on quantum machines with globally defined cost functions with large numbers of qubits. Therefore, each qubit is evaluated using a local cost function [16]. In [1], a layerwise learning strategy in combination with local cost functions was suggested to improve training efficiency in quantum circuits. For our model, we followed one of the guidelines to prevent the occurrence of Barren plateaus.

C. Robustness and Effectiveness

Current quantum machines have an intermediate size of qubits, and consequently, only a limited amount of data can be used, which may lead to feature loss in data. Therefore, it is almost impossible to test the classifier's robustness and effectiveness and compare it with a classical model. In addition, quantum noise is an obstacle when implementing quantum machine learning algorithms. Further, noises and the number of qubits differ from one quantum machine to another, which may yield dissimilar performance of the same machine learning model depending on which quantum machine it is tested. Due to the highly long training time, we initially trained and tested our model on a quantum simulator for the total duration of the training period and the number of used qubits to evaluate its robustness and efficiency using known matrices.

III. RELATED WORKS

Image classification in machine learning is training a model to recognize a defined set of target classes. With new developments in the quantum computing area, researchers have started to develop quantum approaches for artificial neural networks.

Cao et al. in reference [17] suggested a quantum neuron approach based on quantum Repeat-until-success (RUS) circuit building blocks to simulate neurons with a threshold activation function. Further, the quantum neurons can process superposition of inputs and maintain entanglement.
Jiang et al. in reference [18] proposed QNN using a tensor product data encoding method for binary and multi-classification problems with a single-qubit rotation gate and multi-qubit controlled-CNOT gate to rotate the feature vectors to orthonormal basis in the hypersphere of Hilbert space. This is accomplished by training the parameters correlating to a label and measuring the vectors to achieve the classification of the classes. The model was tested on three datasets and reached a training accuracy between 77% and 100%,

with an average sample size of 533 and an average feature size of 7.

Mathur et al. in reference [19] used two QNNs methods to perform binary classification on two different medical image datasets. Further, they proposed a unary amplitude encoding method to convert the data points into quantum states. The first approach uses a hybrid approach where training and forward inference is made on a NISQ using quantum circuits and gates to estimate the inner product between two feature vectors in the Hilbert space system. In contrast, the second approach uses a quantum pyramid circuit structure with orthogonal neural networks where trained matrices are orthogonal matrices. It is believed that such approaches improve accuracy and avoid the vanishing problem. An accuracy between 70%-96% was reached on training and test set depending on the dataset, quantum hardware, method, and inference.

Chalumuri et al. in reference [20] used a hybrid approach, i.e., a combination of quantum and classical approaches for a binary classification problem. The approach uses amplitude embedding encoding to convert the data onto quantum states. The hidden layer consists of CNOT gates for entanglement with a rotation gate at each qubit for state updates. The state of the qubit was measured using Pauli-Z measurements. It must be mentioned that the rotational parameters of the gate were optimized on a classical computer and fed back to the network until the desired output was found. The model reached a validation accuracy of 82.51%, whereas the classical Artificial Neural Networks (ANNs) only 39.86%.

Another quantum hybrid approach was proposed by Mutiara et al. in reference [21] to classify handwritten images of the numbers 3 and 6. Binary coding was used to convert the data into quantum data. Further, the model uses PQC layer for training the quantum data. The model reached a validation accuracy of 86.69% and 100% for testing.

IV. PROPOSED SOLUTION

This section discusses a proposed implementation of the problem identified in this paper. There are steps, namely, data encoding, model circuit, and measurement, to learn on classical data sets using QML.

A. Data Encoding

The dataset for this problem describes Acute Lymphoblastic Leukemia (ALL) diagnosis, a highly prevailing cancer provided as JPG files by [22]. ALL consists of 3,256 peripheral blood smear (PBS) images taken from 89 patients and divided into two classes: benign and malignant lymphoblastic. The former is categorized into three subtypes: Early, Pre-B, and Pro-B as in Table. 1. ALL dataset images were loaded as JPG files in the pixel size of 32x32, where each image was linearized and stored in an array. After that, Principal Component Analysis (PCA) was applied to reduce the dimensionality of the data to 4-8 data points in order to encode the data into a quantum state so the quantum machine could process that. PCA is a technique for reducing the dimensionality of large datasets in an interpretable way such that most of the information in the data set is preserved. The method increases interpretability but at the same time minimizes information loss [23]. Fig. 1 shows the dimensionality reduction of the dataset. The data points of Early, Pre-B, Pro-B are well separated. In contrast to the remaining clusters, the data points of Benign are overlapping. Thus, the quantum classification subtypes of malignant lymphoblastic were considered, which resulted in a 3-class classification problem.

Data Encoding for Quantum Machine Learning is the process of translating classical data into quantum states. Encoding can be categorized into digital encoding, which represents data as qubit strings, and analog encoding, which represents data in the amplitudes of a state. Contrary to classical machine learning, encoding data into quantum states is challenging and must be well analyzed as it can influence the runtime complexity of a quantum algorithm [24]. However, there are several proposed ways to encode classical data into the quantum machine. It is worth mentioning that there is no rule on how to encode data into quantum states, and for that reason, it is still an open research question and problem.

Classical data transformation into a Hilbert space can be done by applying data encoding circuits to map input feature vectors to feature space using so-called feature maps. A feature map in the quantum machine learning domain is, in general, a variational circuit. Qiskit, for instance, provides three different types of feature maps such as PauliFeatureMap (Pauli Expansion circuit), ZFeatureMap (First order Pauli Z-evolution circuit), and ZZFeatureMap (Second-order Pauli-Z evolution circuit). Further, varying and increasing the depth of a feature map results in higher performance and more entanglement into the model. Once the feature map has been applied, the quantum computer can analyze the input data in this feature space. The model circuit classifier can find a hyperplane to separate the data. Another method is to use amplitude encoding, which encodes the pixel intensities as probability amplitudes of different states of the system.

Benign	Early	Pre-B	Pro-B
504	985	963	804

Table 1: The Acute Lymphoblastic Leukemia (ALL) dataset



Figure 1: Dimensionality Reduction with PCA of the dataset

B. Quantum Classification

The quantum approach for image classification can be accomplished using quantum variational circuits for yielding better results.

Variational or parameterized quantum classifiers (VQC) are quantum artificial neural networks using quantum model circuits depending on free parameters. The quantum model circuits are constructed from quantum gates, which are reversible transformations on a quantum mechanical analog of an n-bit register. The circuit is based on unitary operations and depends on external parameters, optimized using classical optimization techniques like gradient descent through training to reduce the loss function. The trainable circuit is described as:

$$f(w,b,x)=y \tag{1}$$

Where f is the quantum neural network with the trainable parameters w, b and the input data x, Qiskit, for example, provides a local variational circuit called RealAmplitudes with the aim to increase the depth of the variational circuits and therefore to introduce more trainable parameters into the model. In general, the RealAmplitudes circuit is a heuristic trial wave function consisting of altering layers of Y rotations and CX entanglements. It must be mentioned that the entanglement pattern can be designed for a specific model or selected from a predefined set. EfficientSU2 is another variational circuit from Qiskit consisting of layers of single-qubit operations using CX entanglements and special unitary matrices of degree 2. Joshi et al. in reference [25], for example, compared local optimizers with VQC using a dataset with 1000 datapoints using 600 for training and remaining for validation where each data has five features testing. The experiment produced an accuracy of 77%.

C. Method

1) Quantum Approach

For the quantum approach, we first applied PCA to reduce the dimensionality of the data to 5 features. After that, we applied a ZZFeatureMap with a feature dimension of 5 and 2 repetition to encode the data into quantum states. Then we compose the ZZFeatureMap with a machine learning circuit classifier. We used the RealAmplitudes with linear entanglement and an initial depth of 1. To solve the exponentially vanishing gradients problem, also known as Barren plateaus, we adapted the approach in reference [26]. For that, we extended the ML base class Trainable Model from Qiskit. Trainable Model takes as an input an instance of a quantum neural network. We defined a Circuit QNN with a parameterized Quantum Circuit as input composed of a ZZF eature Map and Real Amplitudes. Further, we defined an interpreter to interpret the expectation values of a circuit.

We assigned labels from the output of a quantum circuit by counting the 1's from the passed bitstring modulo the number of the model classes. We used Gradient Descent as optimizer with Cross-Entropy as loss function. During the training, we increased the circuit layerwise starting with one layer of rotation with a short depth of circuits repetition and performed local optimization of the circuit parameters to increase the layer of rotation as shown in Fig 2. We bound already optimized parameters from the previous layer with the current layer, defined an objective function along with the gradients of this objective function to pass it to the optimizer. Such an approach increases the model's effectiveness and enables the development of larger circuits with linearized optimization of the parameters.

Due to the current quantum hardware limitation and long training period, we varied with up to 100 randomly selected images from the dataset with a feature size of 4-8 to train and test the model. The model with the best accuracy resulted in a 5-qubit quantum circuit using RealAmplitudes with a maximum of 5 repetitions and linear entanglement. Algorithm. 1 shows the assigned code and fig. 3 shows the initial architecture of the quantum model using five qubits for each pixel of the input data. The circuit uses 10 Hadamard gates, 52 CNOT gates for entanglement, 30 P-gates for rotations around the Z-axis, and 20 RY-rotations.

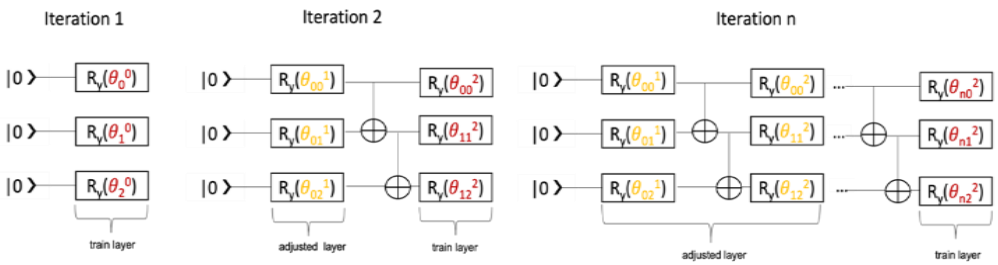


Figure. 2 Layerwise training approach

ALGORITHM 1: VQC for Quantum approach consisting of a FeatureMap composed with a RealAmplitudes

```
num_qubits=5
feature_map=ZZFeatureMap(feature_dimension=num_qubits, reps=2)
ansatz=RealAmplitudes(num_qubits, entanglement="linear")
qc=feature_map.compose(ansatz)
print(feature_map.decompose().count_ops())

OrderedDict([('cx',40),('p',30),('h',10)])
```

2) Hybrid Approach

The hybrid approach contrasts with the quantum approach, is a classical artificial neural network hybridized with a quantum circuit. Here, the data was separated into training and test set, where 75% of the data was used for training and 25% for testing. The classical part of the model consists of 3 convolutional 2D layers with a 3x3 kernel size for feature extraction. To down sampling the detection of features, we used MaxPooling with a pool size of 2x2. Weights and biases were applied to the convolved input for the forward propagation and passed through a nonlinear activation function.

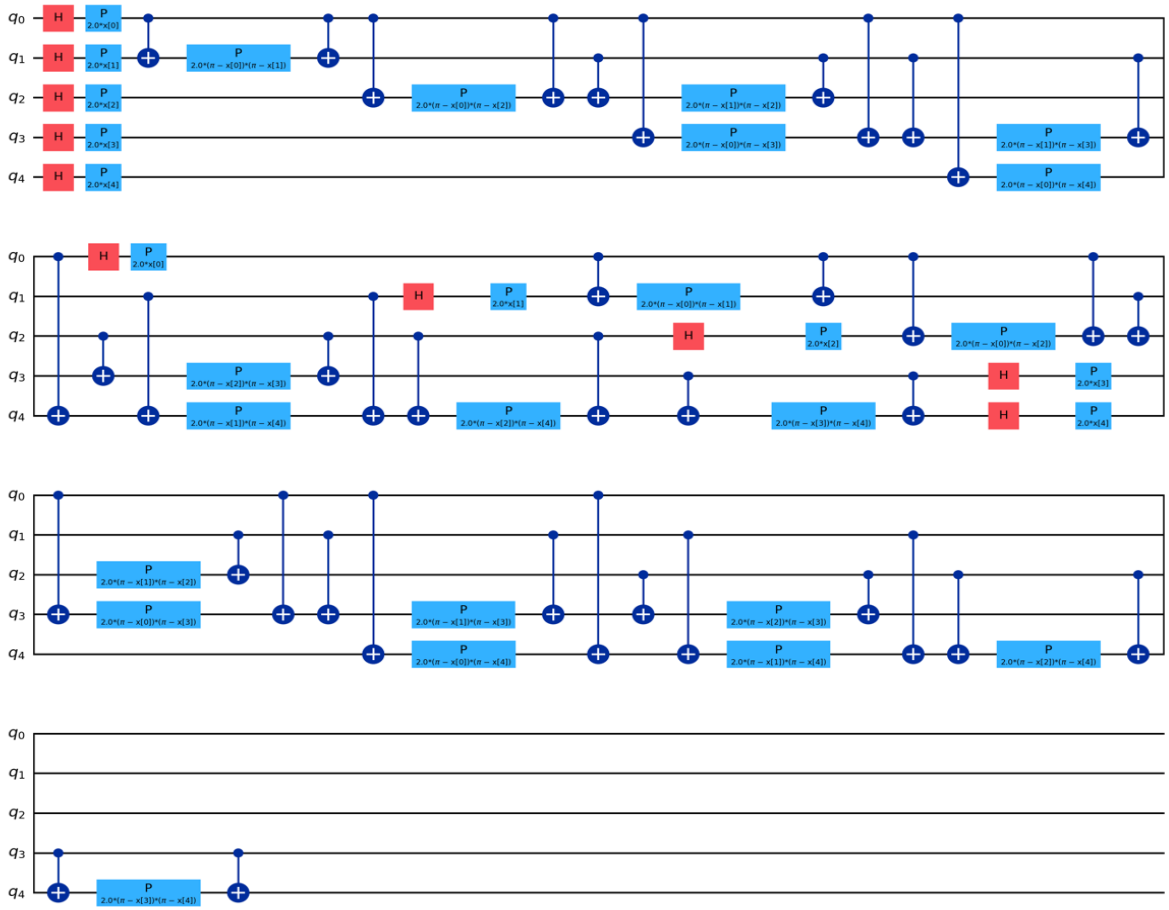


Figure 3. Achitecture of the quantum approach using a circuit of ZZFeatureMap for data encoding and RealAmplitudes as an example of VQC

Here we used the rectified linear activation function (ReLU), which outputs the input directly if it is positive; otherwise, zero. ReLU is a popular activation function for training artificial neural networks because it makes the model easier to train and results in improved performance in most cases. Furthermore, the model consists of two fully connected layers. To prevent overfitting and increase the performance of the model, we applied dropout to randomly drop out units in the network during training. We applied two fully-connected layers to connect every neuron from one layer to another. The values of the last neurons of the fully-connected layer were then fed as parameters into a Quantum circuit. After that, we applied rotation on all qubits and measured the output of the parameterized quantum circuit. With the backpropagation, it is possible to find the gradients of the parameters with respect to the loss function. In this, we applied the parameter shift rule and is defined as follows:

$$\nabla_\theta \text{Quantum Circuit}(\theta) = \text{Quantum Circuit}(\theta + s) - \text{QuantumCircuit}(\theta - s) \quad (2)$$

Where θ are the parameters of the quantum circuit and s is the macroscopic shift. By creating two copies of

the parameter and shifting one right and one left, the difference of the two shifted parameters concludes to the gradient for the parameter. To optimize the neuralnetworks, we used Adaptive Moment Estimation (Adam) as an optimizer with a learning rate of 0.001. To minimize the loss in our training, we used Cross-Entropy as our loss function.

3) Classical Approach

Classical neural network is an algorithm that can take in an input image, assign crucial learnable weights and biases to distinct objects in the image, and distinguish one from the other. The classical approach has a similar architecture as the classical part of the quantum hybrid approach. Here, we also split the data into training and testing set with 75% for training and the remaining 25% for testing. We used three 2D convolutional layers with a kernel size of 3x3 and MaxPooling with a pool size of 2x2. We also used dropout to prevent overfitting and increase our model's performance. After that, we applied two dense to connect the neurons from the previous layer. Furthermore, we used ReLU as an activation function for the hidden layers and Softmax in the last layer to convert the data into a probability distribution.

We used Adam as an optimizer with a learning rate of 0.001 and Cross-Entropy as a loss function.

V. MATHEMATICAL MODEL

A. Quantum Approach

For the quantum approach, we first encode the classical data into quantum states using ZZFeatureMap with two repetitions. As already mentioned in the previous section, ZZFeatureMap is a Second-order Pauli-Z evolution circuit with a data input transformation described as in: $\overrightarrow{(x)} \in \mathbb{R}^n$:

$$U_{\phi\overrightarrow{(x)}} = exp\left(i \sum_{S\subseteq[n]} \phi_S\overrightarrow{(x)} \prod_{i\epsilon S} P_i\right) \tag{3}$$

Where $P_i \in \{I,X,Y,Z\}$ is a variable and denotes the Pauli matrices. The index S describes the connectivity between different qubits or data points: $S\in$ { $\binom{n}{k}$ $combinations$, k=1,…n}. It must be mentioned that per default, the data mapping

$$\emptyset_S\ is\ (\vec{x}) = \begin{cases} \prod_{j\in S}^{x_0} & if\ k = 1 \\ (\pi - x_i) & otherwise \end{cases} \tag{4}$$

Constructing the ZZFeatureMap from the Pauli Expansion circuit is described as $k = 2, P_0 = Z, P_{0.1} = ZZ$. After that, we composed the ZZFeatureMap with the RealAmplitudes. The RealAmplitudes is a two-local circuit consisting of alternating layers of Y rotations denoted as:

$$RY(\theta) = \exp\left(-i\frac{\theta}{2}Y\right) = \begin{pmatrix} cos\frac{\theta}{2} & -sin\frac{\theta}{2} \\ sin\frac{\theta}{2} & cos\frac{\theta}{2} \end{pmatrix} \tag{5}$$

and CX entanglements. After that, we used the forward pass to obtain the objective function. We then calculated the objective function by using the forward pass. We pass the objective values together with the gradient of the objective function to the optimizer, which is the Gradient Descent. The gradient descent is described as:

$$\theta - \eta * \nabla_\theta \mathcal{J}(\theta; x^i; y^i) \tag{6}$$

where θ are weights and biases, η learning rate, ∇ gradient, $\mathcal{J}$ loss, and i one data item. We bound the optimal parameters received from the optimizer to the ansatz and increased the layer size afterward. We repeated the process five times.

B. Hybrid Approach

For the forward propagation, we fed the model a three channels image of the size 32x32 pixels as in Table. 2. After that, the first 2D convolution filter with a filter size of 16 and a kernel size of 3x3 was applied at every position of the input. The output shape of the convolution to the input shape is defined as follows:

$$W_2 = [(W_1 - F + 2P)/S] + 1 \text{ x } D$$
$$H_2 = [(H_1 - F + 2P)/S] + 1 \text{ x } D \tag{7}$$

Where W is the width, H is the height of the input dimension, F is the size of the filter/kernel, S is the stride, P is the padding, and D is the Depth/Number of filters. After applying the first 2D convolution filter the output dimension is: [(32 - 3 + 2 * 0) / 1] +1 x 16 = (30x30x16). After that, the weighted sum of the input is calculated and fed through the ReLU activation function, which is defined as follows:

$$f(x) = \begin{cases} x, & x > 0 \\ 0, & otherwise \end{cases} \tag{8}$$

To reduce the number of parameters and computation in the network, we applied MaxPooling with a pool size of 2x2. The pooling layer is defined as follows:

$$W_2 = [(W_1 - F)/S] + 1 \text{ x } D$$
$$H_2 = [(H_1 - F)/S] + 1 \text{ x } D \tag{9}$$

Where W is the width, H is the height of the input dimension, F is the size of the filter/kernel, S is the stride, and D is the Depth/Number of filters of the previous layer. The output of the image after passing through the MaxPooling is: [(30–2) / 2] + 1 x D = (15x15x16). After that, another 2D convolutional filter of the kernel size 32 with MaxPooling and dropout

was applied. The last 2D convolutional layer is of a kernel size of 64. The last two layers of the network from the classical part are fully-connected layers with a volume size of 64x2x2x64 and 64x3. The output of the last fully connected layer is used as inputs for the 5-qubit parameterized circuit. After that, we performed an RY-rotation by the angle θ on all inputs. The output is measured in the z basis calculating the σ_z expectation denoted as: $\sigma_z = \sum_i z_i p(z_i)$.
To calculate the gradient of the loss function with respect to the parameters, we applied the parameter shift rule as described in previous sections. We used Categorical Cross-Entropy as a loss function described as:

$$-\sum_{i=0} T_i \log(S_i) \tag{10}$$

Where T_i is the truth value and S_i is the Softmax probability. We used Adam as a gradient descent optimizer with a learning rate of 0.001.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1,16,30,30]	448
Conv2d-2	[-1,32,13,13]	4,640
Conv2d-3	[-1,64,4,4]	18,496
Dropout2d-4	[-1,64,2,2]	0
Linear-5	[-1,64]	16,448
Linear-6	[-1,3]	195
Dropout2d-7	[-1,3]	0

Total params: 40,227
Trainable params: 40,227
Non-trainable params: 0

Input size (MB): 0.01
Forward/backward pass size (MB): 0.16
Params size (MB): 0.15
Estimated Total size (MB): 0.33

Table 2: The architecture of the hybrid model using three 2D Convolutional layers, MaxPooling, 2 Fully Connected-Layers

C. Classical Approach

As mentioned in the previous section, the classical approach has a similar architecture as the classical part of the quantum hybrid approach as shown in Algorithm. 2. We fed the model a three channels image of the size 32x32 pixels. After that, the first 2D convolution filter with a filter
size of 16 and a kernel size of 3x3 was applied. After computing the weighted sum of the inputs and feeding the output of the weighted sum through a ReLU function and applied MaxPooling with a pool size of 2x2 after. Then we applied a second 2D convolutional filter with a filter size of 32 and kernel size of 3x3. The third 2D convolution filter uses 64 filters with a kernel size of 3x3. We applied a dropout of 0.25. We Flatten the output of the convolutional layers using a fully connected layer. After that, we used two dense layers, one with dense value 64 and activation function ReLU and another with dense value 3 with activation function SoftMax, to convert the output to a probability distribution. The backpropagation computes the gradient of the loss function with respect to all learnable parameters by using the chain rule to calculate the partial derivatives, also known as the gradient descent algorithm. After computing the partial derivatives, all learnable parameters can be updated and optimized.

ALGORITHM 2: The Architecture of the classical CNN model

```
model=Sequential ()
model.add(Conv2D(filters=16, kernel_size=(3,3), activation='relu',input_shape=(img_height,img_width,3)))
model.add(Conv2D(filters=32, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(filters=64, kernel_size=(3,3), activation='relu')),
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(rate=0.25))
model.add(Flatten())
model.add(Dense(units=64, activation='relu'))
model.add(Dropout(rate=0.25))
model.add(Dense(units=3, activation='softmax'))
model.summary()
```

VI. RESULTS AND DISCUSSION

To evaluate the performance of the proposed models in classifying the ALL dataset, a confusion matrix was generated. The confusion matrix calculates four statistical measures namely true positive (TP), also known as sensitivity, recall, or hit rate, true negative (TN), also known as specificity, or selectivity, false positive (FP), also known as type 1 error, and false negative (FN), also known as type 2 error:

$$Acccuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}} \tag{10}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{11}$$

$$F1\ Score = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{12}$$

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{13}$$

A. Quantum Approach

We trained the model on 53 images for 30 iterations and evaluated 22 images. Table. 3 and fig. 4 show the results of the quantum approach. An accuracy of 64% was reached, 50% precision, 29% recall, 36% f1-score for Early, 67% precision, 80% recall, 73% f1-score for Pre-B, 67% precision, 80% recall, and 73% f1-score for Pro-B.

	precision	Recall	F1-score	support
Early	0.50	0.29	0.36	7
Pre-B	0.67	0.80	0.73	5
Pro-B	0.67	0.80	0.73	10
Accuracy			0.64	22
Macro avg	0.61	0.63	0.61	22
Weighted avg	0.61	0.64	0.61	22

Table 3: The results of quantum approach

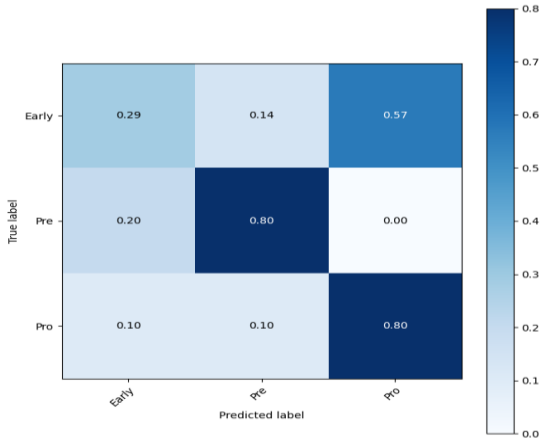


Figure. 4. Quantum approach - Confusion Matrix for Multi-class classification of All Dataset

B. Hybrid Approach

We trained the hybrid model with 40, 227 parameters and 150 epochs on 2064 images and evaluated 688 images using a batch size of 32. The training took approximately 8.5 hours. Table. 4 and fig. 5 show the results of the hybrid approach. An accuracy of 99% was reached, 98% precision, 100% recall, 99% f1-score for Early, 100% precision, 97% recall, 99% f1-score for Pre-B, 99% precision, 100% recall, and 100% f1-score for Pro-B.

	precision	recall	F1-score	support
Early	0.98	1.00	0.99	235
Pre-B	1.00	0.97	0.99	259
Pro-B	0.99	1.00	1.00	194
Accuracy			0.99	688
Macro avg	0.99	0.99	0.99	688
Weighted avg	0.99	0.99	0.99	688

Table 4: The results of hybrid approach

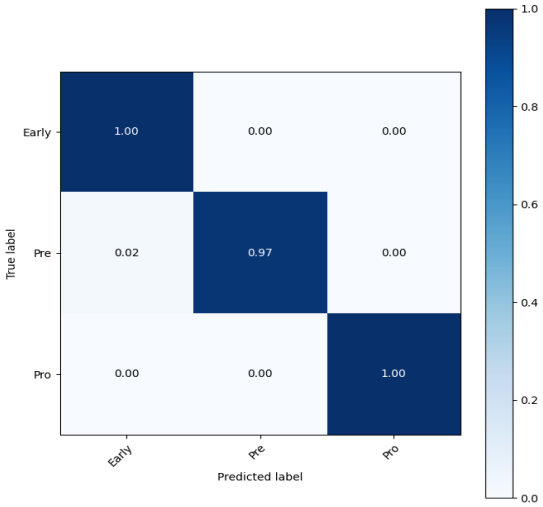


Figure. 5. Hybrid approach - Confusion Matrix for Multi-class classification of All Dataset

C. Classical Approach

The classical approach has trained the hybrid with 171,299 parameters and 150 epochs on 2057 images and evaluated 681 using a batch size of 32. We excluded some of the images for clarity. The training took approximately 3 min. Table. 5 and fig. 6 show the results of the classical approach. An accuracy of 98% was reached, 97% precision, 98% recall, 98% f1-score for Early, 100% precision, 98% recall, 99% f1-score for Pre-B, 98% precision, 98% recall, and 98% f1-score for Pro-B.

	precision	recall	F1-score	support
Early	0.97	0.98	0.98	246
Pre-B	1.00	0.98	0.99	240
Pro-B	0.98	0.98	0.98	201
Accuracy			0.98	687
Macro avg	0.98	0.98	0.98	687
Weighted avg	0.98	0.98	0.98	687

Table 5: The results of classical approach

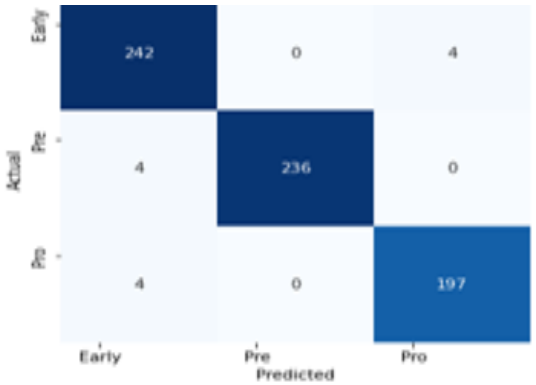


Figure. 6. Classical approach - Confusion Matrix for Multi-class classification of All Dataset

VII. CONCLUSION AND FUTURE WORK

We developed three models, namely a quantum, hybrid, and a classical artificial neural network for multi-class classification of (Acute Lymphoblastic leukemia (ALL)) data, and reached an accuracy of 64%, 98%, and 99%. The current quantum model delivers a performance of only 64%. It is due to the need for customized quantum data encoding circuits for such complex data. Another reason is that relevant data was lost during the dimensionality reduction. In addition, with the current quantum hardware limitation, the training time is very long compared to the classical variant and that for a small dataset. Further, noticed with the quantum variant that depending on the division of the data, the testing accuracy varied very strongly but still did not give more than 64% accuracy. We believe that with a custom FeatureMap the accuracy can be increased. Despite its long training, the hybrid model did quite well and gave us a good accuracy, 99%. For future work, we will develop custom FeatureMaps to improve our performance. We also want to test our model on different Quantum computers, if possible, as we could not do this due to the training time and therefore could only test them on simulators.

REFERENCES

[1] F. Phillipson, "Quantum Machine Learning: Benefits and Practical Examples," In Proceedings of the International workshop on Quantum Software Engineering & Programming (QANSWER), Spain, pp. 51-56, 2020.

[2] M. Broughton, et al. "TensorFlow Quantum: A Software Framework for Quantum Machine Learning," ArXiv preprint ArXiv:2003.02989, 2020.

[3] T. Khan, and A. Robles-Kelly, "Machine Learning: Quantum vs. Classical," IEEE Access, vol. 8, pp. 219275-219294, 2020.

[4] G. Sergioli, R. Giuntini and H. Freytes, "A new quantum approach to binary classification," PLOS ONE, pp. 1-14, 2019.

[5] M Schuld, I.. Sinayskiy and F. Petruccione," An introduction to quantum machine learning," Contemporary Physics, vol. 56, no. 2, pp. 172-185, 2015.

[6] Z. Abohashima, M. Elhosen, E. Houssein and W. Mohamed, "Classification with Quantum Machine Learning: A Survey," 2020, arxiv.org/pdf/2006.12270.pdf.

[7] S. Adhikary, S. Dangwal and D. Bhowmik, "Supervised learning with a quantum classifier using a multi-level system," Quantum Information Processing, vol. 19, no. 3, pp. 1-12, 2020.

[8] J. Preskill, "Quantum Computing in the NISQ era and beyond," Quantum, vol. 2, 2018.

[9] L. Gentini, A. Cuccoli, S. Pirandola, P. Verrucchi and L. Banchi, "Noise-resilient variational hybrid quantum-classical optimization," Physical Review, vol. 102, no. 5, 2020.

[10] S. Sarma, D.-L. Deng and L.-M. Duan, "Machine learning meets quantum physics," arXiv preprint arXiv:1903.03516, 2019.

[11] S. Lloyd and C. Weedbrook, "Quantum generative adversarial learning," Physical Review Letters, vol. 121, no. 4, 2018.

[12] N. Mathur et al., "Medical image classification via quantum neural networks," arxiv.org/pdf/2109.01831.pdf, 2021.

[13] M. Schuld, A. Bocharov, K. Svore and N. Wiebe, "Circuit-centric quantum classifiers," Physical Review, vol. 101, no, 3, 2020.

[14] R. LaRose and B. Cole, "Robust data encodings for quantum classifiers," Physical Review, vol. 102, no. 3, 2020.

[15] M. Cerezo, A. Sone, T. Volkoff, L. Cinco, P. Coles, "Cost function dependent Barren plateaus in shallow parameterized quantum circuits," Nature communications, vol. 12, no. 1, pp. 1-12, 2021.

[16] Arrasmith, M. Cerezo, P. Czarnik, L. Cincio and PJ. Coles, "Effect of barren plateaus on gradient-free optimization," Quantum, vol. 5, 2021.

[17] Y. Cao, G. Guerreschi and A. Aspuru-Guzik." Quantum Neuron: an elementary building block for machine learning on quantum computers," arXiv preprint arXiv:1711.11240, 2017.

[18] J. Jiang, X. Zhang, C. Li, Y. Zhao and R. Li, "Generalization Study of Quantum Neural network," arXiv preprint arXiv:2006.02388, 2020.

[19] N. Mathur, J. Landman, Y. Li, M. Strahm, S. Kazdaghli, A. Prakash, I. Kerenidis, "Medical image classification via quantum neural networks," arXiv prePrint arXiv:2109.01831, 2021.

[20] Chalumuri, R. Kune and B. S. Manoj, "Training an Artificial Neural Network Using Qubits as Artificial Neurons: A Quantum Computing Approach," Procedia Computer Science, vol. 171, pp. 568-575, 2020.

[21] Mutiara, M. Slamet, R. Refianti and Y. Sutanto, "Handwritten Numeric Image Classification with Quantum Neural Network using Quantum Computer Circuit Simulator" International Journal of Advanced Computer Science and Applications(IJACSA), vol. 11, no. 10, pp. 310-319, 2020.

[22] M. Aria, M.cGhaderzadeh, D. Bashash, H. Abolghasemi, F. Asadi and A. Hosseini, "Acute Lymphoblastic Leukemia (ALL) image dataset." Kaggle, DOI: 10.34740/KAGGLE/DSV/2175623, 2021.

[23] Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments." Philosophical Transactions of the royal society, vol. 374, pp. 1-14, 2016.

[24] M. Weigold, J. Barzen, F. Leymann and M. Salm., "Data Encoding Patterns for Quantum Algorithms," In HILLSIDE Proc. of Conf. on Pattern Lang. of Prog. 22, 2020.

[25] N. Joshi, P. Katyayan and S. Ahmed, "Evaluating the Performance of Some Local Optimizers for Variational Quantum Classifiers," In journal of Physics, vol. 1817, no. 1, 2021.

[26] F. Tacchino, PK. Barkoutsos, C. Macchiavello, D. Gerace, I. Tavernelli and D. Bajoni, "Varational learning for quantum artificial neural networks," In 2020 IEEE International Conference on quantum computing and engineering (QCE), pp. 130-136, 2020.

★★★