Computational Advantage in Hybrid Quantum Neural Networks: Myth or Reality?

Muhammad Kashif *†, Alberto Marchisio*†, Muhammad Shafique*†
* eBrain Lab, Division of Engineering, New York University Abu Dhabi, PO Box 129188, Abu Dhabi, UAE
† Center for Quantum and Topological Systems, NYUAD Research Institute, New York University Abu Dhabi, UAE
Emails: {muhammadkashif, alberto.marchisio, muhammad.shafique}@nyu.edu

Abstract*—Hybrid Quantum Neural Networks (HQNNs), under the umbrella of Quantum Machine Learning (QML), have garnered significant attention due to their potential to enhance computational performance by integrating quantum layers within traditional neural network (NN) architectures. Despite numerous state-of-the-art applications, a fundamental question remains: *Does the inclusion of quantum layers offer any computational advantage over purely classical models? If yes/no, how and why?

In this paper, we analyze how classical and hybrid models adapt their architectural complexity in response to increasing problem complexity. To this end, we select a multiclass classification problem and perform comprehensive benchmarking of classical models for increasing problem complexity, identifying those that optimize both accuracy and computational efficiency to establish a robust baseline for comparison. These baseline models are then systematically compared with HQNNs by evaluating the rate of increase in floating-point operations (FLOPs) and number of parameters, providing insights into how architectural complexity scales with problem complexity in both classical and hybrid networks. We utilize classical machines to simulate the quantum layers in HQNNs, a common practice in Noisy Intermediate-Scale Quantum (NISQ) era. Our analysis reveals that, as problem complexity increases, the architectural complexity of HQNNs, and consequently their FLOPs consumption, despite the simulation overhead associated with quantum layer's simulation on classical hardware, scales more efficiently (53.1% increase in FLOPs from 10 features (low problem complexity) to 110 features (high problem complexity)), compared to classical networks (88.1%). Moreover, as the problem complexity increases, classical networks consistently exhibit a need for a larger number of parameters to accommodate the increasing problem complexity. Additionally, the rate of increase in number of parameters is also slower in HQNNs (81.4%) than classical NNs (88.5%). These findings suggest that HQNNs provide a more scalable and resource-efficient solution, positioning them as a promising alternative for tackling complex computational problems.

I. Introduction

Quantum Machine Learning (QML) is an emerging research area that integrates quantum computing with machine learning (ML) techniques, aiming to enhance computational power and efficiency in processing complex datasets [1]–[4]. Quantum Neural Networks (QNNs) are a core component of QML, employing quantum circuits/layers to mimic the structure and function of classical neural networks (NNs) [5], [6]. Hybrid Quantum Neural Networks (HQNNs) advance this concept by combining quantum and classical NN, leveraging the strengths of both quantum and classical computation [7]–[9]. HQNNs are particularly advantageous in the Noisy Intermediate-Scale Quantum (NISQ) era, where they can capitalize on quantum capabilities for specific sub-tasks while relying on classical resources to manage error-prone operations [10], [11]. In a typical HQNN architecture, one or more hidden layers of a classical NN are replaced by a trainable quantum layer [12], [13], as shown in Fig. 1.

A growing body of state-of-the-art research employs HQNNs across a wide range of applications [14]–[22]. Despite the notable computational speedups demonstrated by quantum computing in various domains [23]–[26], no prior study has systematically investigated whether the integration of *quantum* components in NNs offers any advantages in learning performance or computational efficiency [27]. Additionally, demonstrating quantum advantage for QML is challenging due to several reasons, as critically discussed in [28].

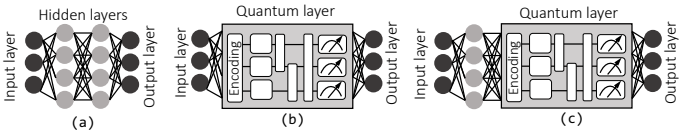

Fig. 1: Illustration of (a) classical NN, (b) HQNN with only quantum hidden layer and (c) HQNN with classical and quantum hidden layers.

A. Open Research Questions

Comparing classical and quantum models is not straight-forward due to their fundamentally different computational paradigms. In the NISQ era, where quantum computers are limited by noise and scale, most research relies on simulating quantum algorithms using classical hardware[1] [27], which is computationally intensive due to the exponential scaling of quantum states [29]. *By examining the increase in architectural complexity (required to address growing problem complexity) and the corresponding increase in computational demands of QML models simulated on classical machines, potential quantum advantages over classical models can be evaluated.* The argument is that, if QML models exhibit superior performance despite the overhead associated with simulating quantum circuits on classical hardware, this suggests that the observed superiority/advantage is likely inherent to the quantum nature of the algorithms and could become more significant with the deployment of universal fault-tolerant quantum computers. *This approach provides valuable insights into the scalability and impact of quantum computing, offering a pragmatic pathway to evaluate its future role in solving complex problems beyond classical capabilities.*

To this end, recently, there have been some efforts on benchmarking QML algorithms, primarily focusing on analyzing which algorithms perform better in specific scenarios [27], [30]. However, the question of whether incorporating the quantum component in NNs framework provides any advantage remains unexplored and is an *open and important* research problem. **In this paper, we raise several key questions to investigate this gap:**

- **Q1:** *What metrics will be appropriate to evaluate and compare the computational complexity of classical and hybrid networks?*
- **Q2:** *Does the quantum part in HQNNs add anything qualitatively different or important?*
- **Q3:** *If the quantum part gives any advantage in HQNNs, and can we demonstrate if it actually is computational advantage?*

B. Our Contributions

We attempt to answer the above questions through a rigorous empirical analysis. **Our main contributions are:**

- We propose a systematic methodology for comparing the computational complexity of hybrid and classical networks. For this purpose we generate a synthetic dataset in such a way that we can

[1]List of quantum simulators (Accessed 15-11-2024): https://www.quantiki.org/wiki/list-qc-simulators

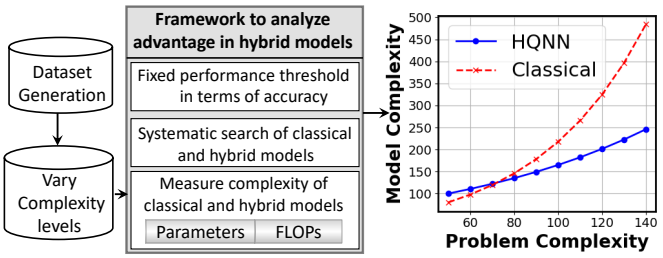

Fig. 2: An overview of our contributions.

increase the complexity of the problem by increasing the number of features. Our novel investigation unleash how both classical and hybrid networks adapt to the complexity of the problem. (**Section III**)

- We perform grid search to identify the classical models that meet a predefined accuracy threshold across varying levels of problem complexity to serve as benchmarks. We then identify the most efficient hybrid network architectures (models that achieve the predefined accuracy threshold) through a similar grid search process. (**Section III-B and III-C**)
- A key finding from our analysis is that *as the problem complexity increases, classical NNs require more sophisticated and resource-intensive architectures to reach a certain accuracy, resulting in higher FLOPs consumption*. In contrast, *HQNNs demonstrate superior scalability and efficiency*. A well-designed hybrid architecture, particularly with a more *expressive quantum layer*, remains largely unaffected by the increasing complexity of the problem. An architecture that effectively solves a simpler problem can often be adapted, to solve a more complex problem. (**Section IV**)
- Another important finding of our analysis is that *classical models consistently require a greater number of parameters across all levels of problem complexity, even at lower complexity levels*. In contrast, *HQNNs generally need fewer parameters, highlighting their efficiency in parameter utilization.* (**Section IV**)

Summary of key results: This study demonstrates the superior scalability of HQNNs compared to classical NNs as problem complexity increases. In our analysis, we find that despite simulation overhead, HQNNs with expressive quantum layers require approximately 7.5% fewer FLOPs and 42.3% fewer parameters to solve complex problems (110 features in our case) than classical networks. Additionally, HQNNs scale more efficiently, with a 53.1% increase in FLOPs while going from simple to complex tasks, compared to 88.5% for classical models. Similarly, the rate of increase in number of parameters for classical models scales is 88.5%, while HQNNs show a slower rate of increase in FLOPs that accounts to 81.4%.

II. Complexity of Neural Networks

There is no single, definitive measure for assessing the NN's complexity, however, several approximate metrics exist [31]–[36]. Therefore, we argue that relying on a single measure may not provide a comprehensive understanding of the network's complexity. Hence, it is advisable to use multiple methods to better estimate the complexity. We use FLOPs and the number of parameters as indicators to assess the network's complexity.

A. Floating Point Operations (FLOPs)

FLOPs are a widely recognized measure of computational complexity in NNs [37]–[40]. FLOPs provide an estimate of the total number of arithmetic operations required to execute a model on a given input. This metric serves as a proxy for understanding the computational resources, such as processing power and energy consumption that are needed to train and evaluate NNs. By quantifying FLOPs, one can objectively compare the efficiency of different architectures, enabling more informed decisions about model design and scalability.

B. Number of Parameters

The number of parameters represent a total number of trainable weights in a NN and is often used as a measure of model complexity, with larger networks (with more parameters) generally being more expressive and capable of capturing complex data patterns [41]. In NNs, the parameter count typically scales with the number of layers and neurons per layer, and is hence useful for evaluating model complexity [42].

III. Our Methodology

In this paper, we argue that two models are comparable if they achieve similar performance (accuracy) on a given task, regardless of differences in their size. Accordingly, we set an accuracy threshold of $\geq 90\%$ for both training and validation, and perform an extensive grid search for models that meet this condition regardless of their size. The detailed methodology is shown in Fig. 3. Below, we explain different steps of our methodology in detail.

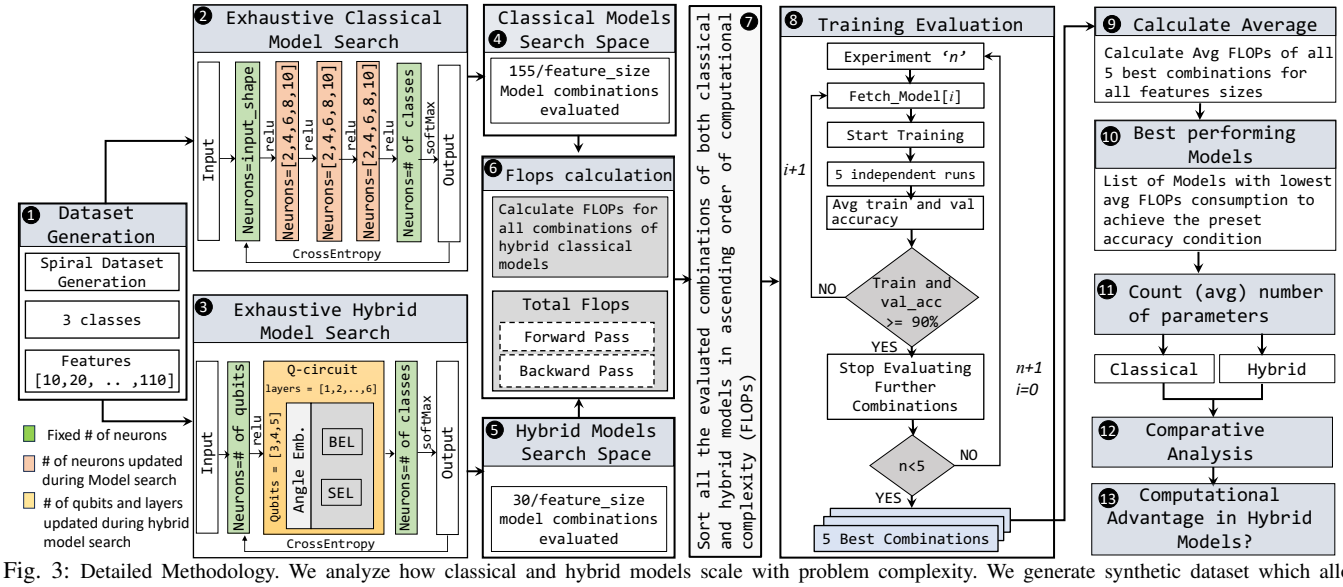

Fig. 3: Detailed Methodology. We analyze how classical and hybrid models scale with problem complexity. We generate synthetic dataset which allows to progressively increase the problem complexity. We then search for classical and hybrid models using grid search approach and compare the FLOPs and parameters. To account for randomness in NNs training, the results are averaged over 5 experiments. Small black boxes show the order of methodology steps.

A. Synthetic Dataset Generation

We generate synthetic datasets as these datasets play a crucial role in benchmarking ML models, offering precise control to systematically increase task complexity and evaluate model performance under varying conditions [27], [43], [44]. We generate a spiral dataset with 1500 data points distributed across 3 classes. The dataset's core structure is a spiral, as shown in Fig. 4(a) with each class represented by a distinct arm, offering a challenging yet learnable pattern for models. To investigate the effect of task complexity, we maintain a fixed number of data points and classes while systematically increasing complexity by augmenting the number of features. The increase in feature dimensions is accompanied by a corresponding increase in noise, further enhancing the complexity of the dataset. Specifically, the additional features introduce subtle variations through non-linear transformations of the existing features. Controlled noise is introduced, scaled with the number of features ($noise = 0.1 + 0.003 \times num\_features$), ensuring a progressive rise in task difficulty. We utilize feature sizes ranging from 10 to 110 in increments of 10. From this point onward, we refer to feature size as complexity level, with the associated noise implicitly applied unless explicitly stated otherwise. This incremental increase in feature dimensions results in a progressively challenging classification task, as illustrated in Fig. 4(b).

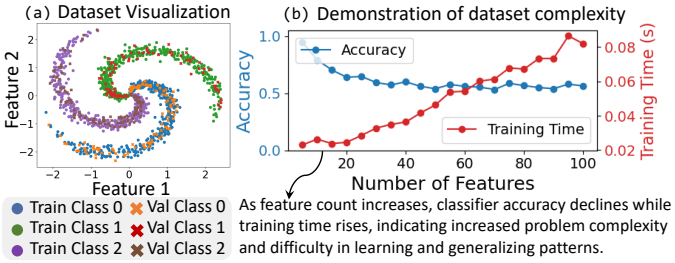

Fig. 4: (a) Visualization of first two features of generated dataset, and (b) Demonstration of increasing problem complexity.

B. Exhaustive Classical Model Search

We use grid search algorithm to systematically search for the models that reach the preset accuracy condition ($\geq 90\%$).

***Working of Grid Search Algorithm*:** For n layers, and m neuron options, our grid search algorithm evaluates all the possible combinations based on formula: Total no. of combinations $= m\frac{m^n-1}{m-1}$. For instance, if $m = [2, 3]$, $n = 2$, this means we can have minimum 1 and maximum 2 layers. The neurons in each layer can be either 1 or 2. Using the general formula, there are 6 possible combinations, i.e., $[2], [3], [2, 2], [2, 3], [3, 2], [3, 3]$.

Classical Model Search Space: We restrict the classical models to have a maximum of $n = 3$ layers, with the number of neurons in each layer chosen from the set $m = \{2, 4, 6, 8, 10\}$ resulting in a search space of total 155 model combinations for classical models for each complexity level.

C. Exhaustive Hybrid Model Search

For hybrid models, the number of neurons in the first and last classical layers are fixed. The number of neurons in the input layer are equal to the number of qubits because we use angle encoding that requires one qubit for one feature [45]. The neurons in the output layer are equal to the number of classes (3) in the dataset.

Hybrid Model Search Space: During the hybrid model search only the quantum layers are varied. We use $[3, 4, 5]$ qubits quantum layers, and for each qubit size, quantum layers of depth $[1, 2, 3, ..., 10]$ are tested, yielding 30 model combinations per feature size.

Quantum Layer Design: For the underlying quantum layers in HQNNs, we use two different types designs which are are separately evaluated: the Strongly Entangling Layer (SEL) and the Basic Entangling Layer (BEL), as shown in Fig. 5. For both layer types, each qubit number is experimented with all the layer depths, to identify the optimal configuration, i.e., that meets the preset accuracy threshold of $\geq 90\%$.

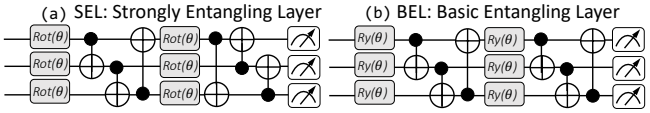

Fig. 5: Example of (a) SEL and (b) BEL quantum layer design, both having 3 qubits and depth of 2 layers. Same structure is repeated for more layers.

D. FLOPs Calculation

We use built-in functions from Tensorflow to compute the FLOPs for both and forward and backward pass. The procedure to compute FLOPs involves converting the `TensorFlow/Keras` models (both classical and HQNNs[2]) into a concrete function using TensorFlow's `convert_variables_to_constants_v2` utility. The frozen computation graph of the model is then analyzed using the `TensorFlow Profiler` with options set to compute the total number of floating-point operations.

For the forward pass, the profiler directly computes FLOPs based on the operations in the model graph. For the backward pass, we define a dummy loss function and use TensorFlow's `GradientTape` to compute gradients, which are also converted to a frozen computation graph. FLOPs for the backward pass are then calculated in a similar manner. The total FLOPs are obtained by summing the FLOPs of the forward and backward passes, providing a comprehensive measure of the model's computational complexity.

E. Models Sorting based on FLOPs consumption

Once the FLOPs are calculated, the model combinations are sorted in ascending order based on computational complexity (higher FLOPs means complex model and vice versa). This sorting ensures that we avoid training all models unnecessarily as the first model in the list that satisfies the accuracy condition is the model with the lowest computational complexity.

F. Models Training

The sorted list of models are trained sequentially for various levels of problem complexity. To account for randomness of parameter initialization and statistical reliability of the results, each model combination subjected to five independent runs. Each run involves training for 100 epochs, and the highest training and validation accuracy across epochs is recorded. The maximum accuracies from all runs are averaged to provide a robust performance measure for each model combination for a given complexity level. If any model achieves an average training and validation accuracy of $\geq 90\%$, further evaluation of additional model combinations for that complexity level is stopped. This entire process is repeated five times to address the stochastic nature of NNs training, where different model configurations might result as optimal. By repeating the procedure, a more accurate estimate of the average complexity required to solve the problem is obtained. Once five top-performing combinations are identified for a given complexity level, the evaluation continues with the next complexity level.

G. Models with Lowest Flops Consumption

At the end of the experimentation process, the average computational complexity required by both classical and hybrid networks is obtained for varying levels of problem complexity. These average complexities were then compared to determine whether hybrid networks provide any computational advantage over classical networks as the complexity of the problem increases. This

[2]We convert quantum layers in HQNNs into Tensorflow/keras layer using pennylane, more details at: https://pennylane.ai/qml/demos/tutorial_qnn_module_tf

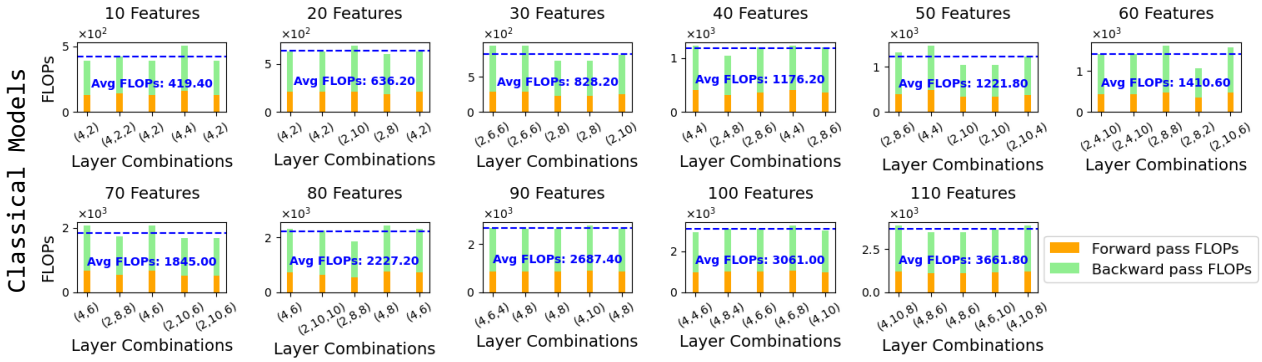

Fig. 6: FLOPs consumption of best-performing classical models for different complexity levels of the problem. Each subplot displays the FLOPs consumption of the top five performing models from five independent runs of model search, corresponding to different feature sizes.

comparative analysis is crucial in evaluating the scalability and efficiency of hybrid models, particularly in scenarios where the problem's complexity poses significant computational challenges for classical approaches.

IV. Results and Discussion

For the experiments, we utilized PennyLane, a Python-based framework designed for differentiable quantum programming [46]. During the training phase, the learning rate was set to 0.001, and a batch size of 8 was employed for both classical and hybrid models, with a total of 100 training epochs.

A. Classical Models: Analysis of FLOPs Consumption

In this section, we present our findings on the computational complexity required by classical models to solve the problem under predefined accuracy conditions, as shown in Fig. 6. By varying problem complexity (increasing features and noise, see Section III-A), we evaluate the scalability and computational efficiency of classical models. We observe that at lower complexity, simpler models achieve the desired accuracy. However, as the problem complexity increases (e.g., up to 110 and associated high noise level), more sophisticated models, i.e., with additional layers and neurons are needed, resulting in a significant increase in FLOPs and hardware resources.

B. Hybrid Models: Analysis of FLOPs Consumption

We now present the results illustrating how both the hybrid models adapt to increasing problem complexity.

Discussion for BEL-based HQNNs: The results in Fig. 7 show the computational complexity of the BEL-based hybrid model for varying problem complexities. For problems with up to 40 features, the same quantum circuit architecture (3 qubits, 2 layers) is sufficient. The increase in FLOPs is mainly due to the increase in classical input layer size as feature count grows, while the quantum layer remains unchanged in depth (layer repetitions) and width (no. of qubits). However, further increase in problem complexity necessitates making the underlying quantum layer more expressive to effectively handle the increased complexity of the problem. This involves adding more qubits and increase quantum layer's depth. These changes in the architecture lead to a significant rise in FLOPs, reflecting an increase in the overall computational demands and greater hardware resource consumption.

Discussion for SEL-based HQNNs: The SEL quantum layer has a more intricate entanglement design than the BEL, enhancing its expressiveness and ability to capture complex patterns, as shown in Fig. 5(b). Unlike BEL, the SEL structure remains fixed at 3 qubits and 2 layers, effectively solving problems with up to 110 features without altering the quantum architecture, as shown in Fig. 8. The increase in FLOPs is attributed to the classical component's growth, particularly in the input layer to accomodate higher feature size, while the quantum layer remains unchanged, providing an efficient framework for handling complex tasks without additional quantum resources.

To make the argument tighter, we count the number of parameters of the best-performing models for both classical and hybrid approaches. This analysis is crucial, as it is possible that hybrid models have more parameters. This would potentially explain why their complexity in terms of rate of increase in FLOPs consumption does not increase as rapidly as classical models—they may be sufficiently expressive from the start. Due to page limitations, we present parameter counts for selective feature sizes $(10, 40, 80, 110)$ *for both classical and hybrid models.*

C. Classical Models: Analysis of # of Parameters

For classical models, we observed a substantial increase in the number of trainable parameters as problem complexity is increased. This is primarily due to the need for more sophisticated architectures, such as additional layers and neurons, to maintain the desired accuracy with for increased complexity (Fig. 9, top panel). The trend suggests that classical models require enhanced expressiveness to manage the increased input dimensionality and noise. The increase in parameters correlates closely with the rise in FLOPs,

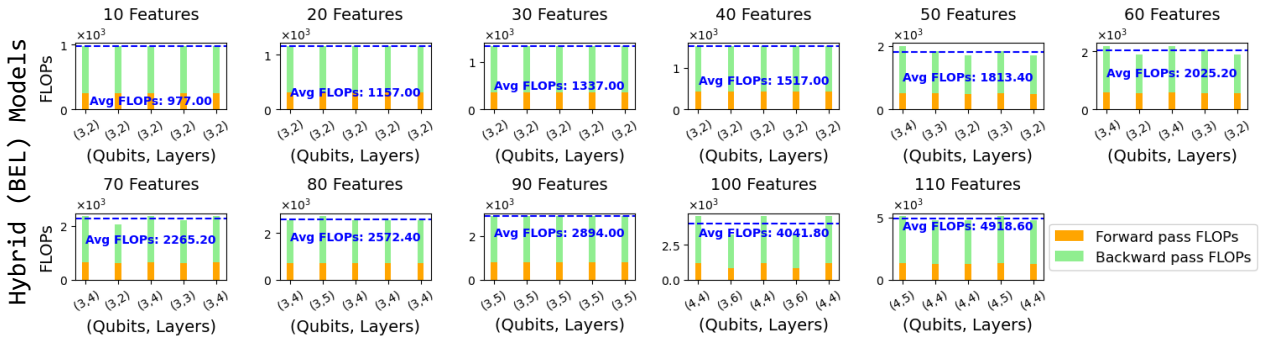

Fig. 7: FLOPs consumption of best-performing hybrid (BEL) models for different complexity levels of the problem. Each subplot displays the FLOPs consumption of the top five performing models from five independent runs of model search, corresponding to different feature sizes.

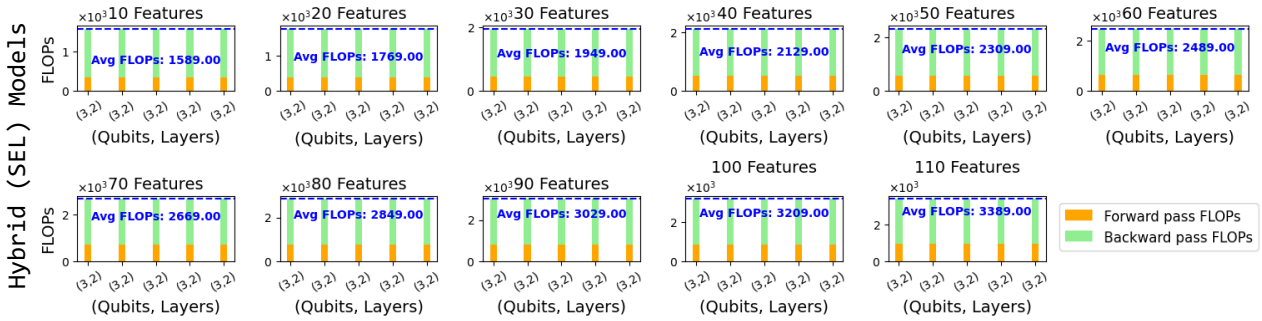

Fig. 8: FLOPs consumption of best-performing hybrid (SEL) models for different complexity levels of the problem. Each subplot displays the FLOPs consumption of the top five performing models from five independent runs of model search, corresponding to different feature sizes.

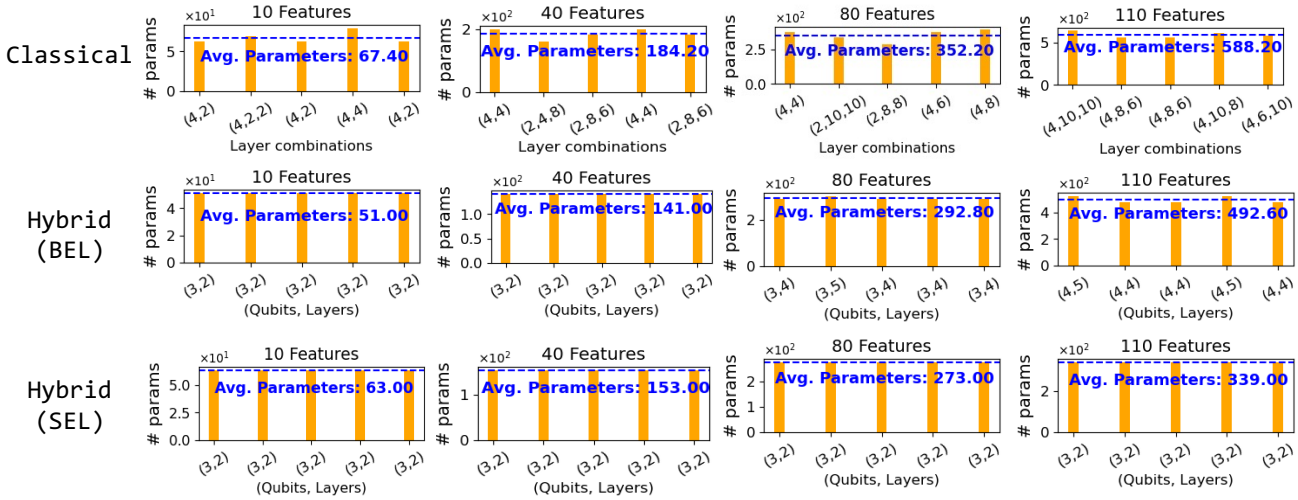

Fig. 9: Number for parameters for five top-performing classical models (top pannel), Hybrid (BEL) Models (middle pannel) and Hybrid (SEL) Models (bottom pannel) for different level of problem complexity.

further reinforcing the notion that classical models demand greater computational resources as problem complexity increases.

D. Hybrid Models: Analysis of # of Parameters

For BEL-based hybrid models, the model complexity remains constant for problem complexity up to 40 features, with an average parameter count lower than that of classical models. Beyond 40 features, however, an increase in the number of qubits and quantum circuit depth is necessary, leading to a moderate rise in parameter count (Fig. 9, middle panel). This increase is mainly due to the larger input layer resulting from the bigger feature size. Despite this, BEL-based hybrid models maintain a lower parameter count compared to their classical counterparts. In contrast, hybrid models with the SEL quantum layer exhibit minimal changes in parameter count across all complexity levels. The same quantum layer configuration is sufficient regardless of the problem complexity, with only a slight increase in parameters due to additional neurons in the input layer. Overall, SEL-based hybrid models demonstrate superior parameter efficiency compared to both classical and BEL-based hybrid models.

E. Classical Vs. Hybrid - Comparative Analysis of FLOPs and # of Parameters

We now compare the classical and hybrid models in terms of FLOPs consumption and the number of parameters. The comparative analysis is aimed to provide a clear picture of how the complexity of classical and hybrid models scale with problem complexity. As discussed in Section III-F, five independent experiments are conducted for each complexity level to identify the best-performing models. For this comparative analysis, we select the smallest model from the set of five best-performing configurations for both classical and hybrid approaches.

a) Comparison of FLOPs Consumption

FLOPs comparison of best-performing classical and both the hybrid models, as a function of the problem complexity is presented in Fig. 10(a). The results demonstrate how these models scale in terms of computational complexity as the problem complexity increases. The classical model exhibited a consistent linear rate of

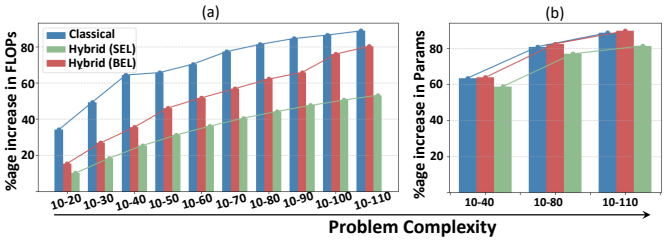

Fig. 10: Comparison of rate of increase in FLOPs and # of parameters of classical and HQNNs with increasing probem complexity. As the problem complexity increases, the rate of increase in FLOPs is more pronounced in classical models than hybrid models. Similarly, the rate of increase of parameter is also better in HQNNs especially with strongly entangling layers.

increase in FLOPs consumption with increasing problem complexity. Specifically, there was an absolute increase of 3285 FLOPs (obtained by subtracting the avg. FLOPs of 10 features from FLOPs of 110 features in Fig. 6), representing an 88.5% rise in computational demands when scaling from 10 to 110 features.

BEL-based hybrid models also exhibit a linear rate of increase in FLOPs but at a slightly lower rate than classical models, across the same complexity levels, with an absolute rise of 3941.6 FLOPs (obtained by subtracting the avg. FLOPs of 10 features from FLOPs

of 110 features in Fig. 7). representing an increase of 80.13% computational demands when scaling from 10 to 110 features.. SEL-based hybrid models also showed a linear increase in computational demand, but at a significantly lower rate than both classical models and BEL-based hybrid models. The absolute increase was only 1800 FLOPs (obtained by subtracting the avg. FLOPs of 10 features from FLOPs of 110 features in Fig. 8), when the number of features increased from 10 to 110, reflecting 53.1% rate of increase in FLOPs. This is substantially lower than the rates observed for classical and BEL-based models, despite the overhead of simulating quantum layers on classical hardware.

An important observation is that the rate of increase in computational demands, measured in terms of FLOPs consumption, is significantly lower in hybrid models compared to classical models. This underscores the superior adaptability of HQNNs to increasing problem complexity, particularly in HQNNs with more expressive underlying quantum layers, such as those utilizing SEL architectures.

b) Comparison of # of Parameters

Fig. 10(b) compares the average number of parameters for classical and both hybrid models as a function of problem complexity. Classical models exhibit a steady rise in parameter count with increasing feature size, reflecting the need for greater number of parameters and expressiveness to maintain a certain performance as problem complexity grows. Specifically, an absolute increase of 520.8 parameters is observed (obtained by subtracting the avg. parameters of 10 features from avg. params of 110 features in Fig. 9(top panel)), corresponding to an 88.5% rate of increase in parameter count, as the problem complexity increases from 10 to 110 features.

BEL-based hybrid models also exhibit an increase in parameter count with growing problem complexity. In particular, an absolute increase of 441 parameters (obtained by subtracting the avg. parameters of 10 features from avg. params of 110 features in Fig. 9(middle panel)), at a rate of 89.6% is observed, which is very close to the rate of increase of parameters in classical models. SEL-based hybrid models demonstrate the slowest parameter growth, with incrementally increased problem complexity. While going from low to high problem complexity an absolute increase of 276 parameters (obtained by subtracting the avg. parameters of 10 features from avg. params of 110 features in Fig. 9(bottom panel)) at a rate of 81.4% is observed, which is significantly lower than both classical and BEL-based hybrid models.

These results indicate that the rate of increase in parameter count as a function of problem complexity aligns closely with the rate of increase in FLOPs consumption. Notably, HQNNs employing more expressive quantum layers, such as SEL, demonstrate exceptional adaptability to increasing problem complexity, requiring minimal modifications to their architectural complexity. This observation suggests a potential computational advantage of incorporating quantum components into neural networks, highlighting the inherent benefits of quantumness in addressing complex problems.

F. Ablation Study: Determining the FLOPs of Quantum Layers

The HQNN framework incorporates classical layers for input preprocessing and postprocessing, as well as data encoding routines, which are computationally intensive when exploiting classical machines to run HQNN models. Once the universal fault-tolerant quantum computers and the development of proper quantum backpropagation techniques for QML algorithms would be developed, the reliance on classical layers and classical optimization routines is expected to diminish. Furthermore, the availability of quantum-native datasets would eliminate the need for data encoding, as the data would inherently exist in a quantum-compatible format. Therefore, we now compute the FLOPs consumption for classical and encoding components of HQNNs, to estimate the FLOPs consumption of actual trainable quantum layers. Table I provides a detailed breakdown. Understandably, as feature size increases, total FLOPs consumption increases for both hybrid models, due to the higher input dimensionality. The hybrid (BEL) models exhibit a more rapid increase, indicating greater sensitivity to input complexity. A substantial portion of the total FLOPs is consumed by classical layers and encoding. For instance, at a feature size of 110, the classical and encoding components in BEL-based hybrid model consumes 3901 FLOPs, representing the majority of the 4797 total FLOPs, whereas, the quantum layer only requires 896 FLOPs, about 18.7% of the total. On the other hand, for more sophisticated HQNN designs, particularly those employing SEL, the FLOPs consumption of the quantum layer remains constant, demonstrating the efficient handling of increasingly complex problems by more expressive quantum networks. This stability underscores the capability of these quantum layers to adapt to problem complexity without a proportional increase in computational demands.

TABLE I: Ablation study Results: Breakdown of FLOPs consumption for different across the different stages of the hybrid networks (in FLOPs). FS=feature size, BC=best combination, TF=total flops, Enc=Encoding, CL=classical layers, QL=quantum layer

Model	FS/BC	TF	Enc+CL	CL	Enc	QL
Hybrid (BEL)	10/(3,2)	977	749	283	749-283=466	228
	40/(3,2)	1517	1289	823	1289-823=466	228
	80/(3,4)	2537	2009	1543	2009-1543=466	528
	110/(4,4)	4797	3901	2769	3901-2769=1132	896
Hybrid (SEL)	10/(3,2)	1589	749	283	749-283=466	840
	40/(3,2)	2129	1289	823	1289-823=466	840
	80/(3,2)	2849	2009	1543	2009-1543=466	840
	110/(3,2)	3389	2549	2083	2549-2083=466	840

V. Conclusion

In this paper, we conducted a comprehensive benchmarking study to evaluate the computational efficiency of Hybrid Quantum Neural Networks (HQNNs) compared to classical neural networks (NNs). Our key findings aim at providing valuable insights to the research questions raised in the introduction as summarized below:

(A1) Since there exists no definitive measure of computational complexity in NNs, relying solely on a single metric to draw conclusions about computational complexity may lead to limited insights. Therefore, we employed two independent metrics, i.e., floating-point operations (FLOPs) and parameter count to evaluate computational complexity. We argue that if the trends observed in the results of these distinct metrics align, the conclusions derived from the analysis are more robust and meaningful.

(A2) Our analysis reveals that as the problem complexity increases, classical NNs demonstrate a substantial rise in both FLOPs and the number of parameters due to the requirement of more sophisticated architecture to maintain a certain accuracy. This indicates limited adaptability of classical NNs to the increasing complexity of the problem. In contrast, HQNNs exhibit a more efficient computational footprint. Specifically, quantum layers within HQNNs consume significantly fewer FLOPs and require fewer parameters as problem complexity grows, compared to classical models. Additionally, HQNNs adapt well to the growing problem complexity since the rate of increase in FLOPs and parameter count (while going from low to high complexity of the problem) is significantly lower in HQNNs compared to classical models. These findings suggest that HQNNs are better suited to adapting to complex problem scenarios, highlighting their potential advantages over classical models in managing computational complexity.

(A3) Given that both independent measures of complexity, i.e., FLOPs and parameter count demonstrate better adaptability in HQNNs to problem complexity over classical networks, we posit that there is strong evidence suggesting that the inclusion of quantum layers offers a computational advantage in NNs. We also advocate that further investigations using additional complexity measures are needed to fully understand the QML benefits over classical ML. Thus, we leave this question partially open for future research to explore

with a broader set of metrics.

In summary, HQNNs offer a scalable, resource-efficient alternative to classical models, positioning them as a promising solution for complex tasks in machine learning.

REFERENCES

[1] M. Schuld *et al.*, "An introduction to quantum machine learning," *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, 2015.

[2] Y. Zhang and Q. Ni, "Recent advances in quantum machine learning," *Quantum Engineering*, vol. 2, no. 1, p. e34, 2020.

[3] D. Peral-García *et al.*, "Systematic literature review: Quantum machine learning and its applications," *Computer Science Review*, vol. 51, p. 100619, 2024.

[4] K. Zaman, A. Marchisio, M. A. Hanif, and M. Shafique, "A survey on quantum machine learning: Current trends, challenges, opportunities, and the road ahead," *CoRR*, vol. abs/2310.10315, 2024. [Online]. Available: https://arxiv.org/abs/2310.10315

[5] A. Abbas *et al.*, "The power of quantum neural networks," *Nature Computational Science*, vol. 1, no. 6, pp. 403–409, 2021.

[6] M.Kashif *et al.*, "The impact of cost function globality and locality in hybrid quantum neural networks on nisq devices," *Machine Learning: Science and Technology*, vol. 4, no. 1, 2023. [Online]. Available: https://dx.doi.org/10.1088/2632-2153/acb12f

[7] M. Kashif *et al.*, "Design space exploration of hybrid quantum–classical neural networks," *Electronics*, vol. 10, no. 23, p. 2980, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/23/2980

[8] K. Zaman *et al.*, "Studying the impact of quantum-specific hyperparameters on hybrid quantum-classical neural networks," *arXiv:2402.10605*, 2024. [Online]. Available: https://arxiv.org/abs/2402.10605

[9] M. Kashif, E. Sychiuco, and M. Shafique, "Investigating the effect of noise on the training performance of hybrid quantum neural networks," in *2024 International Joint Conference on Neural Networks (IJCNN)*. IEEE, Jun. 2024, p. 1–10. [Online]. Available: http://dx.doi.org/10.1109/IJCNN60899.2024.10651363

[10] M. Kashif and S. Al-Kuwari, "The unified effect of data encoding, ansatz expressibility and entanglement on the trainability of hqnns," *International Journal of Parallel, Emergent and Distributed Systems*, vol. 38, no. 5, pp. 362–400, 2023. [Online]. Available: https://doi.org/10.1080/17445760.2023.2231163

[11] D. Bokhan, A. S. Mastiukova, A. S. Boev, D. N. Trubnikov, and A. K. Fedorov, "Multiclass classification using quantum convolutional neural networks with hybrid quantum-classical learning," *Frontiers in Physics*, vol. 10, p. 1069985, 2022.

[12] M. Kashif and S. Al-Kuwari, "Demonstrating quantum advantage in hybrid quantum neural networks for model capacity," in *2022 IEEE International Conference on Rebooting Computing (ICRC)*, 2022.

[13] M. Kashif and M. Shafique, "Hqnet: Harnessing quantum noise for effective training of quantum neural networks in nisq era," *arXiv:2402.08475*, 2024. [Online]. Available: https://arxiv.org/abs/2402.08475

[14] L. Domingo *et al.*, "Binding affinity predictions with hybrid quantum-classical convolutional neural networks," *Scientific Reports*, vol. 13, no. 1, p. 17951, 2023.

[15] Y. Dong *et al.*, "An improved hybrid quantum-classical convolutional neural network for multi-class brain tumor mri classification," *Journal of Applied Physics*, vol. 133, no. 6, 2023.

[16] L. Domingo, M. Chehimi, S. Banerjee, S. H. Yuxun, S. Konakanchi, L. Ogunfowora, S. Roy, S. Selvaras, M. Djukic, and C. Johnson, "A hybrid quantum-classical fusion neural network to improve protein-ligand binding affinity predictions for drug discovery," *arXiv preprint arXiv:2309.03919*, 2023.

[17] S. Shi *et al.*, "Hybrid quantum-classical convolutional neural network for phytoplankton classification," *Frontiers in Marine Science*, vol. 10, p. 1158548, 2023.

[18] N. I. andothers, "Qfnn-ffd: Quantum federated neural network for financial fraud detection," 2024. [Online]. Available: https://arxiv.org/abs/2404.02595

[19] Z. Kaseb, M. Möller, G. T. Balducci, P. Palensky, and P. P. Vergara, "Quantum neural networks for power flow analysis," *Electric Power Systems Research*, vol. 235, p. 110677, 2024.

[20] H.-Y. Chen *et al.*, "Deep q-learning with hybrid quantum neural network on solving maze problems," *Quantum Machine Intelligence*, vol. 6, no. 1, p. 2, 2024.

[21] A. Wang *et al.*, "Shallow hybrid quantum-classical convolutional neural network model for image classification," *Quantum Information Processing*, vol. 23, no. 1, p. 17, 2024.

[22] E. Paquet *et al.*, "Quantumbound–interactive protein generation with one-shot learning and hybrid quantum neural networks," *Artificial Intelligence Chemistry*, vol. 2, no. 1, p. 100030, 2024.

[23] F. Arute *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[24] H.-S. Zhong *et al.*, "Quantum computational advantage using photons," *Science*, vol. 370, no. 6523, p. 1460–1463, Dec 2020. [Online]. Available: http://dx.doi.org/10.1126/science.abe8770

[25] Y. Wu *et al.*, "Strong quantum computational advantage using a superconducting quantum processor," *Phys. Rev. Lett.*, vol. 127, 2021. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.127.180501

[26] L. S. Madsen and F. e. Laudenbach, "Quantum computational advantage with a programmable photonic processor," *Nature*, vol. 606, no. 7912, pp. 75–81, 2022. [Online]. Available: https://doi.org/10.1038/s41586-022-04725-x

[27] J. Bowles *et al.*, "Better than classical? the subtle art of benchmarking quantum machine learning models," *arXiv:2403.07059*, 2024. [Online]. Available: https://arxiv.org/abs/2403.07059

[28] M. Schuld and N. Killoran, "Is quantum advantage the right goal for quantum machine learning?" *PRX Quantum*, vol. 3, p. 030101, Jul 2022. [Online]. Available: https://link.aps.org/doi/10.1103/PRXQuantum.3.030101

[29] J. Preskill, "Quantum computing in the NISQ era and beyond," *Quantum*, aug 2018. [Online]. Available: https://doi.org/10.22331%2Fq-2018-08-06-79

[30] J. Schnabel and M. Roth, "Quantum kernel methods under scrutiny: A benchmarking study," *arXiv:2409.04406*, 2024.

[31] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[32] Z. Lu *et al.*, "The expressive power of neural networks: A view from the width," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[33] Y. Lee *et al.*, "Neural complexity measures," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 9713–9724.

[34] L. Szymanski *et al.*, "Conceptual complexity of neural networks," *Neurocomputing*, vol. 469, pp. 52–64, 2022.

[35] J. Holstermann, "On the expressive power of neural networks," *arXiv preprint arXiv:2306.00145*, 2023.

[36] D. A. Ehrlich *et al.*, "A measure of the complexity of neural representations based on partial information decomposition," *Transactions on Machine Learning Research*, 2023.

[37] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the International Conference on Machine Learning*, 2019, p. 6105–6114.

[38] Z. Guo *et al.*, "Flops-efficient filter pruning via transfer scale for neural network acceleration," *Journal of Computational Science*, vol. 55, p. 101459, 2021.

[39] S.-C. Hsia *et al.*, "Convolution neural network with low operation flops and high accuracy for image recognition," *Journal of Real-Time Image Processing*, vol. 18, no. 4, pp. 1309–1319, 2021.

[40] J. Chen *et al.*, "Run, don't walk: Chasing higher flops for faster neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 12 021–12 031.

[41] I. Goodfellow *et al.*, *Deep learning*. MIT press, 2016.

[42] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 8, pp. 1553–1565, 2014.

[43] Y. Liu *et al.*, "Synthetic benchmarks for scientific research in explainable machine learning," *arXiv:2106.12543*, 2021. [Online]. Available: https://arxiv.org/abs/2106.12543

[44] P. Orzechowski and J. H. Moore, "Generative and reproducible benchmarks for comprehensive evaluation of machine learning classifiers," *Science Advances*, vol. 8, no. 47, p. eabl4747, 2022.

[45] R. LaRose and B. Coyle, "Robust data encodings for quantum classifiers," *Phys. Rev. A*, vol. 102, p. 032420, Sep 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.102.032420

[46] V. Bergholm *et al.*, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," *arXiv*, 2018. [Online]. Available: https://arxiv.org/abs/1811.04968