# Quantum Convolutional Neural Network vs VQC

Justin Beltran & Siri Stenstrup Hjermind

Find all code, notebooks, models and data at: https://github.com/justindbeltran/QCNN

# QCNN versus VQC: Which Quantum Classifier is better?

Most agree that VQC beats classical neural networks but what about QNN's?

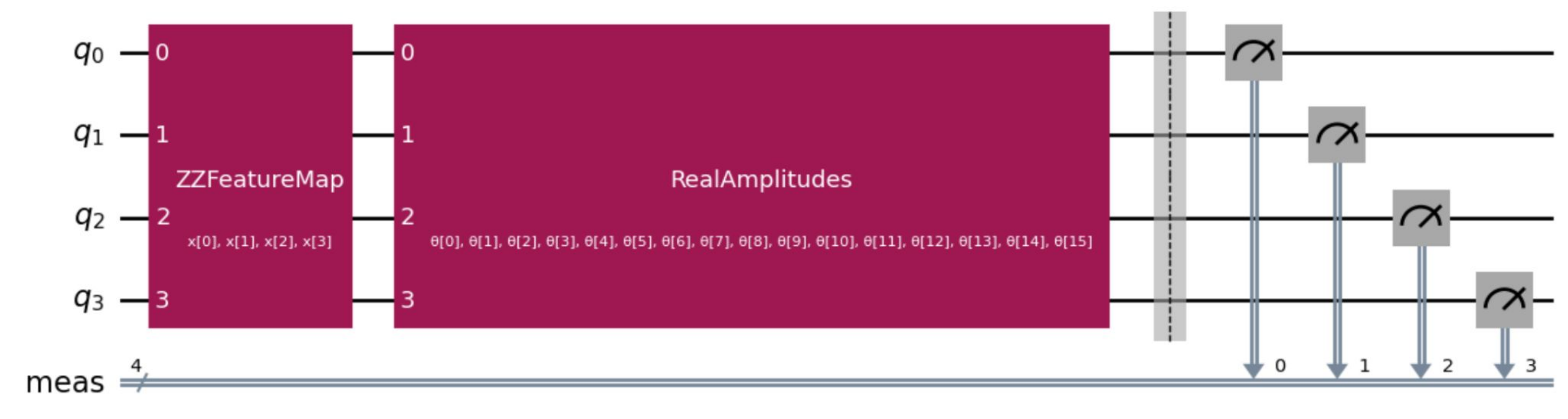
Both utilize quantum machine learning techniques that instantiate quantum circuit with input and weight parameters and then pass to a classical optimizer for updates.

They differ in their underlying circuits and qubit measurement schemes. VQC measures the entirety of the circuit but QCNN measures only the minimum amount of qubits needed for the output space.

**Abstract:** In this work we propose a quantum alternative to Artificial Neural Networks in classification tasks. We design a set of different neural networks and quantum circuits and test their performances. We found that a Variational Quantum Classifier can outperform a classical model using far less free parameters and, thus, being more efficient. Further, a complex classification task requires deeper quantum circuits, which nevertheless grow at a slower pace than the number of neurons needed in a Neural Network for the same task.

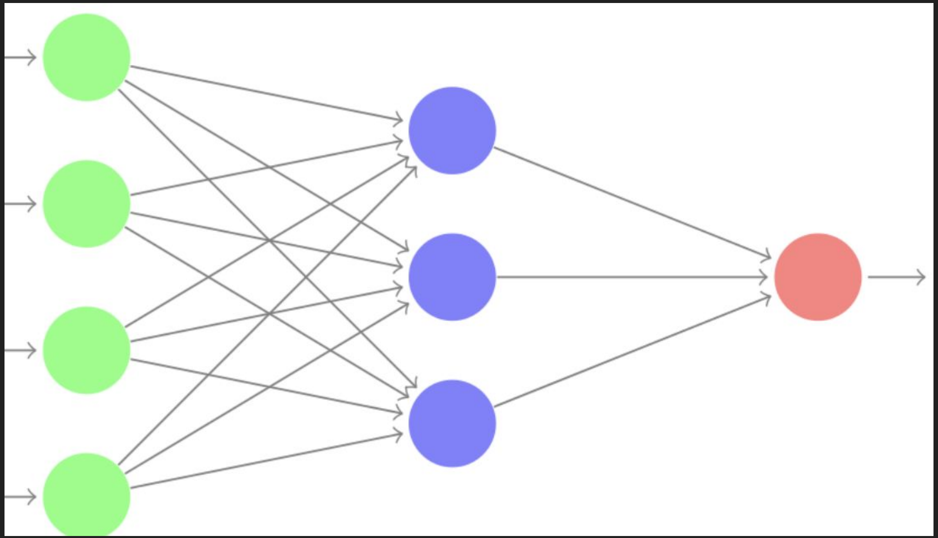
Several theoretical studies have shown that VQCs are more capable than conventional deep neural networks[44–47], in the sense that quantum models train more accurately and/or faster, when compared to classical models of comparable size. Recent results have numerically demonstrated that certain quantum architectures can perform better than their classical counterparts under specific condition.

# VQC - Circuit, Parameters, Measurement



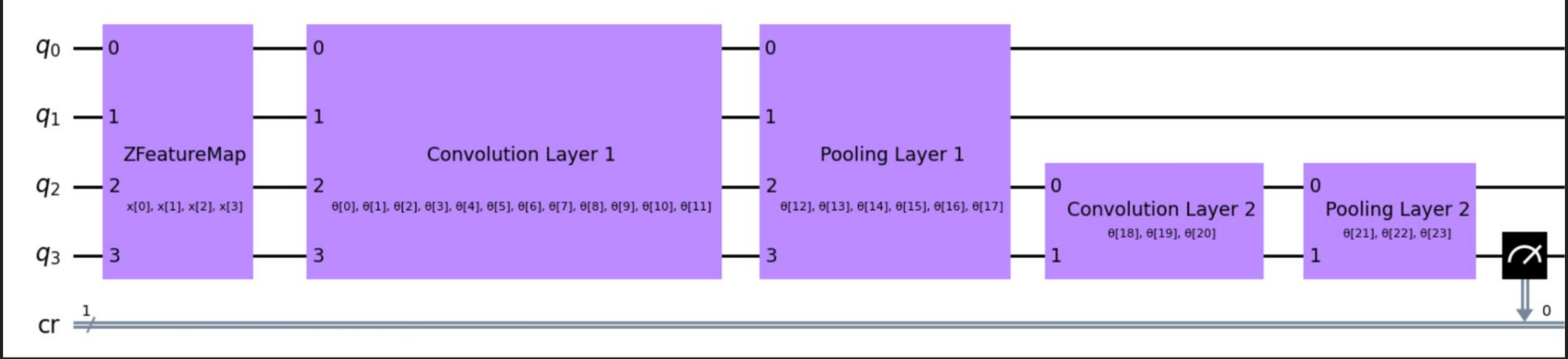

General 4-qubit VQC circuit where an *x* parameter vector is the input for the data set and a *θ* parameter vector for the weights and then the optimizer updates these vectors based on the total measurement returned.

# QCNN - Quantum Convolutional Neural Network



A neural network is inspired by the human brain, it’s based on a series of interconnected nodes, or neurons, organized in a layered structure, with parameters that can be trained by applying deep learning training strategies.

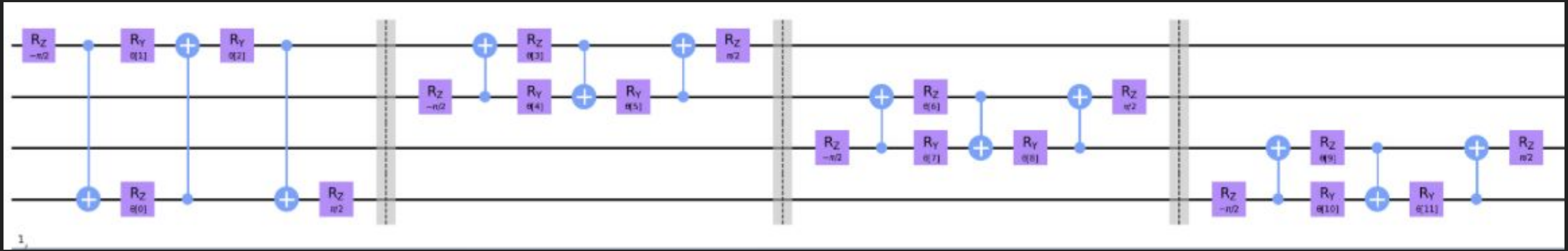
# QCNN - Circuit, Parameters, Measurement



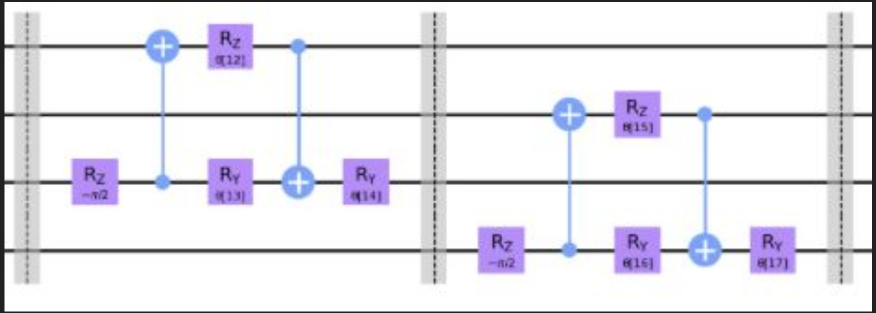

General 4-qubit QCNN circuit where an *x* parameter vector is the input for the data set but the weight parameter vector spans the entirety of every layer, labelled *θ*. The optimizer updates these vectors based only on the return of a single qubit.

# QCNN - Circuit Layers Decomposed

## Convolution Layer for 4 qubits



## Pooling Layer for 4 qubits



# Binary Classification

Binary Classification involves classifying an array of data into 2 possible types.

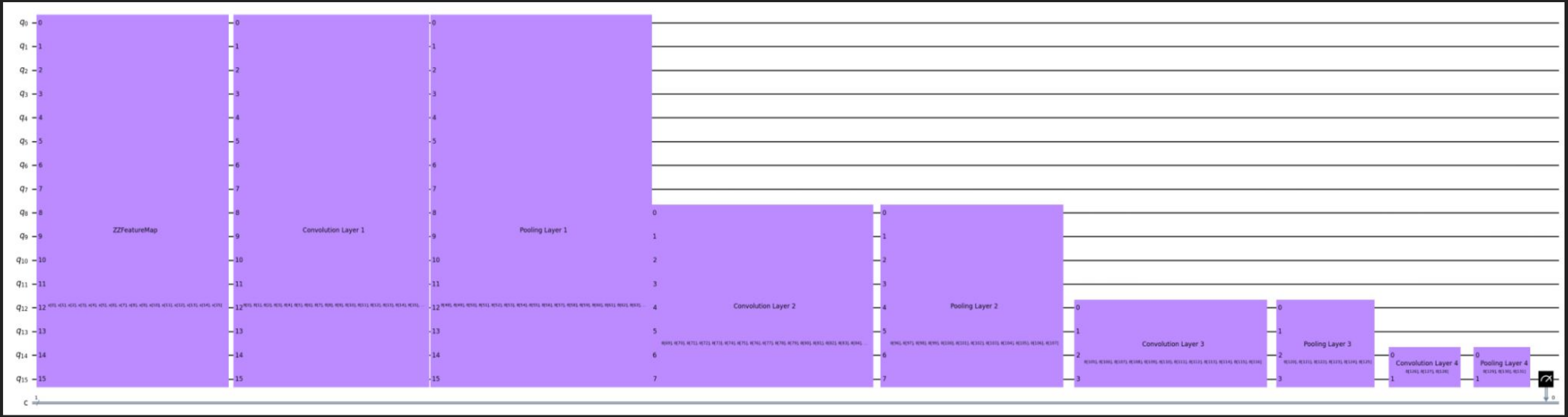
We tested QCNN and VQC on an open source, benchmarking dataset for that consisted of raw Sonar data that consisted of float values and asked them to classify whether a given sonar return was a rock or mine.

The baseline performance of predicting the most prevalent class is a classification accuracy of 53%. The top performer achieved an accuracy of 78%.

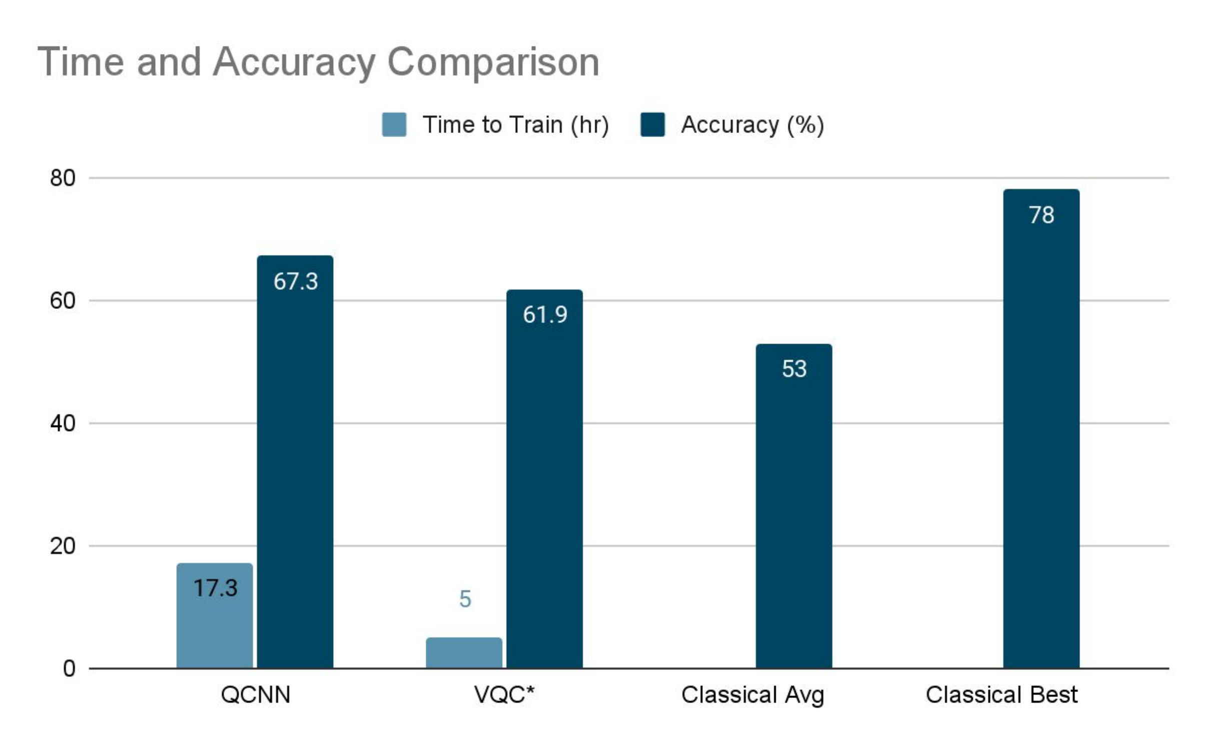
| Data 1 | Data 2 | Data 3 | ... | Data 15 | Data 16 | Ans |
|---|---|---|---|---|---|---|
| 0.0200 | 0.0371 | 0.0428 | ... | 0.0660 | 0.2273 | R |
| 0.0453 | 0.0523 | 0.0843 | ... | 0.7464 | 0.9444 | R |
| 0.0262 | 0.0582 | 0.1099 | ... | 0.6479 | 0.6931 | R |
| 0.0100 | 0.0171 | 0.0623 | ... | 0.1729 | 0.2131 | R |
| 0.0762 | 0.0666 | 0.0481 | ... | 0.4528 | 0.5326 | R |
| ... | ... | ... | ... | ... | ... | ... |
| 0.0187 | 0.0346 | 0.0168 | ... | 0.1801 | 0.2200 | M |
| 0.0323 | 0.0101 | 0.0298 | ... | 0.1235 | 0.1534 | M |
| 0.0522 | 0.0437 | 0.0180 | ... | 0.0683 | 0.1503 | M |
| 0.0303 | 0.0353 | 0.0490 | ... | 0.0673 | 0.1444 | M |
| 0.0260 | 0.0363 | 0.0136 | ... | 0.1302 | 0.1708 | M |

# Binary Classification - Circuits Used

Below is the actual circuit used for Binary Classification. A key difference is that a ZZFeatureMap was implemented over a ZFeatureMap to increase accuracy. The VQC default circuits were left as is because homework 10 showed that those typically achieved the best convergence.



# Binary Classification - Results




Here QCNN and VQC performed better than the classical average perhaps with more time to run they could outperformed the best classical model.

*VQC ran for over 24hrs (and is still currently running) a smaller 5 hour max model was saved and used here.

# Multi-Type Classification

Multi-Type Classification involves classifying an array of data into more than 2 possible types.

We tested QCNN and VQC on an open source, benchmarking dataset for multi-type classification that consisted of X-Ray data comparing different types of seeds.

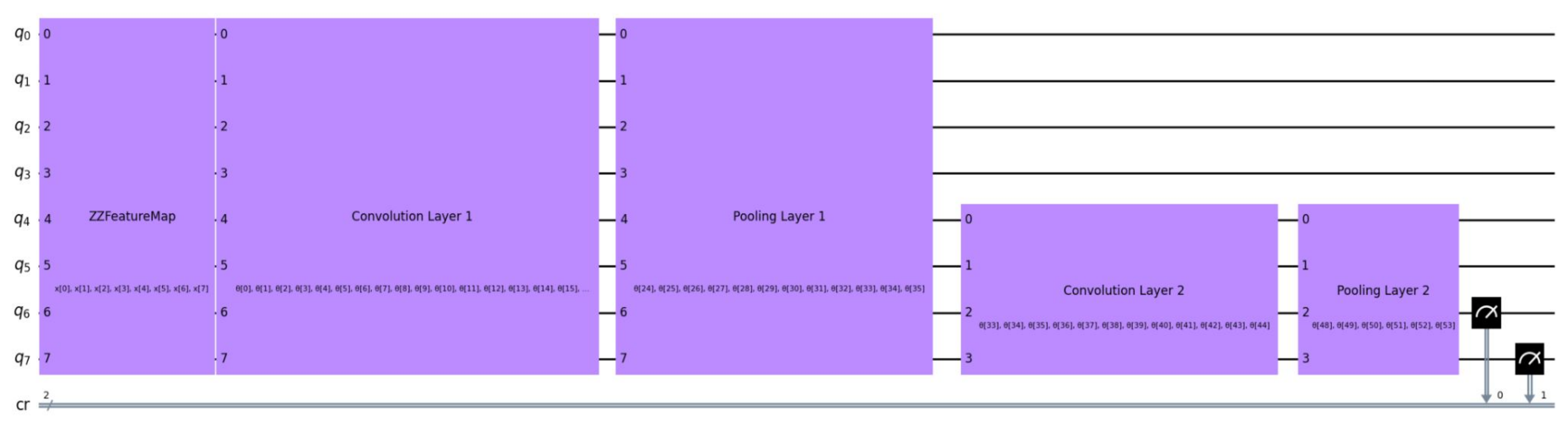
The data set also provided a baseline performance of prediction with the most prevalent class having a classification accuracy of 28%.

| Area | Perime | Density | Length | Width | Coeff | Groove | Class |
|---|---|---|---|---|---|---|---|
| 15.26 | 14.84 | 0.871 | 5.763 | 3.312 | 2.221 | 5.22 | 1 |
| 14.88 | 14.57 | 0.8811 | 5.554 | 3.333 | 1.018 | 4.956 | 1 |
| 14.29 | 14.09 | 0.905 | 5.291 | 3.337 | 2.699 | 4.825 | 1 |
| 13.84 | 13.94 | 0.8955 | 5.324 | 3.379 | 2.259 | 4.805 | 1 |
| 16.14 | 14.99 | 0.9034 | 5.658 | 3.562 | 1.355 | 5.175 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 12.19 | 13.2 | 0.8783 | 5.137 | 2.981 | 3.631 | 4.87 | 3 |
| 11.23 | 12.88 | 0.8511 | 5.14 | 2.795 | 4.325 | 5.003 | 3 |
| 13.2 | 13.66 | 0.8883 | 5.236 | 3.232 | 8.315 | 5.056 | 3 |
| 11.84 | 13.21 | 0.8521 | 5.175 | 2.836 | 3.598 | 5.044 | 3 |
| 12.3 | 13.34 | 0.8684 | 5.243 | 2.974 | 5.637 | 5.063 | 3 |

Sample of Dataset Used

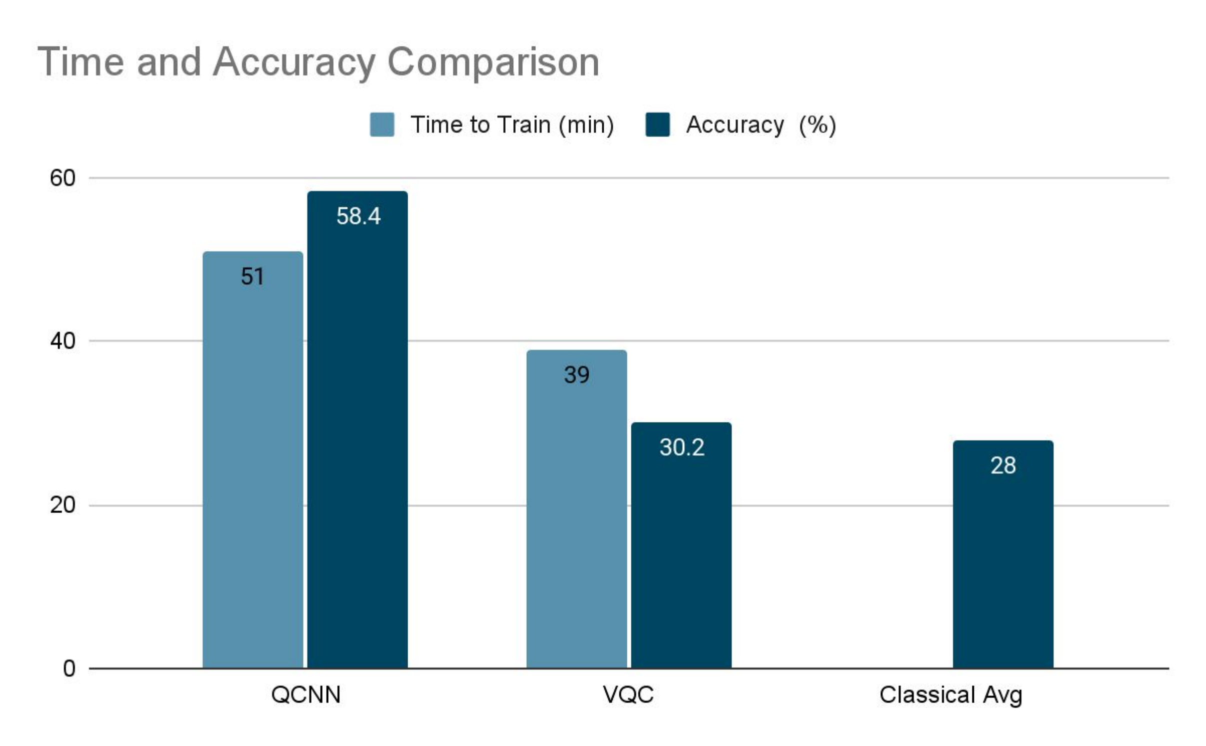
# Multi-Type Classification - QCNN Circuit Modifications

From the dataset there are 7 features and 3 possible outputs. This implementation of a QCNN halves each convolution layer so both the input and output must be in $2^n$ form so instantiate 8 qubits with 2 output qubits to represent [0,3] output values.




Note: VQC does not have these requirements thus it did not need modifications.

# Multi-Type Classification - Results




The QCNN greatly outperformed VQC and the average classical case in terms of accuracy at the cost of more time.

Classical Avg time to train is not relevant for comparison nor was it provided.

# Difficulties and Takeaways

- The main advantages of QCNN and VQC come from training on a quantum computer as we have repeatedly seen they are too noisy in their current state to train on. This resulted in some incredibly lengthy simulations, some of which crashed computers during the process as the width of data grew.
- QCNN gains its advantage over VQC when the space of inputs is smaller than the space of outputs allowing for more convolution layers to be created. Where even 8 inputs to 2 outputs QCNN gained significant advantage.
- Implementing a ZZFeatureMap for QCNN resulted in higher accuracy for binary classification but a lower accuracy for multi-type.

Find all code, notebooks, models and data at: [https://github.com/justindbeltran/QCNN](https://github.com/justindbeltran/QCNN)

# Areas of Improvement or Exploration

- While not tested here, modifying the shrinkage rate of QCNN, such as 3:1 or 4:1, of the convolutional circuit may reduce time of computation at the cost of accuracy and vice versa.
- VQC has two integrated repeatable circuits. They are its feature map, default of ZZFeatureMap, and ansatz, default of RealAmplitudes. The repetition of these circuits can be set when composing them. By increasing their length beyond the default of *reps*=2, a better accuracy at the cost of time could be achieved. This could help it to outperform QCNN potentially.

Find all code, notebooks, models and data at: https://github.com/justindbeltran/QCNN

# Conclusion

While some claim that Variation Quantum Classification will outperform Classical Convolutional Neural Networks, the concept of a convolutional neural network continues. Implementing a Quantum CNN clearly has some advantages over its classical counterpart as well as VQC. As noise control and error correction improves quantum machine learning will too.

Find all code, notebooks, models and data at: https://github.com/justindbeltran/QCNN

# References:


- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- https://qiskit-community.github.io/qiskit-machine-learning/tutorials/11_quantum_convolutional_neural_networks.html
- https://qiskit-community.github.io/qiskit-machine-learning/tutorials/09_saving_and_loading_models.html
- https://machinelearningmastery.com/standard-machine-learning-datasets/
- https://diposit.ub.edu/dspace/bitstream/2445/140318/1/GIL%20FUSTER%20Elies%20Miquel.pdf
- https://doi.org/10.1038/s41598-022-24082-z


Find all code, notebooks, models and data at: https://github.com/justindbeltran/QCNN