DeepQMLP: A Scalable Quantum-Classical Hybrid Deep Neural Network Architecture for Classification

Mahabubul Alam
Department of Electrical Engineering
Penn State University
mxa890@psu.edu

Swaroop Ghosh
Department of Electrical Engineering
Penn State University
szg212@psu.edu

Abstract—Quantum machine learning (QML) is promising for potential speedups and improvements in conventional machine learning (ML) tasks (e.g., classification/regression). The search for ideal QML models is an active research field. This includes identification of efficient classical-to-quantum data encoding scheme, construction of parametric quantum circuits (PQC) with optimal expressivity and entanglement capability, and efficient output decoding scheme to minimize the required number of measurements, to name a few. However, most of the empirical/numerical studies lack a clear path towards scalability. Any potential benefit observed in a simulated environment may diminish in practical applications due to the limitations of noisy quantum hardware (e.g., under decoherence, gate-errors, and crosstalk). We present a scalable quantum-classical hybrid deep neural network (DeepQMLP) architecture inspired by classical deep neural architecture. In DeepQMLP, stacked shallow Quantum Neural Network (QNN) models mimic the hidden layers of a classical feed-forward multi-layer perceptron network. Each QNN layer produces a new and potentially rich representation of the input data for the next layer. This new representation can be tuned by the parameters of the circuit. Shallow QNN models experience less decoherence, gate errors, etc. which make them (and the network) more resilient to quantum noise. We present numerical studies on a variety of classification problems to show the trainability of DeepQMLP. We also show that DeepQMLP performs reasonably well on unseen data and exhibits greater resilience to noise over QNN models that use a deep quantum circuit. DeepQMLP provided up to 25.3% lower loss and 7.92% an even higher ability to approximate functions, QNN holds great potential for the future.

A conventional QNN architecture is shown in Fig. 1(b). In a typical QNN model, the input data is encoded into a quantum state using a suitable encoding scheme (e.g., angle encoding, amplitude encoding, etc.) [9]. The encoding is followed by layers of PQC with tunable parameters (w1,w2,... in Fig. 1(b)). These parameters are analogous to weights in a classical neural network. In the end, the output quantum state of the QNN is measured (sampled) on the appropriate basis (e.g., the default Pauli-Z measurement basis in IBM quantum computers [10]). The sampling process is repeated many times with the same parameters. A cost function is derived from the measurements. A classical optimizer (e.g., gradient-descent) updates the parameter values to minimize the cost.

The choice of the PQC can have a significant impact on the performance (e.g., trainability) of a QNN model. For instance, deep PQC with lots of parameters may be desirable for learning but may experience vanishing gradient problems (also referred to as barren plateaus) making it harder for the gradient-based optimizers to navigate through the solution space [11]. Moreover, quantum computers are plagued with various noise sources such as gate error, readout error, decoherence, and crosstalk [12]. The output quantum state can be random (i.e., meaningless) if the noise accumulation is high. A large amount of noise can also induce a barren plateau in the QNN solution space [13].

Shallow-depth circuits are preferred for QNN to avoid the aforementioned issues [11], [14]. However, shallow-depth circuits may often be unable to approximate complex functionality (similar to shallow classical neural networks with small number of parameters).

In this article, we propose two new quantum-classical hybrid deep neural network architectures: Quantum Multi-Layer Perceptrons (QMLP) and DeepQMLP to partially address the aforementioned issues. Both architectures are inspired by conventional Multi-Layer Perceptron (MLP) networks used in deep learning. In MLP, multiple layers of neurons are used to define and search through a solution space for a given ML task (Fig. 1(a)). Neurons of successive layers are connected through trainable weights. The first and the last layers of MLP are commonly referred to as input and output layers. The internal layers are referred to as hidden layers. Typically, MLP models contain multiple hidden layers. In QMLP, the hidden layer of an MLP is mimicked by a QNN layer as shown in Fig. 1(c). The QNN takes a quantum encoded representation of the classical data and produces an output representation (e.g., Pauli-Z expectation values of the qubits) which is fed to the classical output layer. The network can be trained with any conventional loss function. However, in this work, we only use cross-entropy loss. In DeepQMLP, multiple shallow-depth QNN models (two used in this work) are used as hidden layers of an MLP (Fig. 1(d)). Each layer produces a new representation for the next layer. For example, the qubit expectation values of the first hidden layer in Fig. 1(d) are used as the inputs to the second hidden layer.
QMLP uses a deep QNN alongside a classical dense output layer to exploit the higher expressive power of QNN. To accommodate a larger search space, the quantum hidden layer in QMLP needs a deep parameterized circuit. However, deep circuits are more error-prone. DeepQMLP addresses the issue by using a series of shallow-depth quantum circuits stacked one after another. The shallower circuits require fewer number of gates and execution time which reduces the accumulation of gate errors and decoherence. Thus, the architecture shows more robustness against noise.

Contributions: We, (a) present two new quantum-classical hybrid neural network architectures (QMLP and DeepQMLP) for classification, (b) show the trainability of the proposed models through numerical studies with 4 synthetic datasets and the ‘iris’ dataset across 78 training runs with varying depth of the parametric quantum circuits, and (c) present an empirical proof-of-concept study to exhibit greater noise resilience of the DeepQMLP architecture.

II. PRELIMINARIES

Qubits and Quantum Gates: Qubit is analogous to classical bits however, a qubit can be in a superposition state i.e., a combination of \(|0\rangle\) and \(|1\rangle\) at the same time. Quantum gates such as single qubit (e.g., Pauli-X (\(\sigma_x\)) gate) or multiple qubit (e.g., 2-qubit CNOT gate) gates modulate the state of qubits. These gates can perform a fixed computation (e.g. an X gate flips a qubit state) or a computation based on a supplied parameter (e.g. the RY(\(\theta\)) gate rotates the qubit along the Y-axis by \(\theta\)). A two-qubit gate changes the state of one qubit (commonly referred to as the target qubit) based on the current state of the other qubit (commonly referred to as the control qubit). For example, The CNOT gate flips the target qubit if the control qubit is in \(|1\rangle\) state. Similarly, the CRZ(\(\theta\)) gate rotates the target qubit along Z-axis by \(\theta\) if the control qubit is in \(|1\rangle\) state.

Noise in Quantum Computation: Quantum gates are error-prone. Besides, qubits suffer from decoherence i.e., they spontaneously interact with the environment and lose states. Therefore, the output of a quantum circuit is erroneous. The deeper quantum circuit needs more time for execution and gets affected by decoherence. More gates in the circuit also increase the accumulation of gate error. Thus, lower depth and the number of gates in the circuit improve noise resiliency. Parallel gate operations on different qubits can affect each other’s performance which is known as crosstalk.

Expectation Value of an Operator: Expectation value is the average of the eigenvalues, weighted by the probabilities that the state is measured to be in the corresponding eigenstate. Mathematically, expectation value of an operator \((\sigma)\) is defined as \(\langle \psi | \sigma | \psi \rangle\) where \(|\psi\rangle\) is the qubit state vector. For example, the expectation value of the Pauli-Z operator \((\sigma_z)\) is \(\langle \psi | \sigma_z | \psi \rangle\). If a qubit yields more \(|0\rangle\) \((|1\rangle\rangle)\) than \(|1\rangle\) \((|0\rangle\rangle)\), its Pauli-Z expectation value will be positive (negative). This value will vary in the range [-1, 1].

Quantum Neural Network: QNN involves parameter optimization of PQC to obtain a desired input-output relationship. The PQC generally consists of three segments: (i) a classical to quantum data encoding (also referred to as embedding in the literature) circuit, (ii) parameterized circuit, and (iii) measurement operations. A variety of encoding methods are available in the literature [15]. For continuous variables, the most widely used encoding scheme is angle encoding [8], [9], [15], [16] where a continuous variable input classical feature is encoded as a rotation of a qubit along the desired axis (X/Y/Z).

For ’n’ classical features, we require ’n’ qubits. In this work, we use RZ(\(X\)) on a qubit in superposition to encode any classical feature ‘X1’ as shown in Fig. 2(b). The H (Hadamard) gate is used to put a qubit in superposition. As the states produced by a qubit rotation along any axis will repeat in 2\(\pi\) intervals (Fig. 2(a)), features are...
QMLP – Iris (with varying number of parameterized circuit layers)

The number of trainable parameters 'n' qubits for a classification problem with 'n' continuous features used. The output from the hidden quantum layer is taken as the Pauli-operations. Again, any suitable parametric circuit structure may be scaled within 0 to 2\(\pi\). We use rotation along the Z-axis (achieving lower loss/higher accuracy at the same number of epochs). For further illustration, we took the trained QMLP models for 'Iris' classification with two parametric layers (referred to as 1L, 2L, 3L, and 4L; 1L corresponds to 1 parametric circuit layer). Cross-entropy loss is used for the loss function, Adagrad with a learning rate of 0.5 is used as an optimizer. Quantum noise is not considered. Note that the learning improved with an increasing number of layers as evident from Fig. 5 (achieving lower loss/higher accuracy at the same number of epochs). For example, the loss was 56.3% lower and the accuracy was 1.35% higher with 4L compared to 1L after 20 epochs. However, increasing the number of layers beyond 4 yielded diminishing returns as evident from Fig. 5(a).

B. DeepQMLP

Note that the performance improvement with an increasing number of layers in QMLP may not hold true under noise. In reality, deeper circuits are more susceptible to quantum noise due to higher gate error accumulation and decoherence. A circuit with 4 layers has twice as many gates and requires twice as much execution time as a 2L circuit. Therefore, the output state will be considerably more erroneous in a 4L circuit compared to a 2L circuit.
Features of ‘Iris’ Dataset

| Feature | Value |
|---------|-------|
| f1      | 1.80  |
| f2      | 6.30  |
| f3      | 3.00  |
| f4      | 2.26  |

Scaled Features (0-2)

| Feature | Value |
|---------|-------|
| f1      | 2.90  |
| f2      | 2.07  |
| f3      | 2.87  |
| f4      | 2.26  |

Fig. 4. The QMLP and DeepQMLP network architectures used in this work (with 2 parametric layers) for ‘Iris’ dataset classification. (f1, f2, f3, f4) are the 4 features of the dataset. Both these networks require 4 qubits. The number of trainable parameters i.e., $\theta_1$, ..., $\theta_{16}$ in these two networks are identical. The final layer is a classical dense layer (3 neurons) with SoftMax activation.

Fig. 5. (a) Loss and (b) accuracy of the trained QMLP models (‘Iris’ dataset with 1, 2, 3, and 4 parametric layers) on the training data during inference under noise (using noise parameters of IBM Melbourne device). The performance may decrease (higher loss/lower accuracy) with added layers under noise. A larger accumulation of noise in deeper circuits corrupts the output quantum state significantly which causes performance degradation.

In this section, we evaluate the trainability of the proposed models through empirical studies on various datasets. We train models with a varying number of parametric circuit layers to investigate their impact on performance. Additionally, we compare the performance of QMLP and DeepQMLP during inference under varying degrees of noise and demonstrate that DeepQMLP is more noise resilient.

Datasets: Apart from the ‘Iris’ dataset, we use four other synthetic datasets - R1_sq, P1_sq, R2_sq, and P2_sq with non-linear decision boundaries for classification as shown in Fig. 8. All these synthetic datasets have 2 features (both features are continuous variables varying between -1 and +1). R1_sq and P1_sq data samples are divided into two classes while R2_sq and P2_sq datasets have 3 classes each (denoted by different colors in Fig. 8). A total of 180 samples from each dataset are randomly picked for the training purpose. To keep the datasets balanced, we choose 90 samples/class for R1_sq and P1_sq and 60 samples/class for R2_sq and P2_sq.

Framework and Setup: We develop a Python framework to implement the networks in this study using PennyLane, Pytorch, and Qiskit frameworks \[10, 17, 18\]. We use the PennyLane framework to model the quantum circuits and Pytorch to model and train the hybrid network. Qiskit is used to perform all the simulations of quantum circuits under noise. In all the training runs, we use an initial learning rate of 0.5 and train the models using mini-batch gradient descent with a batch size of 30. We use the Adagrad optimizer which updates the parameters in these two networks are identical. The last hidden quantum layer is connected to a densely connected classical layer similar to QMLP. The circuit structure of these hidden layers can be different from each other. However, in this work, we use identical circuit structures for all the hidden layers. The training procedure is similar to QMLP.

Example 2: An example of the DeepQMLP architecture is shown in Fig. 4 to classify the ‘Iris’ dataset. The 4 input classical features are encoded into the qubits of the first hidden layer using angle encoding. The four expectation values of the qubits from the output of the first hidden layer (E1, E2, E3, and E4) are encoded as a quantum state in the next hidden layer using angle encoding. The output of the second hidden layer feeds the classical dense layer with 3 neurons. Overall, the network has $(2*4+1^2 + 4*3)$ or 28 parameters (16 circuit parameters and 12 classical weights). In Fig. 6(b), we show the training loss and accuracy of DeepQMLP over 50 epochs of training (with 4, 6, and 8 cumulative circuit layers). We show the performance of QMLP with a similar number of circuit parameters in Fig. 5(a). Note that, the performance (noiseless) of the DeepQMLP models stayed close to the QMLP models with the same number of parameters (except the 4 layer one). For example, the difference in training loss and accuracy between 8 layer QMLP and DeepQMLP models was below 1%. This empirical study shows that both QMLP and DeepQMLP architectures are trainable and under ideal scenarios (noiseless), the DeepQMLP model performance is at par with the QMLP models. In the following section, we show that the DeepQMLP models may show greater noise resilience during inference due to shallower circuits.
Trainability of QMLP and DeepQMLP: In the previous section, we have shown the training cost/accuracy curves of QMLP and DeepQMLP with varying number of layers to classify the ‘Iris’ dataset. Here, we pick the synthetic datasets and train them using QMLP and DeepQMLP architectures with 4, 6, and 8 parametric layers. Note that, a 4 Layer QMLP model has the same number of trainable parameters as a 4 Layer DeepQMLP model (with the same dataset, the same number of trainable parameters) were on the lower side at the end of the training. DeepQMLP showed 20.86% lower loss and 1.12%, higher accuracy over the QMLP models as shown in Fig. 9(d) (indicated by the yellow double arrows). At these noise levels, both QMLP and DeepQMLP models showed similar loss and accuracy over the training data (for scaling factors of 0.25, 0.5, 1.0, 2.0, and 4.0). Note that, the nominal 0.1% and 1% error probabilities of the single-qubit and two-qubit gate errors are at par with the reported noise levels of the current generation of IBM quantum computers. We avoided other quantum device architectural constraints (e.g., limited connectivity) to avoid unnecessary complexity in the comparison. The results are shown in Fig. 9(a)-(e).

For lower error values, both QMLP and DeepQMLP models showed similar loss and accuracy over the training data (for scaling factors of 0.25, 0.5, and 1.0). This is expected because, at lower noise levels, the output states of the quantum circuits (both in QMLP and DeepQMLP) are not far from the ideal. However, at a higher noise level (scaling factor 2), the DeepQMLP model showed lower loss and higher accuracy over the QMLP models as shown in Fig. 9(d) (indicated by the yellow double arrows). At these noise levels, both QMLP and DeepQMLP hidden quantum layers produce erroneous output states that are far from the ideal. However, the shallow-depth DeepQMLP hidden layer outputs are less erroneous compared to the QMLP hidden layers because of a smaller number of gates per hidden layer. Therefore, the overall network performance is less affected by gate noises in DeepQMLP models. On average, DeepQMLP showed 20.86% lower loss and 1.12%, higher accuracy over the QMLP models at 2x scaling of the noise. The gap increased further at 4x scaling of the noise as evident in Fig. 9(e). On average, DeepQMLP models showed 25.3% lower loss and 7.93% higher accuracy over the QMLP models at 4x noise scaling. This study indicates a greater noise resilience of DeepQMLP over QMLP.

V. CONCLUSION

We present two new quantum-classical hybrid neural network architectures QMLP and DeepQMLP for classical data classification. We show the trainability of these models through empirical studies on 5 different datasets and 78 training runs. We also show that the
trained models generalize well. The DeepQMLP model shows greater noise resilience over the QMLP models (up to 25.3% lower loss and 7.93% higher accuracy). These architectures provide new directions to develop large-scale machine learning applications.

Acknowledgements: The work is supported in parts by NSF (CNS-1722557, CNS-2129675, CCF-1718474, OIA-2040667, DGE-1723687, DGE-1821766, and DGE-2113839) and seed grants from (CNS-1722557, CNS-2129675, CCF-1718474, OIA-2040667, DGE-1723687, DGE-1821766, and DGE-2113839) and seed grants from Penn State ICDS and Huck Institute of the Life Sciences.

REFERENCES

[1] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell et al., “Quantum supremacy using a programmable superconducting processor,” Nature, vol. 574, no. 7779, pp. 505–510, 2019.

[2] E. Pednault, J. Gunnels, D. Maslov, and J. Gambetta, “On “quantum supremacy”,” IBM Research Blog, vol. 21, 2019.

[3] E. Farhi and H. Neven, “Classification with quantum neural networks on near term processors,” arXiv preprint arXiv:1802.06902, 2018.

[4] N. Killoran, T. R. Bromley, J. M. Arrazola, M. Schuld, N. Quesada, and S. Lloyd, “Continuous-variable quantum neural networks,” Physical Review Research, vol. 1, no. 3, p. 033063, 2019.

[5] I. Cong, S. Choi, and M. D. Lukin, “Quantum convolutional neural networks,” Nature Physics, vol. 15, no. 12, pp. 1273–1278, 2019.

[6] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, “Expressive power of parametrized quantum circuits,” Physical Review Research, vol. 2, no. 3, p. 033125, 2020.

[7] L. G. Wright and P. L. McMahon, “The capacity of quantum neural networks,” in CLEO: QELS Fundamental Science. Optical Society of America, 2020, pp. JM4G-5.

[8] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Wernher, “The power of quantum neural networks,” arXiv preprint arXiv:2011.00027, 2020.

[9] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, “Circuit-centric quantum classifiers,” Physical Review A, 2020.

[10] A. Cross, “The ibm q experience and qiskit open-source quantum computing software,” in APS March Meeting Abstracts, vol. 2018, 2018, pp. L58–003.

[11] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven, “Barren plateaus in quantum neural network training landscapes,” Nature communications, vol. 9, no. 1, pp. 1–6, 2018.

[12] M. Alam, A. Ash-Saki, and S. Ghosh, “Design-space exploration of quantum approximate optimization algorithm under noise,” in 2020 IEEE Custom Integrated Circuits Conference (CICC). IEEE, 2020, pp. 1–4.

[13] S. Wang, E. Fontana, M. Cerezo, K. Sharma, A. Sone, L. Cincio, and P. Coles, “Noise-induced barren plateaus in variational quantum algorithms,” Bulletin of the American Physical Society, 2021.

[14] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles, “Cost function dependent barren plateaus in shallow parametrized quantum circuits,” Nature Communications, vol. 12, no. 1, pp. 1–12, 2021.

[15] M. Schuld, R. Sweke, and J. J. Meyer, “Effect of data encoding on the expressive power of variational quantum-machine-learning models,” Physical Review A, vol. 103, no. 3, p. 032430, 2021.

[16] S. Lloyd, M. Schuld, A. Ijaz, J. Izac, and N. Killoran, “Quantum embeddings for machine learning,” arXiv preprint arXiv:2011.03622, 2020.

[17] V. Bergholm, J. Izac, M. Schuld, C. Gogolin, M. S. Alam, S. Ahmed, J. M. Arrazola, C. Blank, A. Delgado, S. Jahangiri et al., “Pennylane: Automatic differentiation of hybrid quantum-classical computations,” arXiv preprint arXiv:1811.04968, 2018.

[18] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., “Pytorch: An imperative style, high-performance deep learning library,” arXiv preprint arXiv:1912.01703, 2019.

[19] A. Ash-Saki, M. Alam, and S. Ghosh, “Qure: Qubit re-allocation in noisy intermediate-scale quantum computers,” in Proceedings of the 56th Annual Design Automation Conference 2019, 2019, pp. 1–6.