Evaluating the Performance of Hybrid Quantum-Classical Convolutional Neural Networks on NISQ Devices

Tristan Nagan and Ritesh Ajoodha

The University of the Witwatersrand, Johannesburg, South Africa,
1484020@students.wits.ac.za,
ritesh.ajoodha@wits.ac.za,
WWW home page: https://www.wits.ac.za/

Abstract. The increase in quantum algorithm development in recent years has spurred the development of new algorithms which out source classically intractable portions of an algorithm to a quantum computer. These hybrid quantum-classical algorithms must be thoroughly studied and analyzed to determine their viability. One such algorithm is the hybrid quantum-classical convolutional neural network.
In this paper, we compared different implementations of hybrid quantum-classical convolutional neural networks and their fully classical counterparts. In particular, we compared their accuracy and loss performance over time. To do this, we implemented both the classical and hybrid quantum-classical algorithms with varying convolutional layers and ran them on a cluster of classical computers and IBM's cloud quantum computers using the low-dimensional Tetris dataset. We found that hybrid quantum-classical convolutional neural networks do not outperform their classical counterparts in terms of accuracy and loss over time when tested on current-state quantum computers, which conflicts with past work. We can attribute the discrepancy to the noise present in current quantum computers affecting the results of the circuit.

Keywords: Machine Learning, Quantum Computing, Convolutional Neural Networks.

1 Introduction

The recent improvements in quantum computing hardware have sparked a surge of funding into the research and development of quantum algorithms. The eventual goal of these algorithms is to one day be used to solve real-world problems in chemistry, finance, and machine learning. The problem is that we do not know how different variations of these algorithms perform when compared to each other and their classical counterparts in terms of accuracy and loss over time. So, the question we must ask is, what is the difference in accuracy and loss over time between hybrid quantum-classical convolutional neural networks and a classical convolutional neural network?

In this paper, we perform an empirical analysis of hybrid quantum-classical convolutional neural networks on current quantum computing hardware and compare them to their fully classical counterparts. The goal is to determine whether these algorithms show any potential of outperforming their classical counterparts (regarding accuracy and loss mentioned earlier), and if so, under what conditions.

The first approaches to hybrid quantum convolutional neural networks [1], [2] only implemented the convolutional layer on a quantum computer. The rest of the network architecture is on a classical computer. Data passes from the input layer on the classical computer to a quantum computer where the convolution takes place. Once the convolution runs to completion, the data passes from the quantum computer back to the classical computer for pooling. The classical computer then passes the data back to the quantum computer to perform the convolution operation again, or the data can continue to the fully connected layer or a classical convolutional layer.

They proposed a convolutional layer made of a variational quantum circuit [5]. This circuit encodes the input data onto several qubits and then performs either a random quantum circuit [1] or a structured circuit [2] on them before taking a measurement. Both approaches are implementable on current quantum hardware and are scalable.

This paper focuses on the random and structured circuit approaches for quantum convolution. We compare the accuracy and loss performance of these approaches against each other and a classical convolutional neural network. We implemented the classical and hybrid quantum-classical algorithms and ran them on a cluster of classical computers and the IBM cloud quantum computers and test them on a low-dimensional synthetic dataset known as the Tetris dataset.

We found that hybrid quantum-classical convolutional neural networks do not out-perform their purely classical counterparts when tested on current-state quantum computers. This conflicts with the results obtained in previous research, which showed hybrid quantum-classical convolutional neural networks out-performing their classical counterparts. We attribute this to the use of ideal quantum simulators in the training and testing of models in previous work, instead of current-state quantum computers.

This research will make the following contributions: (a) an analysis of the accuracy and loss over time for hybrid quantum-classical convolutional neural networks with a varying number of convolutional layers and varying number of labels in the training and testing data, (b) an analysis of the structure of these quantum convolutional layers which will determine why a structure or random quantum circuit outperforms the other, (c) an explanation of the use of a quantum circuit as a generalized feature map and how it may extract different high-level spatial features then a classical convolutional layer, (d) an analysis which highlights the flaws in these algorithms, if any, and hopefully leads to the development of better algorithms which address these flaws.

Having introduced the problem area and briefly outlined the literature, methodology, and results, the rest of this work will provide a more detailed discussion on each of these topics. Section 2 will provide context to this research by giving a more in-depth discussion on the related literature within this field. Section 3 covers the methodology used in this research. Finally, we will present our results along with a discussion and conclusion.

2 Related Work

2.1 Quantum Convolutional Neural Networks

The first quantum algorithm to adapt the classical convolutional neural network (CNN) was developed to solve quantum many-body problems which are complex systems that are too difficult to solve theoretically [3]. Past attempts to solve these systems involved using classical machine learning algorithms. This provided mixed levels of success thus [3] proposed this algorithm to address the shortcomings of past approaches by taking the convolutional and pooling layers from a classical CNN and implementing them in the quantum space to reduce the complexities of these problems. This algorithm has two main problems: first, it has no image processing uses like a classical CNN; and second, to solve a problem of significant complexity it requires more qubits than are feasible on current quantum hardware.

The second approach to quantum CNNs was developed with image processing in mind [4]. This algorithm is a complete translation of the classical CNN into the quantum realm with convolutional layers, pooling layers, an activation function and a fully connected layer with backpropagation. This approach has two main problems: first, is the need for quantum RAM (QRAM) which has no reliable implementation; and second, the number of qubits needed to implement this algorithm efficiently on current hardware being too high.

The authors did, however, run a numerical simulation of their algorithm against a classical CNN of similar architecture and found that in terms of speed the quantum CNN could run significantly faster than its classical counterpart and provide a similar accuracy.

To address the problems of scalability for current quantum hardware and bypass the need for QRAM, researchers turned to hybrid approaches using variational quantum circuits to replace only the convolutional layer of an otherwise classical CNN. Thus the first hybrid algorithm was proposed [1]. The authors outline a quantum analogue for the classical convolutional layer which they call a "quanvolutional" layer. The purpose of this layer is the same as the classical convolutional layer because it extracts high-level spatial-features from the input image. The layer consists of a variational quantum circuit (VQC) which encodes window data on $n \times n$ qubits (where n represents the window size), performing a random quantum circuit on these qubits and then measuring them to produce a matrix.

The simplest use of this layer is replacing the convolutional layer with the quantum analogue in a CNN composed of a single convolutional layer. We can extend this use by replacing 1 out of multiple convolutional layers with this quanvolutional layer, replacing multiple layers with quanvolutional layers in varying orders or replacing the convolutional layers entirely with quanvolutional layers. The latter options put considerable strain on current hardware, and an increase in layer numbers and layer complexity may hit a limitation in hardware. This, however, is still more practically feasible then a full quantum algorithm.

The second hybrid algorithm proposed [2] is fairly similar to the first one. The key difference between them is that latter focused exclusively on a structured circuit for their VQC.

When simulated, and compared to its classical counterparts the quantum CNN consistently outperformed the classical CNN across several layer architectures with the single-layer quantum CNN outperforming a single layer CNN by almost 60% in accuracy and 25% in loss for five label classification and even outperformed a two-layer CNN by 20% in accuracy and 10% in loss for five label classification. This may show that a structured quantum circuit may provide an advantage over a random quantum circuit. This contradicts the initial findings of [1], who found that there is no difference in performance between a random and structured quantum circuit.

3 Methodology

We will now cover the methodology used in this work. We will start by going over the data used to train and test our models. Then we will cover the structure of the models and the quantum circuits. Following this, we will cover the metrics used to evaluate the performance of our models. Finally, we will cover the version design platforms and environments we used to code and run our models.

3.1 Data

The data used in this research is a synthesized Tetris dataset. It comprises 3×3 grayscale images resembling the blocks in the popular arcade game Tetris. There are five basic shapes that can be in a variety of orientations. We chose this dataset for its low dimensionality. A bonus of using this synthesized dataset is that we can synthesize the amount of data we need. The goal of the convolutional neural network is to classify an image as one of the five (or two) shapes regardless of its orientation.

3.2 Models

The classical and hybrid quantum-classical models share the same basic network structure as seen in figure 1.

The 2×2 window in the image passes to the convolutional layer (in figure 1, the layer is quantum, but it can also be classical) and then moves to the pooling layer. We can repeat the convolutional and pooling layers. The output of the pooling layer is flattened and passed to the fully-connected layer, where we take the SoftMax to determine the model prediction.

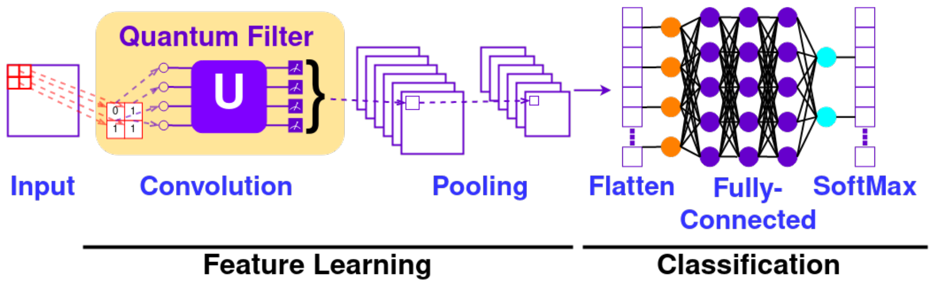

Fig. 1. An overview of the convolutional neural network structure.

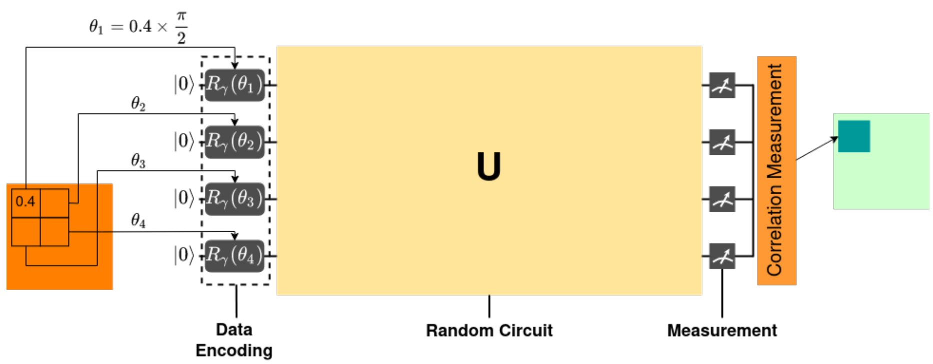

Fig. 2. The random quantum convolutional layer.

The Random convolutional layer is shown in figure 2. The 2×2 window in the image is encoded onto the qubits using an R_y gate. This result passes to the random circuit and is then measured. This result moves to the pooling layer, and the pipeline continues, as shown in figure 1.

The Structured convolutional layer is shown in figure 3. The 2×2 window in the image is encoded onto the qubits using an R_y gate similar to the random quantum convolutional layer. This result is then passed to the first layer of single-qubit gates (which is another R_y gate) and then passed to the first layer of two-qubit gates (which are interlaced CNOT gates). This structure of single-qubit gates, followed by two-qubit gates, is repeated N times (where N is $n \times n$ and n is the number of qubits).

In [1], the random circuit weights were not trainable. Instead, the random circuit is simply a non-linear transformation of the input data. In this work, we trained the weights of the random circuit to see if this affects the accuracy of the model.

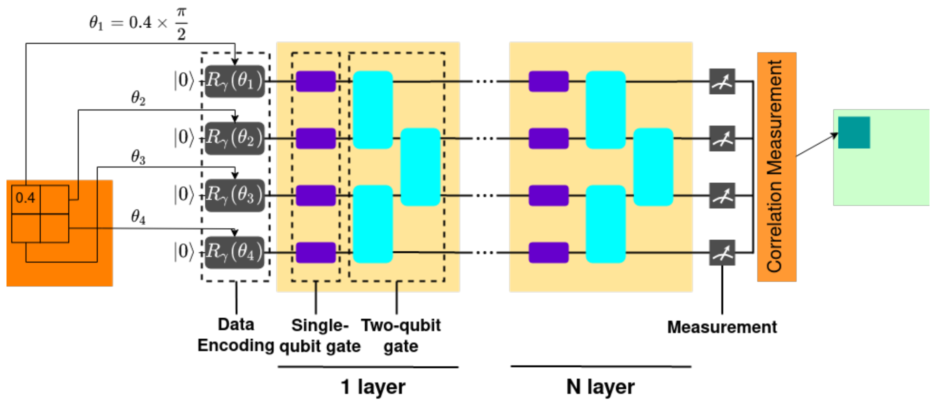

Fig. 3. The structured quantum convolutional layer.

3.3 Metrics

To keep testing consistent with previous studies, we measured the change in accuracy and loss of our convolutional neural networks.

3.4 Design Platform and Environments

The classical and hybrid quantum-classical convolutional neural network algorithms were implemented in Python using Pytorch for the Deep Learning pipeline and IBM's open-source framework for quantum algorithm development, Qiskit. We integrated them using Pennylane, a cross-platform python library for quantum machine learning.

We trained the models on a classical computer through a Quantum Simulator. We did this to reduce training time as each convolution operation is done sequentially and could sit on a queue for up to 15 minutes before being run on an IBM quantum computer. For each image, the algorithm must perform four convolutions, and in the training dataset, there are 804 images for five label classification. Meaning, the algorithm must perform 3216 convolution operations, which could each take 15 minutes in the worst case. This high number of convolution operations, paired with calculating the derivative of the weights of the quantum circuit during backpropagation, increases the training time significantly. We tested the models on IBM's five qubit Ourense Quantum Computer via the IBM Quantum Experience. To ensure that the simulator used to train the models was as close to a real quantum computer, we applied a noise model to the simulator. The noise model was taken from IBM's five qubit Ourense Quantum Computer so that our training and testing environments would be as similar as possible.

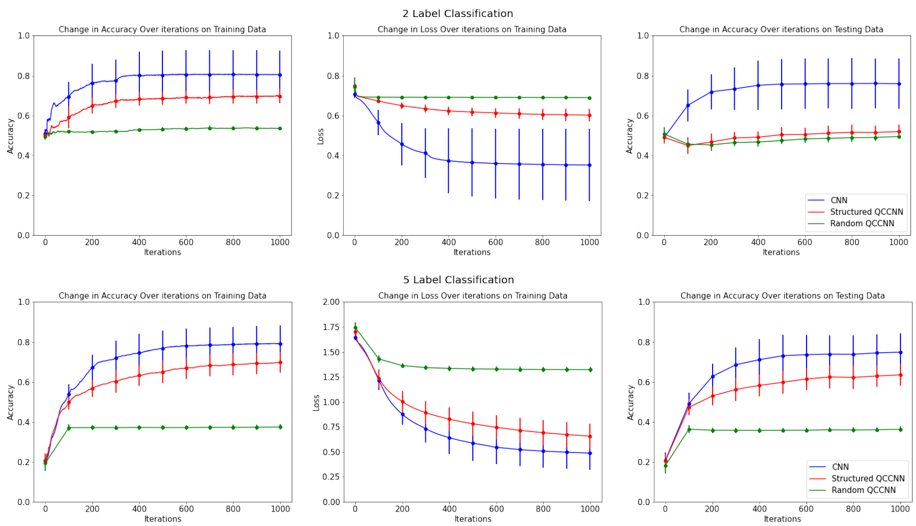

Fig. 4. Results for 2 label and 5 label classification in terms of accuracy and loss on training and testing data over time.

4 Results and Discussion

We show the results for two label and five label classification in terms of accuracy and loss on the training and testing data over time in figure 4. From the results, we see that the Classical CNN consistently outperformed both the structured and random QCCNNs, conflicting with the results obtained in previous work [1], [2]. This discrepancy between the results obtained in this work and the results obtained in previous work may be due to two factors:

– The simulators used in previous works may have been analogues for perfect quantum computers (i.e. quantum computers without noise). In this work, we made an effort to ensure that the simulator used to train the models was as close to a real current-state quantum computer as possible.
– The results in previous works may have been obtained by both training and testing the models in a quantum simulator.

The results obtained in previous work may be considered obtainable in an ideal situation, where quantum computers are devoid of noise. The results obtained in this work are a reflection of how noise in current-state quantum computers can negatively impact results. Our results confirm that the structured QCCNN does indeed out-perform the random QCCNN, but we found that it out-performed it by more than expected. This outcome could be due to making the weights of the random circuit trainable since the number of trainable weights within the circuit may not be sufficient to train our model, and this would cause underfitting. Increasing the number of trainable parameters in both the random and structured quantum circuits may improve the accuracy of these models.

5 Conclusion

In this paper, we compared different implementations of hybrid quantum-classical convolutional neural networks and their fully classical counterparts. In particular, we compared their accuracy and loss performance over time. Our results directly conflicted with the results found in previous work outlining these algorithms. We determined that this discrepancy can be attributed to the use of ideal quantum simulators and training and testing their models in these ideal quantum simulators instead of current-state quantum computers in previous work. In this work, we opted to train our models in a quantum simulator with a noise model in place to more accurately simulate the conditions found in a real quantum computer where we would test these models. Our results show how noise can impact results and skew them away from the results found in ideal situations. Implementing a quantum convolution circuit that is robust against noise may be a possible avenue for future work.

6 Acknowledgments

This work is based on the research supported in part by the National Research foundation of South Africa (Grant number:121835)

References

1. Henderson, M., SHakya, S., Pradhan, S., Cook, T.: Quanvolutional neural networks: powering image recognition with quantum circuits. Quantum Machine Intelligence vol. 2, pages 1-9, 2020.
2. Liu, J., Lim, K. H., Wood, K. L., Huang, W., Guo, C., Huang, H.: Hybrid Quantum-Classical Convolutional Neural Networks. *arXiv preprint arXiv:1911.02998*, 2019, https://arxiv.org/abs/1911.02998
3. Cong, I., Choi, S., Lukin, M.: Quantum Convolutional Neural Network. Nature Physics vol. 15, pages 1273-1278, 2019.
4. Kerenidis, I., Landman, J., Prakash, A.: Quantum Algorithms for Deeo Convolutional Neural Networks. International Conference on Learning Representations, 2020.
5. Wilson, C. M., Otterbach, J. S., Tezak, N., Smith, R. S., PollorenoA. M., Karalekas, Peter J., Heidel, S., Sohaib Alam, M., Crooks, G. E., da Silva, M. P.: Quantum Kitchen Sinks: An algorithm for machine learning on near-term quantum computers. *arXiv preprint arXiv:1806.08321*, 2018, https://arxiv.org/pdf/1806.08321.pdf