Final Evaluation Project Report

TOPIC

Signature Verification system using Quantum Neural Networks

Problem Statement

Signature Verification using Enhanced Quantum Neural Network

We aim to use a Quantum Neural Network for identifying forged and genuine signatures.

We all know the importance of signatures in our present modern world. All documents in financial services or on letters, documents, checks, security documents etc. are validated by ones signatures.

As so much research is going on quantum computing. Signature Verification is also one of the application of Quantum Computing.

So, our aim was to improve the existing ways of signature verification using mathematical concepts like Quantum Computing and combining them with advanced computing algorithms like Neural Networks.

State of the art

Signature forgery detection finds its application in many fields like net banking, passport verification system, it acts as a form of confirmation of identity in public examinations, credit card transactions, bank checks and many more. Therefore for growing demand for protection of individual identity, the design of an automatic signature verification system is needed.

This system if performs efficiently will be able to differentiate a real signature from a forged signature. We will be taking advantage of the advanced machine learning algorithms like Neural Networks and combining them with the advanced concepts of Quantum Computing to improve the presently used algorithms.

It can be used in various sections of society like banking and government bodies where handwritten signatures are of the highest importance. These concepts can also help improve other authentication and validation systems.

Limitations

1) Not having a real Quantum Computing Environment.

2) Measuring the result of the computation tells us an answer, but for some quantum algorithms, not necessarily the correct answer. Because the result of some quantum algorithms is based on the probability that was configured by the quantum operations, these computations are run multiple times to get a probability distribution and refine the accuracy of the results. Assurance that an operation returned a correct answer is known as quantum verification and is a significant challenge in quantum computing.

3) Due to limited sources we could not work on very large dataset.

4) Inconsistent and non uniformity in a person's signature might lead to incorrect predictions of legitimate and fraudulent signatures.

Objectives and Work Distribution

Our objective is to explore and implement Quantum Neural Network and apply it for Signature Verification and to make the measurement outcome correspond to the correct binary label of the input images.

Work Distribution

We divided our project into 4 phases -

Phase 1 - 1)Study concept of Machine Learning 2) Basic Neural Network 3) Image Classification 4) Concept of Signature Verification

Phase 2 - 1) Convolutional Neural Network 2) Study about Quantum Computing
3) Data Preprocessing 4) Implementation of Signature verification using CNN

Phase 3 - 1) More work on Data Preprocessing 2) Studied concept of Quantum Enhanced Neural Network

Phase 4 - 1) Try to implement Signature Verification using Quantum Neural Network.

Proposed Design

We plan to design a model that takes a signature as an input and then predicts whether that signature is a real or a forged signature using quantum computing.

Without access yet to an actual free quantum computer, we ran a classical computer as a simulator of a quantum device.
→ Initialize qubits to the desired state → Perform operation to transform the state of qubits → Measure the state of new qubits
Quantum operations are similar to logic operations in classical computing, such as X, H, XX, ZZ etc. An operation can be as basic as flipping a qubit state from 1 to 0 or entangling a pair of qubits, to using multiple operations in series to affect the probability of a superposed qubit collapsing one way or the other.

We apply these gate circuit on our date and create our model circuit. This model will learn and identify the characteristics of a real signature and the characteristics of a forged signature. These characteristics will help the model to learn more about the real and forged signatures. This result will be cross checked with the original data to cross verify and predict the success and accuracy rate of the model.

cirq.X

Matrix:

[[0, 1],

[1, 0]]

cirq.h

The Hadamard gate.

Matrix:

[[s, s],

[s, -s]]

s=sqrt(0.5)

cirq.XX

The tensor Product of two X gates.

cirq.ZZ

The tensor product of two Z gates.

Basic Architecture

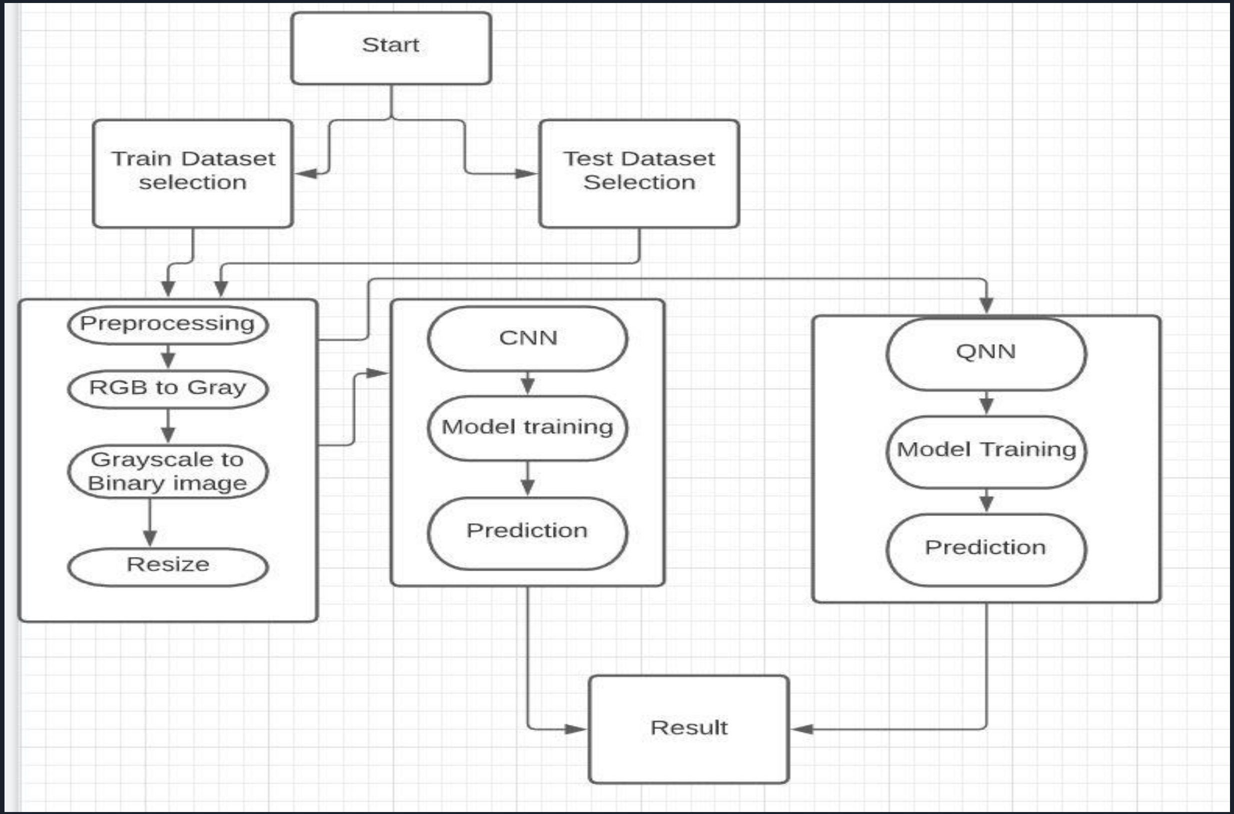
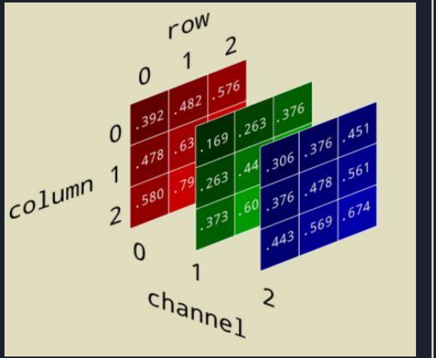


Image Preprocessing

RGB to Grayscale :-



We apply an average method for converting RGB to grayscale images where we convert 3 planes into one plane by taking the average of the value of each cell.

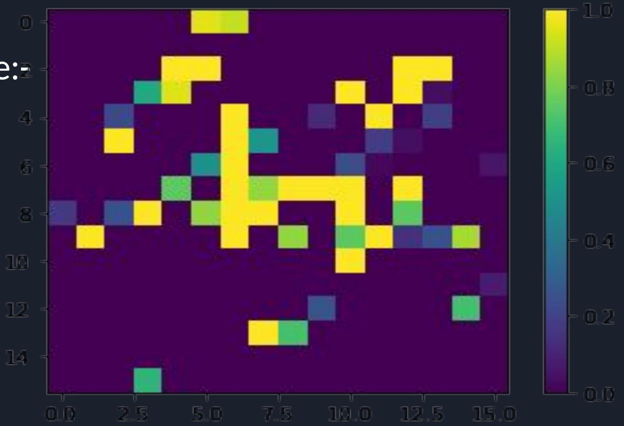
Grayscale to Binary:-

The Grayscale image format is converted into binary by replacing the value with 0 and maximum after doing comparison with threshold value.

Resizing:-

In this project the image matrix is rescaled to 16*16*1.This is done because the images of current resolution are difficult to process using the quantum networks.

After complete preprocessing image looks like:-



Implementation of QNN model

We have followed the following steps to make this model.

1) Load raw data.
2) Preprocess the images.
3) Resizing of the images to fit the data in the quantum network.
4) Remove any contradictory examples.

   This is done to filter the dataset to remove images that are labeled as belonging to both classes.

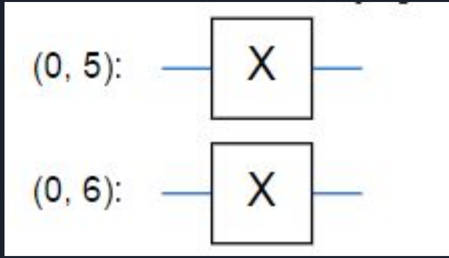
we ran a classical computer as a simulator of a quantum device. For which we used tensorflow quantum and applying following processes

5) Convert binary images to Cirq circuits(using cirq library).

   First we create qubits at every indices according to the size of the image.

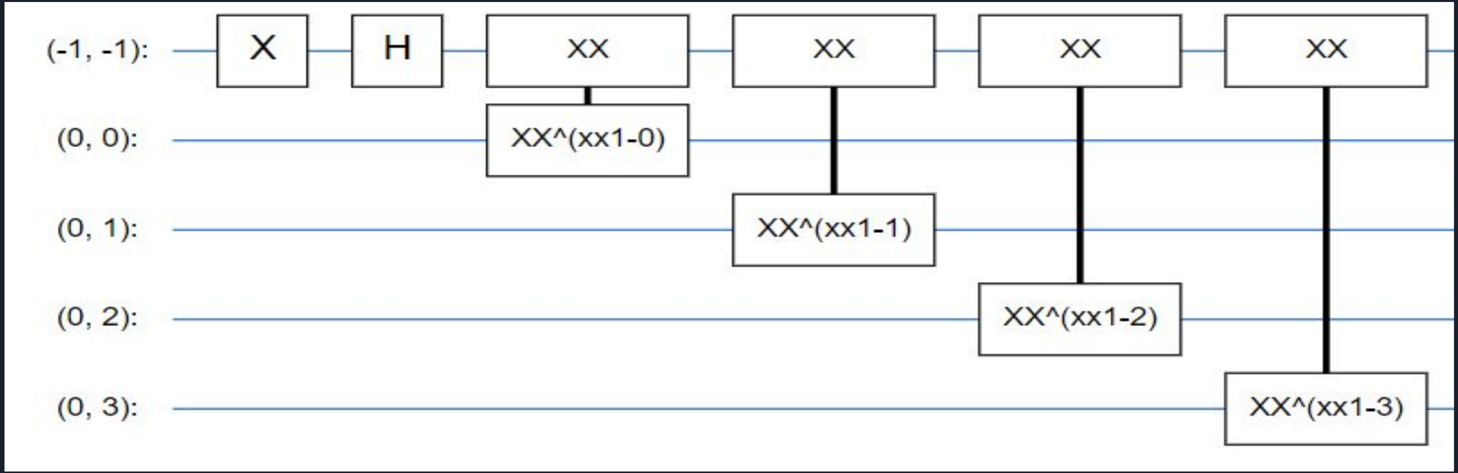
   Our image size is 16*16 so (0,0),(0,1),(0,2)........(15,13),(15,14)(15,15)

   We apply XGate circuit at every indices where value>0.



6) Converts the Cirq circuits to TensorFlow Quantum circuits.

7) Then we add layers using some more random gates to the circuit X, H, ZXX, HH, ZZ.



8) After the circuit design we have used *Parametrized Quantum Circuit* layer, tfq.layers.PQC, to train the model circuit on the quantum data.

Results and Analysis

We divided the dataset in two files one for training data and one for testing data.

The train data had 1648 unique images and test data contain 499 unique images then we apply QNN and CNN both for these datasets and comes with the accuracy as shown in the table.

Model	Accuracy
Quantum Neural Network	0.4918
Convolutional Neural Network	0.4920

```
[31] qnn_history = model.fit(
          x_train_tfcirc_sub, y_train_hinge_sub,
          batch_size=32,
          epochs=EPOCHS,
          verbose=1,
          validation_data=(x_test_tfcirc, y_test_hinge))

     Epoch 1/10
     43/43 [==============================] - 26s 599ms/step - loss: 1.0478 - hinge_accuracy: 0.4743 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 2/10
     43/43 [==============================] - 27s 624ms/step - loss: 1.0478 - hinge_accuracy: 0.4751 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 3/10
     43/43 [==============================] - 26s 611ms/step - loss: 1.0478 - hinge_accuracy: 0.4784 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 4/10
     43/43 [==============================] - 27s 622ms/step - loss: 1.0478 - hinge_accuracy: 0.4760 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 5/10
     43/43 [==============================] - 27s 639ms/step - loss: 1.0478 - hinge_accuracy: 0.4735 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 6/10
     43/43 [==============================] - 27s 626ms/step - loss: 1.0478 - hinge_accuracy: 0.4735 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 7/10
     43/43 [==============================] - 27s 637ms/step - loss: 1.0478 - hinge_accuracy: 0.4768 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 8/10
     43/43 [==============================] - 29s 665ms/step - loss: 1.0478 - hinge_accuracy: 0.4751 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 9/10
     43/43 [==============================] - 27s 617ms/step - loss: 1.0478 - hinge_accuracy: 0.4768 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918
     Epoch 10/10
     43/43 [==============================] - 27s 623ms/step - loss: 1.0478 - hinge_accuracy: 0.4751 - val_loss: 1.0100 - val_hinge_accuracy: 0.4918

[32] qnn_results = model.evaluate(x_test_tfcirc, test_y)

     16/16 [==============================] - 4s 243ms/step - loss: 1.0100 - hinge_accuracy: 0.4918
```

```
Epoch 1/5
52/52 [==============================] - 136s 3s/step - loss: 0.6932 - accuracy: 0.5185 - val_loss: 0.6931 - val_accuracy: 0.5040
Epoch 2/5
52/52 [==============================] - 136s 3s/step - loss: 0.6931 - accuracy: 0.4870 - val_loss: 0.6931 - val_accuracy: 0.5040
Epoch 3/5
52/52 [==============================] - 137s 3s/step - loss: 0.6931 - accuracy: 0.4748 - val_loss: 0.6931 - val_accuracy: 0.4960
Epoch 4/5
52/52 [==============================] - 136s 3s/step - loss: 0.6931 - accuracy: 0.5130 - val_loss: 0.6931 - val_accuracy: 0.5040
Epoch 5/5
52/52 [==============================] - 137s 3s/step - loss: 0.6931 - accuracy: 0.5155 - val_loss: 0.6931 - val_accuracy: 0.4920
<tensorflow.python.keras.callbacks.History at 0x7fa6337017b8>
```

This shows that even for small dataset using Quantum Neural Network instead of CNN there is no major change in accuracy, but the processing of the images in terms of time and space utilization efficiency has considerably increased in case of quantum neural networks which leads our study to conclude that quantum in future would be able to successfully solve classically intractable problems and would exponentially increase the capacity of neural networks in terms of data storage and parallelism.

Our dataset contain 20000 images but it was not possible to run the model with that much amount of data so we took 3500 images in training dataset and applied the QNN model and got the accuracy of 0.5083

```
qnn_history = model.fit(
      x_train_tfcirc_sub, y_train_hinge_sub,
      batch_size=32,
      epochs=EPOCHS,
      verbose=1,
      validation_data=(x_test_tfcirc, y_test_hinge))
```

```
Epoch 1/5
110/110 [==============================] - 59s 538ms/step - loss: 0.9994 - hinge_accuracy: 0.5088
Epoch 2/5
110/110 [==============================] - 61s 557ms/step - loss: 0.9994 - hinge_accuracy: 0.5088
Epoch 3/5
110/110 [==============================] - 60s 544ms/step - loss: 0.9994 - hinge_accuracy: 0.5073
Epoch 4/5
110/110 [==============================] - 59s 540ms/step - loss: 0.9994 - hinge_accuracy: 0.5073
Epoch 5/5
110/110 [==============================] - 59s 533ms/step - loss: 0.9994 - hinge_accuracy: 0.5083
```

While CNN was not working well with that much amount of data and comes with an accuracy of 0.5057

```
Epoch 1/5
110/110 [==============================] - 279s 3s/step - loss: 0.6932 - accuracy: 0.5071
Epoch 2/5
110/110 [==============================] - 277s 3s/step - loss: 0.6931 - accuracy: 0.4951
Epoch 3/5
110/110 [==============================] - 276s 3s/step - loss: 0.6931 - accuracy: 0.4989
Epoch 4/5
110/110 [==============================] - 275s 2s/step - loss: 0.6931 - accuracy: 0.5020
Epoch 5/5
110/110 [==============================] - 278s 3s/step - loss: 0.6931 - accuracy: 0.5057
<tensorflow.python.keras.callbacks.History at 0x7fa636026630>
```

This show that CNN is challenging to learn efficiently if the given dimension of data or model becomes too large and Quantum Neural Network (QNN) provides a new solution to a problem to solve with CNN using a quantum computing environment, or a direction to improve the performance of an existing learning model.

Conclusion of the report

Our team has completed the project by executing a 4-phase plan. Though the project has certain limitations due to the limited computation and memory resources, we have explored and tried various feasible techniques and optimizations to improve the project and present an insightful comparative study on Classical and Quantum approaches of machine learning to a real world problem of signature verification. After researching on quantum neural networks for the whole semester we as a team believe that the project has got a lot of scope to broaden in the coming era of quantum computing and in the coming years quantum will be able to solve the problems which could not be solved by the classical methods till now.

Future Scope

Based on our analysis we concluded that Quantum Concepts with machine learning models not only perform better but also faster. However to make Quantum Concepts more practical and available is still under research and will be essential in the future. As we all know data is increasing at an exponential rate in the present world and to process this huge and complex data tradinitional methods may not be as useful. We will require new methods, softwares and hardwares to keep up with the data that we produce and consume. In the near future we will have gate model quantum computers with a sufficient number of qubits and sufficiently high gate fidelity to run circuits with enough depth to perform tasks that cannot be simulated on classical computers.Quantum computing offers a very promising future in this scenario.