Evaluating Accuracy and Robustness of Quanvolutional Neural Networks

Korn Sooksatra

August 8, 2021

Abstract

It cannot be denied that machine learning can provide benefits for many areas and applications. Further with the rise of quantum computing, machine learning algorithms have begun to be implemented in a quantum environment, and these are called quantum machine learning. There are several attempts to implement deep learning in quantum computers. Nevertheless, they were not fully successful. Then, a convolutional neural network combined with an additional quantum layer was discovered and called quanvolutional neural network (QNN). A QNN has shown a higher performance over a classical CNN. In this work, we aim to evaluate accuracy, loss values and adversarial robustness of QNNs compared to CNNs. As a result, QNNs could achieve better accuracy and loss values than the classical ones and also show their robustness against adversarial examples generated their classical versions. Our code and its supplementary files are available in `https://github.com/kornsook/Quanvolutional-neural-network`.

1 Introduction

With the rise of deep learning on these days, significant improvements of many areas have been noticed, such as computer visions, natural language processing and autonomous cars. Since its performance is still not satisfying in some applications, there still exist some works to increase its accuracy. Further, deep learning has been known to be vulnerable to adversarial examples discovered in [10, 5]. Thus, we aim to address those problems of deep learning.

After the improvement of quantum computing recently, Schuld *et al.* [9] implemented machine learning algorithms in quantum computers and introduced them as quantum machine learning. Later, there are several existing works [4, 3, 2, 8] that attempted to create deep learning models in quantum computing. Moreover, Henderson *et al.* [6] designed a hybrid neural networks composed of a quantum layer and classical layers and called it quanvolutional neural networks.

Inspired by the idea in [6], we desire to evaluate if quanvolutional neural networks (QNN) are more accurate and more robust than classical convolutional neural networks (CNN). Therefore, the contributions of this report can be summarized as follows:

- We discuss the idea of QNN and propose our QNN.
- We extensively construct experiments to evaluate the performance and robustness of QNNs.

2 Quanvolutional neural networks (QNN)

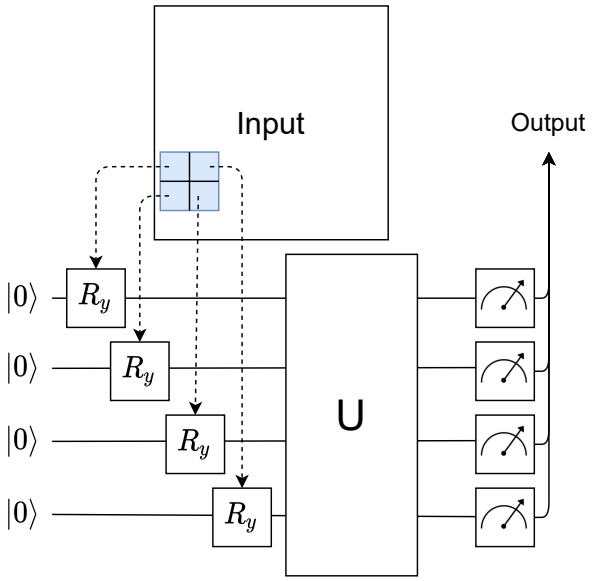


Figure 1: A filter of quanvolutional layer

These networks were first proposed in [6] and shown that they could improve the performance of classical convolutional neural networks (CNN). In general, a QNN consists of two parts: a quanvolutional layer and CNN. Specifically, the idea of this is to add one more quanvolutional layer as the first layer of CNN, and this layer is simply a quantum circuit that mimics a convolutional layer. One circuit in a quantum layer is a filter; hence, if we desire to have n filters in the layer, we need n circuits. The number of qubits in each circuit is proportional to the size of kernel. For example, if the size of kernel is 2x2, then the number of qubits is $2 \cdot 2 = 4$. Further, in a circuit, the initial states of the qubits are $|0\rangle$, and those states first pass R_y gates whose phases are π multiplied with the values from the kernel in the input. Then, all outputs of R_y gates go to a U opertor which is a random circuit consiting of several one-qubit and two-qubit

gates. After that, all the outputs of the U operation are measured and then result in an output. This method and circuit can be illustrated in Figure 1.

After the quantum layer, we obtain the output that has n channels where n is the number of filters. Therefore, as being noticed, it is very similar to a convolutional layer. It simply applies a quantum circuit instead of a convlutional operation used in a convlutional layer. At last, this output is fed to CNN.

3 Experimental design

We constructed these experiments on Google Colab and used Python version 3 mainly with Pennylane module for implementing quantum layers and Tensorflow module for implementing machine learning models (i.e., CNNs). We use MNIST dataset [7] as our input, and this dataset is images of digits (i.e., zero to nine). Our models need to classify these images into the digits. To be fair, we trained all models for 30 epochs with the same training dataset consisting of 50 images and test them with the same 30 images. All the models are described as follows:

1. **Linear model**. This model has only the output layer which has ten neurons for the outputs of ten digits.

2. **Quantum linear model**. The first layer of this model is a quanvolutional layer, and the other part is the linear model.

3. **CNN model**. The first layer of this model is a convlutional layer with 3 filters. The kernel size and stride of each filter are 3x3, and the rest of the model is the linear model.

4. **Quantum CNN model**. This is a quanvolutional neural network, and its architecture is the same as the CNN model with an additional quanvolutional layer in the front of it.

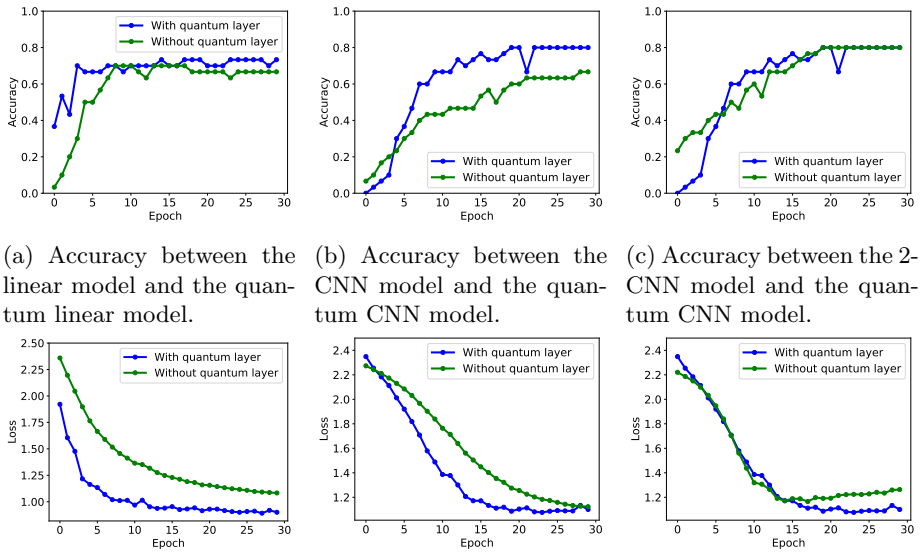
5. **2-CNN model**. This model has two convolutional layers, and the first one has 5 filters with 3x3 kernel size and 1x1 stride. The rest of the model is the CNN model.

It is worth noting that our quanvolutional layer has 5 filters with 3x3 kernel and 1x1 stride, and we compute the average of the results in each filter as its output. Noticeably, the setting of the quanvolutional layer is the same as the first convolutional layer in the 2-CNN model because we aim to test if a quanvolutional layer can outperform a convolutional layer. All the experiments are listed as follows:

1. Comparison with respect to accuracy and loss values between 1) the linear model and the quantum linear model, 2) the CNN model and the quantum CNN model and 3) the 2-CNN model and the quantum CNN model during training.

2. Evaluation of robustness of all models with adversarial examples generated from the linear model, CNN model and 2-CNN model. Note that we chose Carlini and Wagner attack (CWA) [1] to find adversarial examples with 0.01 step size.

4 Results



(a) Accuracy between the linear model and the quantum linear model.

(b) Accuracy between the CNN model and the quantum CNN model.

(c) Accuracy between the 2-CNN model and the quantum CNN model.

(d) Loss values between the linear model and the quantum linear model.

(e) Loss values between the CNN model and the quantum CNN model.

(f) Loss values between the 2-CNN model and the quantum CNN model.

Figure 2: Comparison of accuracy and loss values between (a, d) the linear model and the quantum linear model, (b, e) the CNN model and the quantum CNN model and (c, f) the 2-CNN model and the quantum CNN model respectively.

First, we compared the linear model to the quantum linear model and found that a linear model with an additional quantum layer achieved higher accuracy and lower loss values than the one without the quantum layer as demonstrated in Figure 2a and 2d respectively. Further, we found the same explicit results when we compared the CNN model to the quantum CNN model as seen in Figure 2b and 2e. However, those results are not surprising since the quantum linear model and quantum CNN model respectively have more layers than the linear model and CNN models. Then, we checked the comparison between the quantum CNN model and the 2-CNN model and surprisingly discovered that the quantum CNN model achieved lower loss values than the 2-CNN model although they achieved the same accuracy as shown in Figure 2c and 2f.

In addition, we tested all the models in term of adversarial robustness against

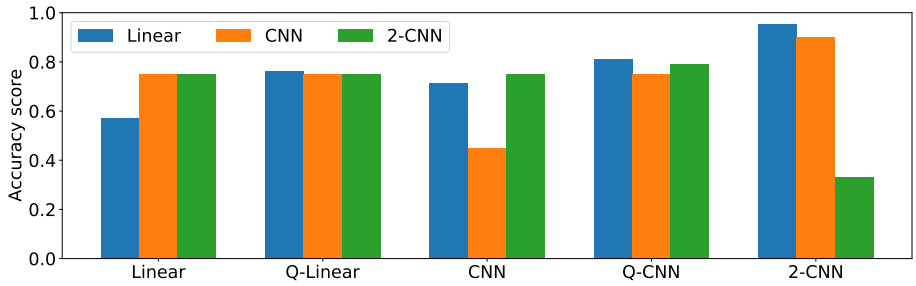


Figure 3: Accuracy scores achieved by the models with adversarial examples generated from the linear model (blue), the CNN model (orange) and the 2-CNN model (green). Note that Q-Linear and Q-CNN are respectively the quantum linear model and quantum CNN model.

CWA which was a strong attack. According to Figure 3, the quantum linear model was more robust than the linear model, and the quantum CNN model was more robust than the CNN model. Therefore, adding a quantum layer could improve robustness of a model because it filters small perturbation generated by CWA out of an input. Furthermore, we found an interesting phenomenon with the 2-CNN model. It could achieve significantly high accuracy scores with adversarial examples generated by the linear and CNN models.

5 Discussion and Conclusion

A quanvolutional neural network (QNN) is a convolutional neural network (CNN) with an additional quantum layer composed of n quantum circuits for n filters. We observed that adding a quantum layer to a model could increase its accuracy and reduce its loss value. Also, when we substituted a convolutional layer with a quantum layer (in the case of the quantum CNN model and 2-CNN model), we noticed that they achieved the same accuracy; nonetheless, a model with a quantum layer could achieve lower loss value than the one with a convolutional layer.

In addition to accuracy and loss value, adversarial examples created from one model could hardly transfer to that model with a quantum layer. Thus, a neural network can benefit from a quantum layer. Explicitly, in this work, we only evaluated robustness of QNNs from adversarial generated from classical ones. We leave evaluation of adversarial robustness of QNNs with adversarial examples generated from QNNs for the future works.

References

[1] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*,

pages 39–57. IEEE, 2017.

[2] Iris Cong, Soonwon Choi, and Mikhail D Lukin. Quantum convolutional neural networks. *Nature Physics*, 15(12):1273–1278, 2019.

[3] Pierre-Luc Dallaire-Demers and Nathan Killoran. Quantum generative adversarial networks. *Physical Review A*, 98(1):012324, 2018.

[4] Edward Farhi, Hartmut Neven, et al. Classification with quantum neural networks on near term processors. *Quantum Review Letters*, 1(2 (2020)):10–37686, 2020.

[5] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[6] Maxwell Henderson, Samriddhi Shakya, Shashindra Pradhan, and Tristan Cook. Quanvolutional neural networks: powering image recognition with quantum circuits. *Quantum Machine Intelligence*, 2(1):1–9, 2020.

[7] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[8] Seunghyeok Oh, Jaeho Choi, and Joongheon Kim. A tutorial on quantum convolutional neural networks (qcnn). In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 236–239. IEEE, 2020.

[9] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[10] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014*, 2014.