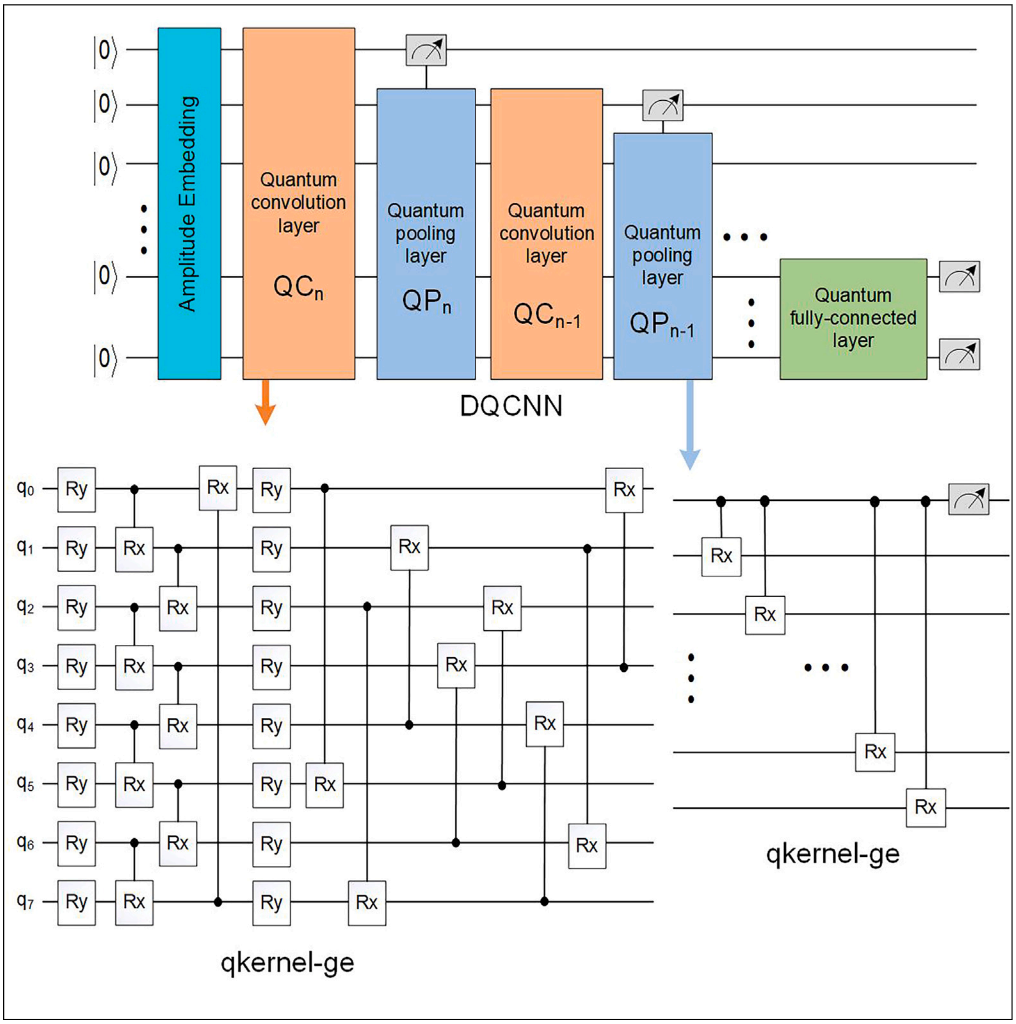
iScience

CellPress
OPEN ACCESS

Article

A degressive quantum convolutional neural network for quantum state classification and code recognition



Qingshan Wu, Wenjie Liu, Yong Huang, Haoyang Liu, Hao Xiao, Zixian Li

wenjiel@163.com

Highlights

Propose a decreasing convolutional neural network model for classification task

Remove the parameter sharing strategy and design a quantum kernel with global eyesight

Design a decreasing quantum pooling layer to avoid large data reduction

Comparison of multiple classification experiments with ATQCNN and HQCNN

Wu et al., iScience 27, 109394
April 19, 2024
https://doi.org/10.1016/j.isci.2024.109394

iScience

CellPress
OPEN ACCESS

Article

A degressive quantum convolutional neural network for quantum state classification and code recognition

Qingshan Wu,[1] Wenjie Liu,[1,2,3,6,*] Yong Huang,[4] Haoyang Liu,[1] Hao Xiao,[5] and Zixian Li[1]

SUMMARY

With the rapid development of quantum computing, a variety of quantum convolutional neural networks (QCNNs) are proposed. However, only $1/2^{\frac{n}{2}}$ features of an *n*-qubits input are transferred to the next layer in a quantum pooling layer, which results in the accuracy reduction. To solve this problem, a QCNN with a degressive circuit is proposed. In order to enhance the ability of extracting global features, we remove the parameters sharing strategy in the quantum convolutional layer and design a quantum convolutional kernel with global eyesight. In addition, to prevent a sharp feature reduction, a degressive parameterized quantum circuit is adopted to construct the pooling layer. Then the *Z*-basis measurement is only performed on the first qubit to control the operations on other qubits. Compared with the state-of-the-art QCNN, i.e., hybrid quantum-classical convolutional neural network, the accuracy of our model increased by 0.9%, 1%, and 3%, respectively, in three tasks: quantum state classification, binary code recognition, and quaternary code recognition.

INTRODUCTION

Quantum computing can offer more efficient solutions than classical computing to solve problems.[1,2] Algorithm complexity may be largely reduced by utilizing specific quantum algorithms. For example, Shor algorithm[3] reduces the time complexity of integer decomposition from 2^n to n; the search complexity of unordered databases has been reduced from N to $\sqrt{N}$ through quantum search algorithms.[4,5] With the rapid development of machine learning and quantum devices, training a quantum neural network (QNN) by combining classical computer and parameterized quantum circuit (PQC), i.e., variational quantum algorithms (VQAs), has attracted more attention. So far, a large number of VQAs have been proposed, such as variational quantum eigensolver, (VQE)[6–8] which is applied to compute the ground states of molecular systems; quantum approximate optimization algorithm (QAOA) [9–11] solves the problem of max-cut and number partitioning; quantum variational error corrector (QVECTOR)[12,13] finds an alternative method for quantum error correction which reduces the number of qubits available on NISQ (noisy intermediate-scale quantum) devices; quantum generative adversarial networks (QGANs)[14–17] focus on generating various fake data for expanding the database by using QNN. Quantum graph neural networks (QGNNs)[18–20] can extract features from non-Euclidean data. Quantum recurrent neural networks (QRNNs)[21–23] are very effective for data with sequence characteristics; it can extract the sequential features and semantic information in the data. Quantum convolutional neural networks (QCNNs)[24–26] can be applied in feature compression and data recognition. In addition, there is also research on the topologies of PQC in terms of expressibility, entangling capability and circuit cost.[27–29]

In a variety of VQAs, our research primarily focuses on QCNN, which is one of the most commonly employed models. In 2019, the concept of a new purely quantum model was first introduced by Cong et al., which is also known as a QCNN.[24] QCNN possesses the ability to precisely identify quantum states associated with topological phases. This capability can be leveraged to create an optimized quantum error correction scheme, surpassing the effectiveness of current methods. It was proved that there is absence of barren plateaus in the circuit structure of QCNN in 2021.[30] In the past several years, a plenty of different QCNN frameworks were proposed for data classification or edge detection.[31–33] These examples demonstrate the immense potential of QCNN in addressing practical applications.

[1]School of Software, Nanjing University of Information Science and Technology, No. 219, Ning Liu Road, Nanjing, Jiangsu 210044, China
[2]Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, No. 219, Ning Liu Road, Nanjing, Jiangsu 210044, China
[3]Jiangsu Province Engineering Research Center of Advanced Computing and Intelligent Services, Nanjing University of Information Science and Technology, No. 219, Ning Liu Road, Nanjing, Jiangsu 210044, China
[4]Anhui Province Key Laboratory of Atmospheric Sciences and Satellite Remote Sensing, Anhui Institute of Meteorological Sciences, No.16, Shi He Road, Hefei, Anhui 230002, China
[5]School of Information Engineering, Huzhou University, No.759, Second Ring East Road, Huzhou, Zhejiang 313002, China
[6]Lead contact
*Correspondence: wenjiel@163.com
https://doi.org/10.1016/j.isci.2024.109394

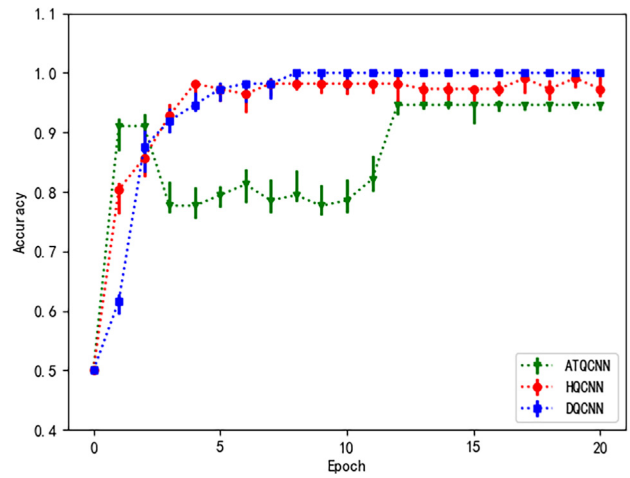


Figure 1. The accuracy of DQCNN, ATQCNN, and HQCNN on quantum state classification

However, there are still some problems in current QCNN. For example, global features are difficult to extract because convolutional kernel only acts on adjacent qubits. On the other hand, quantum pooling layers reduce the number of qubits by half, resulting in large-scale feature reduction. The low expressibility of QCNN, i.e., the low accuracy in data classification, is caused by these problems. In order to solve these problems, we propose a degressive quantum convolutional neural network (DQCNN). The restrictions on quantum kernel parameter sharing are removed, and a new quantum convolutional kernel with global eyesight (qkernel-ge) is designed to replace the origin quantum convolutional layer. Qkernel-ge in fact acts as a feature extractor with a global view. Instead of reducing the number of qubits by half, the quantum pooling layer with star topology abandons half features by measuring the first qubit, which effectively avoids massive feature reduction caused by halving the qubits. Based on the experiments of quantum state classification and code recognition, the accuracy of DQCNN is verified to be higher than that of the state-of-the-art QCNN.

RESULTS

The performance of DQCNN is verified according to two supervised learning tasks, quantum state binary classification and code recognition (binary and four classes). PennyLane and TensorFlow are selected as the framework of quantum machine learning. The loss value is calculated by *SparseCategoricalCrossentropy*, and the Adam gradient descent algorithm is chosen as the optimizer. The experiments are implemented on a Linux server with two Xeon(R) Gold 5220R CPUs, 256 GB RAM, and two A100 GPUs; the software environment is python 3.9. ATQCNN and hybrid quantum-classical convolutional neural network proposed by google,[34] HQCNN, were selected as the comparison objects. All experiment codes and data will be public and available at GitHub[35] after this paper is accepted.

Quantum state classification

In order to test the classification capability of DQCNN on quantum data, 560 quantum cluster states are randomly generated and excited to form a dataset, where 80% of the dataset are used to training, and the remaining data act as a tester. The label values of quantum cluster state are set to 0 and 1. The batch size, the learning rate, and the total training epoch are set to 64, 0.025, and 20, respectively.

DQCNN and ATQCNN are 7-qubits models. There are 6 quantum convolutional and pooling layers in DQCNN, and ATQCNN repeats the quantum convolutional and pooling layer three times. HQCNN consists of three 7-qubits components and a classical fully connected layer, and there is one quantum convolutional and pooling layer in each component. The final results are shown in Figure 1.

It can be seen from Figure 1 that the test accuracy of DQCNN, ATQCNN, and HQCNN reaches 100%, 94.6%, and 99.1%, respectively. Although DQCNN was slightly weaker than the other two in the early stages, DQCNN converged to 100% accuracy with the fastest speed. In the meantime, the convergence curve of ATQCNN is very oscillating, while that of DQCNN is more moderate. The experiment on quantum state classification shows that DQCNN has higher performance than ATQCNN and HQCNN on quantum data.

Code recognition

To explore the applicability and performance of DQCNN on classical data, new malicious code recognition experiments are performed. CIC-MalDroid2020[36] is selected as our experiment dataset. CICMalDroid2020 consists of five categories of code, the first four of which are different kinds of malicious code, and the fifth is benign code.

In the paper of ATQCNN, there are no published experimental data and code. For the sake of fair comparison, the original code feature values provided by the website are chosen as the dataset[37]; the dataset contains 139 extracted features for 11,598 APK files comprising frequencies of system calls. The data are sorted according to frequency of occurrence, and the 128 most common features are retained.

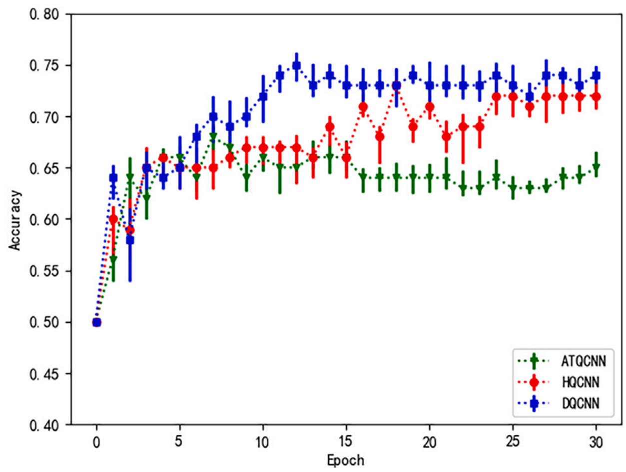


Figure 2. The accuracy of DQCNN, ATQCNN, and HQCNN on malicious code recognition

Recognition between malicious and benign codes

In this experiment, 250 adware malicious and 250 benign codes were selected as the dataset; the label values are set to 0 and 1. After the dataset is randomly shuffled, 400 data act as the training set and 100 data act as the test set. The final test accuracy of DQCNN, ATQCNN, and CNN is shown in Figure 2, where the learning rate, the batch size, and the total training epoch are set to 0.025, 100, and 30, respectively.

DQCNN, ATQCNN, and HQCNN achieved accuracy rates of 74%, 68%, and 73%, respectively. Obviously, the performance of DQCNN is much higher than that of ATQCNN; DQCNN is not only more accurate but also less volatile. Although the highest accuracy of HQCNN is close to that of DQCNN, DQCNN converges faster and smoother than HQCNN. Therefore, we believe that DQCNN has the best performance in malicious code recognition.

Recognition between four categories of malicious codes

In order to further explore the ability of DQCNN to identify complex problems, an experiment to distinguish four types of malicious code is performed. The dataset consists of 250 pieces of each malicious code (1,000 data), where 80% are for training and 20% are for testing; the label values of adware, Banking, SMS malware, and Riskware are respectively set to 0, 1, 2, and 3. The final test accuracy of DQCNN, ATQCNN, and HQCNN is shown in Figure 3, where the learning rate is set to 0.025. The batch size is 100, and the total training epoch increase to 40.

The accuracy rates of DQCNN, ATQCNN, and HQCNN are 57%, 44%, and 54%, respectively. From Figure 3, it can be seen that the accuracy of DQCNN and HQCNN is significantly ahead of that of ATQCNN. The overall accuracy of DQCNN is higher than that of HQCNN although the accuracy is slightly lower than that of HQCNN at a few epochs.

DISCUSSION

In this article, we propose a DQCNN. Amplitude encoding is applied to encode the data for saving the number of qubits. DQCNN is more sensitive to global features because the convolutional layer of DQCNN removes the restriction that the qkernels only act on adjacent qubits. The pooling layer with single qubit measurement strategy is intended to solve the problem of serious feature data loss. Compared to the original quantum pooling layer, the pooling layer of DQCNN reduces the amount of data from N to $N/2$ instead of from N to $\sqrt{N}$ and it is more friendly for feature selection. Although the parameters of the convolutional layer and the pooling layer of DQCNN rise from O(1) to O(n), the experimental results prove that the performance of DQCNN is demonstrated at rising cost.

We also found that there is an exploratory space in DQCNN; the probability of barren plateaus increases with a rise in model complexity. There may be more suitable parameter optimization strategies, such as layer-wise training strategies, to decrease the difficulty of optimizing DQCNN. On the other hand, more application areas of DQCNN are yet to be explored. DQCNN still has great potential in unsupervised learning tasks or dimensionality reduction, such as quantum clustering algorithm, quantum autoencoder, quantum principal-component analysis algorithm, etc. In the face of special data structures, such as temporal data or graph data, how to renovate DQCNN is also one of our future studies.

Limitations of the study

There are some limitations in our study. The limitations of these improvements include three key aspects. Firstly, in the future, it is necessary to study new topologies of quantum pooling layers, mainly exploring ways to reduce circuit depth. Currently, the depth of pooling layer circuits is dependent on the number of qubits. Secondly, more efforts are needed to delve deeper into the relationship between data size and feature reduction rate, as the demand for feature reduction rate varies with the increase of problem size. To achieve this goal, we plan to use high-dimensional data and use different feature reduction rates to reconstruct the model and study the stride configuration of DQCNN. This extended analysis aims to explore the generalization of DQCNN. In addition, as DQCNN can currently

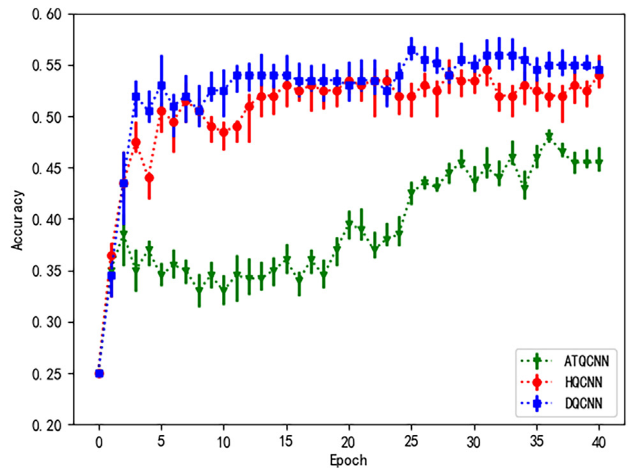


Figure 3. The accuracy of DQCNN and ATQCNN in the recognition of 4 categories of malicious code

be viewed as a black box as a whole, it is necessary to explore its interpretability, and this study is expected to further elucidate the efficiency of DQCNN.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Materials availability
  - Data and code availability
- EXPERIMENTAL MODEL AND SUBJECT DETAILS
- METHOD DETAILS
- QUANTIFICATION AND STATISTICAL ANALYSIS

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.isci.2024.109394.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (62071240), the Innovation Program for Quantum Science and Technology (2021ZD0302901), the Natural Science Foundation of Jiangsu Province (BK20231142 and BK20220804), the Postgraduate Research & Practice Innovation Program of Jiangsu Province (KYCX23_1370), and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

AUTHOR CONTRIBUTIONS

Q.W. and W.L. designed this study. Q.W., Y.H., and H.L. analyzed and drafted the manuscript. Q.W., H.L., and H.X. completed numerical experiments. Z.L. and W.L. have revised the manuscript. All authors were involved in explaining the concept and results of the data. All authors have reviewed and approved the final version of the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: November 17, 2023
Revised: January 29, 2024
Accepted: February 28, 2024
Published: March 4, 2024

REFERENCES

1. Barends, R., Kelly, J., Megrant, A., Veitia, A., Sank, D., Jeffrey, E., White, T.C., Mutus, J., Fowler, A.G., Campbell, B., et al. (2014). Superconducting quantum circuits at the surface code threshold for fault tolerance. Nature *508*, 500–503. https://doi.org/10.1038/nature13171.
2. Debnath, S., Linke, N.M., Figgatt, C., Landsman, K.A., Wright, K., and Monroe, C. (2016). Demonstration of a small programmable quantum computer with atomic qubits. Nature *536*, 63–66. https://doi.org/10.1038/nature18648.
3. Shor, P.W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Rev. *41*, 303–332. https://doi.org/10.1137/S0036144598347011.
4. Grover, L.K. (1996). A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 212–219. https://doi.org/10.1145/237814.237866.
5. Liu, W., Wu, Q., Shen, J., Zhao, J., Zidan, M., and Tong, L. (2021). An optimized quantum minimum searching algorithm with sure-success probability and its experiment simulation with Cirq. J. Ambient Intell. Humaniz. Comput. *12*, 10425–10434. https://doi.org/10.1007/s12652-020-02840-z#citeas.
6. Cao, Y., Romero, J., Olson, J.P., Degroote, M., Johnson, P.D., Kieferová, M., Kivlichan, I.D., Menke, T., Peropadre, B., Sawaya, N.P.D., et al. (2019). Quantum chemistry in the age of quantum computing. Chem. Rev. *119*, 10856–10915. https://doi.org/10.1021/acs.chemrev.8b00803.
7. Hastings, M.B., Wecker, D., Bauer, B., and Troyer, M. (2014). Improving Quantum Algorithms for Quantum Chemistry. Preprint at arXiv. https://doi.org/10.48550/arXiv.1403.1539.
8. McClean, J.R., Romero, J., Babbush, R., and Aspuru-Guzik, A. (2016). The theory of variational hybrid quantum-classical algorithms. New J. Phys. *18*, 023023. https://doi.org/10.1088/1367-2630/18/2/023023.
9. Farhi, E., Goldstone, J., and Gutmann, S. (2014). A Quantum Approximate Optimization Algorithm. Preprint at arXiv. https://doi.org/10.48550/arXiv.1411.4028.
10. Farhi, E., and Harrow, A.W. (2016). Quantum Supremacy through the Quantum Approximate Optimization Algorithm. Preprint at arXiv. https://doi.org/10.48550/arXiv.1602.07674.
11. Zhou, L., Wang, S.T., Choi, S., Pichler, H., and Lukin, M.D. (2020). Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. Phys. Rev. X *10*, 021067. https://doi.org/10.1103/PhysRevX.10.021067.
12. Johnson, P.D., Romero, J., Olson, J., Cao, Y., and Aspuru-Guzik, A. (2017). QVECTOR: An Algorithm for Device-Tailored Quantum Error Correction. Preprint at arXiv. https://doi.org/10.48550/arXiv.1711.02249.
13. Nautrup, H.P., Delfosse, N., Dunjko, V., Briegel, H.J., and Friis, N. (2019). Optimizing quantum error correction codes with reinforcement learning. Quantum *3*, 215. https://doi.org/10.22331/q-2019-12-16-215.
14. Dallaire-Demers, P.L., and Killoran, N. (2018). Quantum generative adversarial networks. Phys. Rev. *98*, 012324. https://doi.org/10.1103/PhysRevA.98.012324.
15. Lloyd, S., and Weedbrook, C. (2018). Quantum generative adversarial learning. Phys. Rev. Lett. *121*, 040502. https://doi.org/10.1103/PhysRevLett.121.040502.
16. Liu, w.j., Zhao, J.J., Zhang, Y., and Ge, Y.B. (2022). A Quantum Conditional Generative Adversarial Network Algorithm. Acta Electron. Sin. *50*, 1586. https://doi.org/10.12263/DZXB.20210512.
17. Liu, W., Zhao, J., and Wu, Q. (2022). A hybrid quantum-classical generative adversarial networks algorithm based on inherited layerwise learning with circle-connectivity circuit. Quantum Inf. Process. *21*, 372. https://doi.org/10.1007/s11128-022-03719-y.
18. Verdon, G., McCourt, T., Luzhnica, E., Singh, V., Leichenauer, S., and Hidary, J. (2019). Quantum Graph Neural Networks. Preprint at arXiv. https://doi.org/10.48550/arXiv.1909.12264.
19. Ai, X., Zhang, Z., Sun, L., Yan, J., and Hancock, E. (2022). Decompositional Quantum Graph Neural Network. Preprint at arXiv. https://doi.org/10.48550/arXiv.2201.05158.
20. Tüysüz, C., Rieger, C., Novotny, K., Demirköz, B., Dobos, D., Potamianos, K., Vallecorsa, S., Vlimant, J.R., and Forster, R. (2021). Hybrid quantum classical graph neural networks for particle track reconstruction. Quantum Mach. Intell. *3*, 29. https://doi.org/10.1007/s42484-021-00055-9#citeas.
21. Takaki, Y., Mitarai, K., Negoro, M., Fujii, K., and Kitagawa, M. (2021). Learning temporal data with a variational quantum recurrent neural network. Phys. Rev. *103*, 052414. https://doi.org/10.1103/PhysRevA.103.052414.
22. Bondarenko, D., Salzmann, R., and Schmiesing, V.S. (2023). Learning Quantum Processes with Memory–Quantum Recurrent Neural Networks. Preprint at arXiv. https://doi.org/10.48550/arXiv.2301.08167.
23. Chen, S.Y.C. (2023). Quantum deep recurrent reinforcement learning. In ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5. https://doi.org/10.1109/ICASSP49357.2023.10096981.
24. Cong, I., Choi, S., and Lukin, M.D. (2019). Quantum convolutional neural networks. Nat. Phys. *15*, 1273–1278. https://doi.org/10.1038/s41567-019-0648-8.
25. Liu, J., Lim, K.H., Wood, K.L., Huang, W., Guo, C., and Huang, H.L. (2021). Hybrid quantum-classical convolutional neural networks. Sci. China Phys. Mech. Astron. *64*, 290311. https://doi.org/10.1007/s11433-021-1734-3.
26. Di, S., Xu, J., Shu, G., Feng, C., Ding, X., and Shan, Z. (2023). Amplitude transformed quantum convolutional neural network. Appl. Intell. *53*, 20863–20873. https://doi.org/10.1016/j.future.2021.07.034.
27. Sim, S., Johnson, P.D., and AspuruGuzik, A. (2019). Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. Adv. Quantum Technol. *2*, 1900070. https://doi.org/10.1002/qute.201900070.
28. Hubregtsen, T., Pichlmeier, J., Stecher, P., and Bertels, K. (2021). Evaluation of parameterized quantum circuits: on the relation between classification accuracy, expressibility, and entangling capability. Quantum Mach. Intell. *3*, 9–19. https://doi.org/10.1007/s42484-021-00038-w.
29. Liu, W.J., and Wu, Q.S. (2023). A Block-Ring Connected Topology of Parameterized Quantum Circuits. Preprint at arXiv. https://doi.org/10.48550/arXiv.2308.10791.
30. Pesah, A., Cerezo, M., Wang, S., Volkoff, T., Sornborger, A.T., and Coles, P.J. (2021). Absence of barren plateaus in quantum convolutional neural networks. Phys. Rev. X *11*, 041011. https://doi.org/10.1103/PhysRevX.11.041011.
31. Li, Y., Zhou, R.G., Xu, R., Luo, J., and Hu, W. (2020). A quantum deep convolutional neural network for image recognition. Quantum Sci. Technol. *5*, 044003. https://doi.org/10.1088/2058-9565/ab9f93/meta.
32. Wei, S., Chen, Y., Zhou, Z., and Long, G. (2022). A quantum convolutional neural network on NISQ devices. AAPPS Bull. *32*, 1–11. https://doi.org/10.1007/s43673-021-00030-3#citeas.
33. Jing, Y., Li, X., Yang, Y., Wu, C., Fu, W., Hu, W., Li, Y., and Xu, H. (2022). RGB image classification with quantum convolutional ansatz. Quantum Inf. Process. *21*, 101. https://doi.org/10.1007/s11128-022-03442-8.
34. Broughton, M., Verdon, G., McCourt, T., Martinez, A.J., Yoo, J.H., Isakov, S.V., Massey, P., Halavati, R., Niu, M.Y., Zlokapa, A., et al. (2021). TensorFlow Quantum: A Software Framework for Quantum Machine Learning. Preprint at arXiv. https://doi.org/10.48550/arXiv.2003.02989.
35. https://github.com/wqs1999/DQCNN.
36. Mahdavifar, S., Alhadidi, D., and Ghorbani, A.A. (2022). Effective and efficient hybrid android malware classification using pseudo-label stacked auto-encoder. J. Netw. Syst. Manage. *30*, 22. https://doi.org/10.1007/s10922-021-09634-4.
37. https://www.unb.ca/cic/datasets/maldroid-2020.html.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and algorithms		
Tensorflow	tensorflow.org	https://www.tensorflow.org/?hl=zh-cn
Python	Python.org	https://docs.python.org/zh-cn/3/
Pennylane	pennylane.ai	https://github.com/PennyLaneAI/pennylane

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact Wenjie Liu (wenjiel@163.com).

Materials availability

This study did not generate new materials.

Data and code availability

- All data reported in this paper will be shared by the lead contact upon request.
- This paper does not report original code.
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

EXPERIMENTAL MODEL AND SUBJECT DETAILS

Our study does not use experimental models typical in the life sciences.

METHOD DETAILS

The method details refers to the DQCNN, it can be seen in the supplemental file named as "METHOD DETAILS: Degressive Quantum Convolutional Neural Network".

QUANTIFICATION AND STATISTICAL ANALYSIS

We apply our DQCNN model to the randomly selected code data. If the algorithm can find the correct labels for the data, the prediction is successful. Therefore, our strategy of quantification and statistical analysis is to calculate the proportion of correctly predicted labels on the test dataset. The accuracy comparson between DQCNN, ATQCNN and HQCNN are given in Figures 1, 2, and 3.