The Effects of Conditional Pooling Techniques in Quanvolutional Circuits of Quantum-Classical Hybrid Neural Networks

Robin Faier[1,*], PD Dr. habil. Jeanette Miriam Lorenz[2] and Hans Ehm[3]

[1]Department of Physics, Ludwig Maximilians Universität, Munich, Germany
[2]Fraunhofer Institute for Cognitive Systems IKS, Munich, Germany
[3]Infineon Technologies AG, Munich, Germany

Abstract

This paper explores the performance of quantum-classical hybrid networks in image classification tasks, focusing on the integration of quantum circuits as alternative feature extractors to traditional convolutional layers. Specifically, it investigates the use of quanvolutional layers, variational quantum circuits that leverage quantum entanglement and quantum gates, in comparison to classical layers. The study examines various quantum pooling techniques, including conditional entanglement gates, and their impact on classification accuracy across datasets with varying complexity. By experimenting with different pooling strategies, both parametrized and non-parametrized, this work assesses their influence on network performance and feature representation. Results indicate that quanvolutional layers in a hybrid network consistently outperform classical convolutional layers in terms of classification accuracy, particularly when applied to datasets with prominent features. Additionally, the findings suggest that quantum entanglement techniques, rather than rotation parameters, play a more significant role in enhancing performance. This paper concludes that quantum-classical hybrid networks offer a promising approach for feature extraction, although the optimal configuration of pooling methods depends on the characteristics of the data. Future research could further explore the interplay between quantum circuits and different pooling strategies for more effective feature representation.

Keywords

Quantum Machine Learning, Quantum Classical Hybrid Neural Network, Quantum Convolutional Neural Networks, Quanvolutional Neural Networks, Quantum Pooling

1. Introduction

In recent years, quantum computing has emerged as a promising avenue for enhancing machine learning models [1, 2], particularly in the domain of image classification. Classical convolutional neural networks (CNNs) [3] have achieved remarkable success in visual tasks by leveraging convolutional layers to detect hierarchical patterns in images. However, these classical networks often struggle with scalability and expressiveness of data, especially when dealing with complex data or large feature spaces [4]. As quantum computing continues to evolve, there has been growing interest in integrating quantum circuits in classical machine learning frameworks to create hybrid models with better feature expression [5]. Quantum-classical hybrid networks are a promising approach that utilize quantum circuits for feature extraction, while relying on classical layers for tasks like decision-making and classification [6].

One such quantum-classical architecture is the quanvolutional network, where quantum circuits, particularly those employing quantum entanglement and quantum gates, serve as an alternative to traditional convolutional layers [5]. These quantum circuits can potentially provide enhanced feature representation by exploiting quantum properties such as superposition and entanglement to generate distributions that are hard to produce for classical computers [6, 7]. However, the precise configuration

AIQxQIA 2025: International Workshop on AI for Quantum and Quantum for AI | co-located with ECAI 2025, Bologna, Italy
*Corresponding author.
robin.faier@campus.lmu.de (R. Faier); jeanette.miriam.lorenz@iks.fraunhofer.de (P. Dr. habil. J. M. Lorenz); hans.ehm@infineon.com (H. Ehm)
0000-0001-6530-1873 (P. Dr. habil. J. M. Lorenz)

of quantum circuits, including the selection of pooling methods, entanglement techniques, and rotation parameters, remains an open question [8].

This paper explores the performance of quanvolutional layers in quantum-classical hybrid networks, comparing them with classical convolutional layers in the context of image classification tasks. Specifically, we investigate various pooling techniques, including conditional and multi-conditional entanglement methods such as CNOT, CCNOT, and CCCNOT, and examine how these pooling strategies impact the networks' ability to learn and classify features from different datasets. Our focus is on the interplay between quantum entanglement, circuit parametrization, and the overall classification accuracy across datasets with varying levels of feature complexity. The goal of this study is to provide insights into the effectiveness of quantum-classical hybrid networks, especially in the context of quanvolutional layers, and to explore how different pooling methods and training parameters affect the network's performance. Additionally, we aim to understand the conditions under which quantum circuits may provide an advantage over classical methods and identify potential avenues for future research.

2. Background

2.1. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a widely used class of deep learning models specifically designed for image and grid-like data [9, 10]. They address the inefficiency of fully connected networks for high-dimensional inputs by leveraging local connectivity and weight sharing, thus significantly reducing the number of parameters [3], which would otherwise increase quadratically for a two-dimensional input. A typical CNN architecture consists of convolutional layers for feature extraction [3] and often alternating pooling layers for significant size reduction, followed by one or more fully connected layers for feature assessment and classification. Non-linear activation functions such as ReLU [11] are applied after each layer to enhance the model's capacity to capture complex patterns.

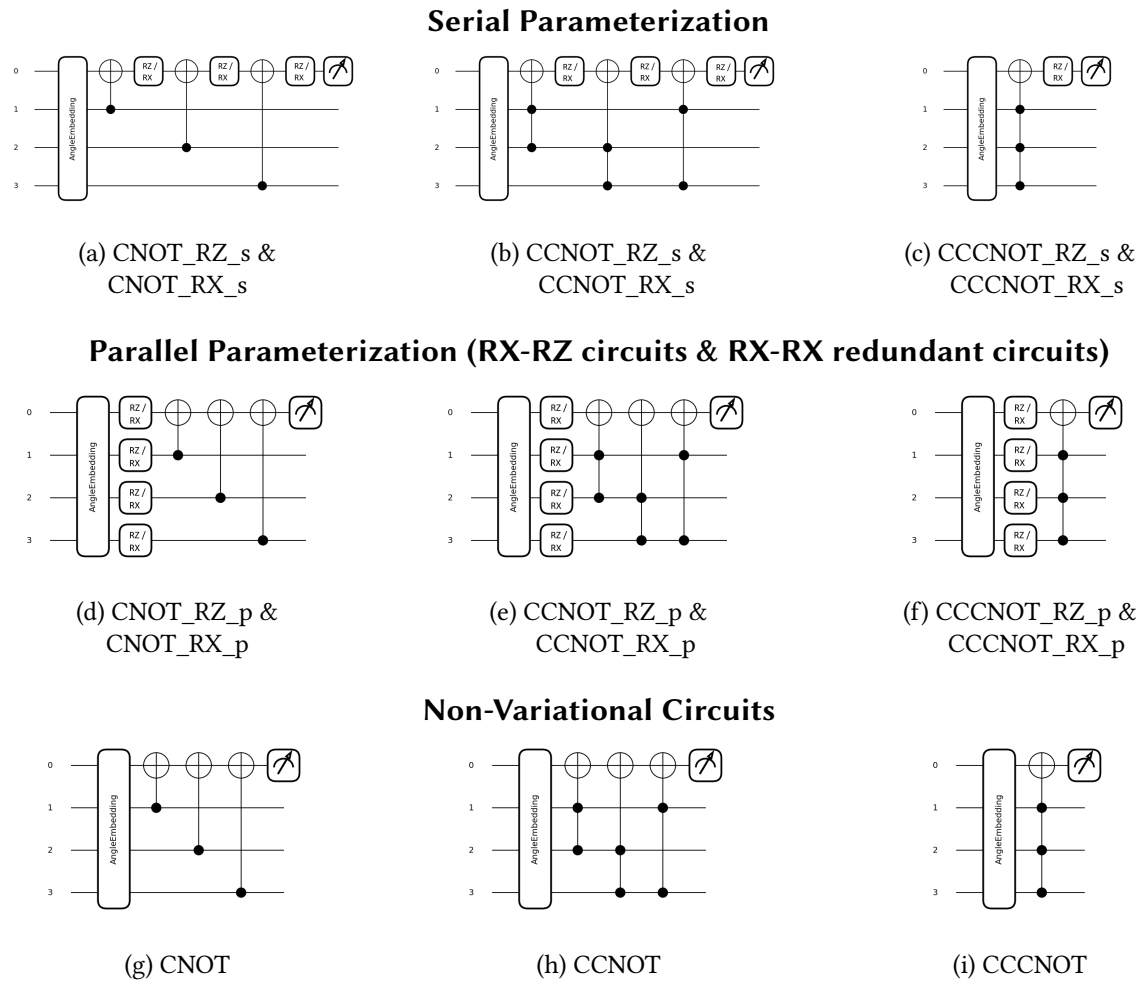
In this work, the convolutional layer serves as a classical baseline for comparison with quanvolutional layers. Both use a 2×2 kernel and reduce the input resolution by a factor of two, enabling a direct evaluation of their feature extraction capabilities within a hybrid quantum-classical architecture.

2.2. Quanvolution

First proposed by Henderson [5], the quanvolutional layer is the quantum computational equivalent of the convolutional layer in classical networks. This layer aims to locally convolve data while ensuring that its parameterization remains translationally invariant [3]. The concept is based on the assumption that multiple similar patterns within the data combine to form more complex features, which can be detected locally [10]. By conserving filter parameters, the number of required parameters is reduced, as the number of parameters is determined by the filter size rather than the dataset size.

The same principle applies to the quanvolutional layer [5], which can be constructed with smaller filters and, consequently, smaller circuits. For two-dimensional data, such as images, at least four panels are required, as this is the minimum number of neighboring elements that can be convolved. Each panel information is embedded on one of four qubits, altered by parameterized gates, and entangled with the others. The appropriate embedding of data in the quantum layer and parameterization in the variational quantum circuit are ongoing subjects of research [12, 7], with both empirical [5, 13] and theoretical [7] approaches available. Given the non-quantum nature of the data used and the lack of a reliable transformation to a quantum circuit, the rationale for studying purely quantum networks at the current state is questionable [14].

Therefore, this work focuses on investigating the performance of a quantum-classical hybrid neural network, specifically exploring the quantum subroutine that replaces the convolutional layer with a quanvolutional one. The study aims to assess not only the effect of parameterization within the quantum circuits but also how effectively the classical network can adapt to and exploit the quantum-generated features. In this sense, the classical network self-optimizes the information flow between the quantum



(a) CNOT_RZ_s & CNOT_RX_s

(b) CCNOT_RZ_s & CCNOT_RX_s

(c) CCCNOT_RZ_s & CCCNOT_RX_s

(d) CNOT_RZ_p & CNOT_RX_p

(e) CCNOT_RZ_p & CCNOT_RX_p

(f) CCCNOT_RZ_p & CCCNOT_RX_p

(g) CNOT

(h) CCNOT

(i) CCCNOT

Figure 1: The subfigures display the constructions of the various pooling methods tested. The first row shows serially parameterized circuits, the second row shows parallel parameterization (including RX-RX redundant circuits), and the third row shows non-variational pooling circuits without any trainable parameters. Each serial and parallel pooling circuit is tested twice: once with X-rotation gates (RX) and once with Z-rotation gates (RZ), as indicated in the subfigure labels and circuit names. These circuits are referred to as single-pooling methods throughout this work, as they use only one type of entangling gate per circuit. The target qubit (the top wire in all diagrams) collects information from all input qubits via entanglement and is measured to obtain a reduced output dimension. In addition to the different parameter gates, the circuits in subfigures (a), (d), and (g) all implement CNOT pooling, while those in subfigures (b), (e), and (h) use CCNOT pooling, and those in subfigures (c), (f), and (i) implement CCCNOT pooling. Every parametrized gate has an individual trainable weight.

and classical parts to improve overall classification performance. The primary scientific question is: Can a classification network benefit from a self-trained quantum subroutine, and are there meaningful differences in performance for various subroutines?

2.3. Quantum Pooling

Pooling layers are commonly employed in classical machine learning to reduce the spatial dimensions of large data, such as high-resolution images, while simultaneously improving computational efficiency. Different pooling techniques impose distinct methods for contracting the input, such as average pooling, which computes the average of the inputs, and max pooling, which selects the highest input value.

This concept of input condensation is adapted in quantum machine learning, where quantum pooling techniques, based on variational circuits, are explored [13, 15]. In these circuits, multiple input qubits are entangled with a designated target qubit such that the final state of the target reflects correlations

among the full register. The target qubit encodes an effective representation of the entire input block through its entangled state. Thus, when this qubit is measured, the outcome probabilistically represents a compressed summary of all the qubits' states, serving as a quantum pooling mechanism. The entire qubit system spans a higher-dimensional Hilbert space (here $2^4 = 16$ dimensions), but by entangling all qubits and measuring only a single target qubit, the effective representation is reduced to the 2-dimensional space of the measured qubit [16]. In this sense, the pooling operation reduces the number of qubits while preserving relevant information for classification. The goal of reducing spatial dimensions and enhancing network efficiency by condensing information from multiple inputs into a more informative output remains consistent [17]. Unlike classical pooling methods such as max or average pooling, which directly aggregate numerical values, quantum pooling indirectly compresses input information through entanglement and measurements. Some quantum pooling methods have shown more stable training processes and improved prediction accuracy compared to unpooled quanvolutional circuits [13]. In this work, we further investigate simple entanglement strategies and their advantages within the context of quanvolutional pooling circuits for quantum-classical hybrid networks.

2.4. Related Work

Quantum convolutional architectures have been studied from both empirical and theoretical perspectives. Cong et al. [18] introduced quantum convolutional neural networks (QCNNs) based on multiscale entanglement structures, showing strong performance in quantum phase recognition. Pesah et al. [16] demonstrated that such architectures are not affected by barren plateaus, supporting their practical trainability.

In more resource-constrained applications, Song et al. [19] proposed a resource-efficient QCNN variant (RE-QCNN) adapted to classical image datasets like MNIST, while Chinzei et al. [20] introduced an equivariant split-parallel QCNN (sp-QCNN) that exploits symmetry for enhanced scalability.

Regarding pooling strategies, Monnet et al. [8] and Hur et al. [21] developed modulated pooling layers combining entanglement and trainable gates. Their work inspired some of the advanced pooling circuits examined in this study. Schuld et al. [7] analyzed how data encoding and circuit topology jointly determine the expressive capacity of quantum models, emphasizing the importance of architectural choices such as embedding schemes and entanglement layout.

Hybrid approaches beyond convolutional architectures have also explored pooling-like ideas for dimensionality reduction. For instance, QUARTA [22], equivariant QCNNs with embedding-dependent pooling structures [23], and interaction layer QCNNs using three-qubit gates [24] highlight how diverse architectural extensions can enhance hybrid quantum models. In contrast, our work provides a systematic empirical study of pooling strategies themselves, isolating their role within quantum-classical hybrid networks.

While these studies provide valuable architectural and theoretical insights, our work contributes a systematic empirical comparison of simple pooling strategies (CNOT, CCNOT, CCCNOT), with and without parameterization, in a hybrid quantum-classical setting. Furthermore, we examine combinations of simple pooling methods, such as CNOT+CCNOT or full CNOT+CCNOT+CCCNOT compositions, to investigate whether increasing the entanglement complexity leads to improved feature extraction. To our knowledge, such a direct comparison of both entanglement structure and parametrization, across multiple datasets and pooling configurations, has not been previously reported.

3. Methodology

3.1. Data

Three datasets are used to analyze the performance of the hybrid networks and assess the quanvolutional capabilities of the circuits for different features. The datasets vary in complexity, the number of features, and the representational ability of features with respect to the number of classes. To gain deeper insights into the impact of a quanvolutional pooling layer on feature perception, two binary-class and one

multi-class classification dataset are used. The first dataset is the BreastMNIST [25] (Set1) from the MedMNIST collection [26], which consists of images of malignant and benign tissues to be classified by the network. The second dataset, Set2, is a reduced version of the MNIST [27] dataset, where only the digits five and seven are to be recognized. The third dataset, Set3, is the full MNIST [27] dataset, containing ten classes of images of handwritten digits from zero to nine. All images are resized to 28x28 pixels for computational efficiency while retaining the most critical features for recognition. The two binary-class datasets are evaluated with training set sizes of 200 and 400 samples to investigate the sensitivity of quantum pooling circuits to limited data availability and to analyze performance trends as the number of training samples increases. This setup reflects practical use cases where (quantum) models must operate effectively under data-scarce conditions, while also enabling a comparative analysis of circuit behavior across different data regimes. For the multi-class dataset (Set3), a larger sample size of 10,000 was chosen to ensure sufficient statistical resolution across ten output classes and to balance the higher variation in circuit configurations with resource constraints during simulation. These datasets provide a range of feature complexity and numbers of classes, offering an opportunity to investigate how quanvolutional pooling layers affect both binary and multi-class classification.

3.2. Quantum-Classical Hybrid Network

The quantum-classical hybrid network consists of a quantum circuit, where pixel information is encoded via angle embedding using X-rotations, followed by a two-layer classical linear model to evaluate the circuit measurements. To map the pixel information onto the qubits, all inputs are normalized to the range $[0,\ 2\pi]$ prior to rotation. The hybrid network is built using Pennylane [28] for the quanvolutional layer and Pytorch [29] for the overall network architecture. Training of the circuit parameters is performed using the Pennylane parameter-shift rule, which involves evaluating the continuous expectation value of the circuit measurement. ADAM [30] is employed as the optimizer, and cross-entropy is used as the loss function for classification. The training for all quanvolutional layers and the convolutional layer is repeated ten times, each with different randomized initial parameters, to ensure that all observations are independent of the initial parameter settings.

3.3. Pooling Techniques in Quanvolutional layers

The pooling methods we focus on are based on conditional NOT gates (CNOT) and multi-conditional NOT gates (CCNOT and CCCNOT/Toffoli), with and without parametrization. Variations in parameters are introduced using trainable RX- and RZ-gates, applied both in parallel and serially within the circuits. In parallel configuration, each qubit is individually parametrized before entanglement, while in serial configuration, the target qubit is parametrized between entanglements. To better understand the effect of entanglement on pooling, independent of parametrization, a non-variational circuit is also used. In this non-variational setup, four neighboring pixels are encoded as input on four qubits, which are then entangled to produce one output value, similar to a convolution layer with a 2×2 filter. These circuits, which consist of a single type of entanglement gate, are referred to as single-pooling methods and are illustrated in Figure 1. The three circuits CNOT_RX_p, CCNOT_RX_p, and CCCNOT_RX_p apply an R_X embedding directly followed by an R_X parameterization, potentially introducing redundant transformations on individual qubits. This redundancy arises because two consecutive R_X rotations on the same qubit axis effectively combine into a single rotation with the sum of both angles. In this configuration, the trainable rotation does not interact with a new degree of freedom but merely acts as a constant shift relative to the embedded input. Since this shift occurs individually on each qubit, it can unintentionally modulate the encoded input amplitudes in a non-informative way and thereby reduce the circuit's sensitivity to actual input. We refer to these as RX-redundant circuits throughout this work, as their structure may interfere with the interpretability and informativeness of the encoded features.

More advanced circuits, known as mixed poolings, are constructed by applying multiple types of entanglement gates sequentially. All mixed pooling circuits are purely entanglement-based, with no parameterized gates introduced. For comparison, we also adapt the modulated pooling circuits from [8],

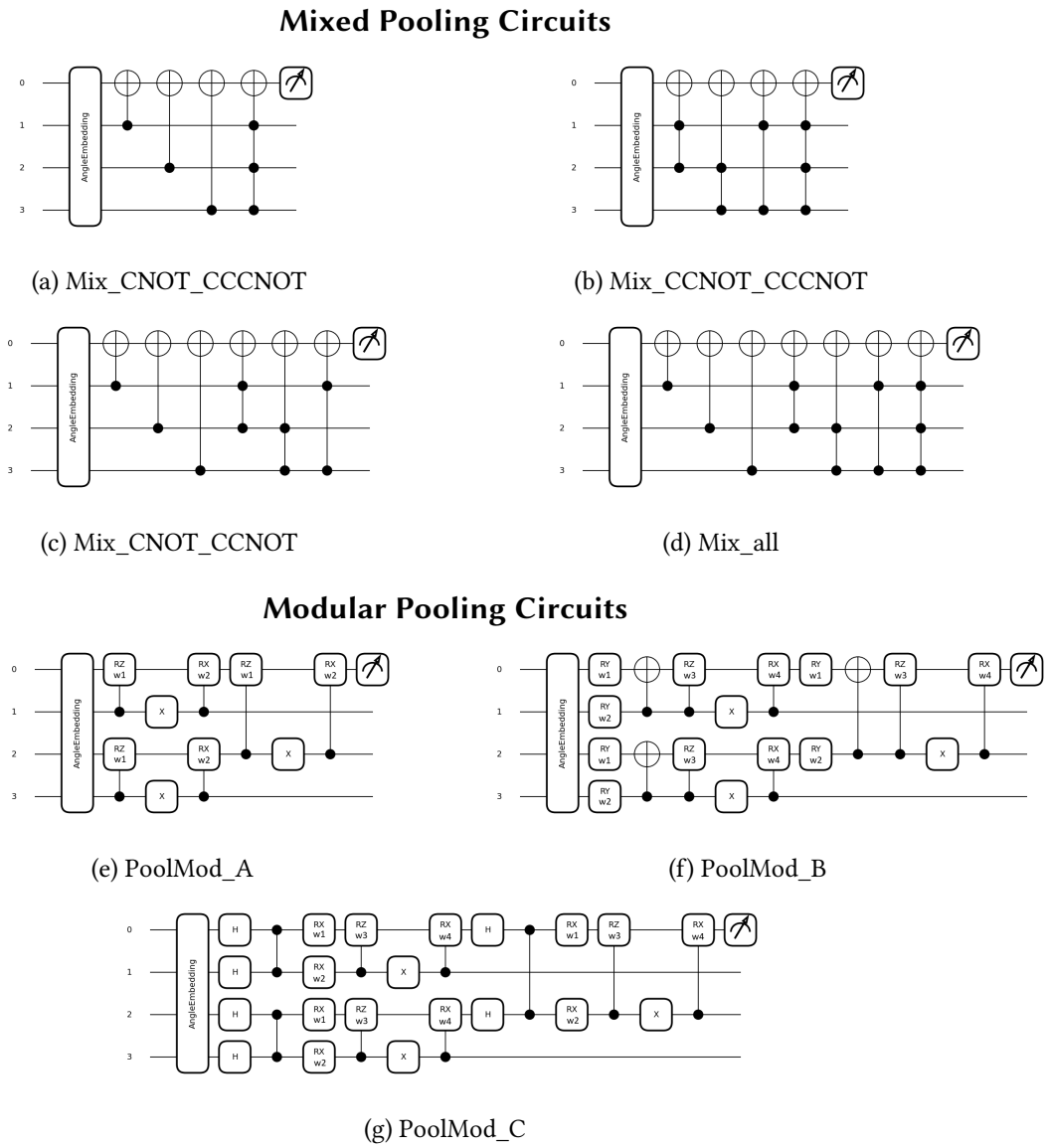


Figure 2: The subfigures illustrate the constructions of various advanced pooling strategies, which we refer to as multi-pooling. Subfigures (a)–(d) display the mixed pooling circuits Mix_CNOT_CCCNOT, Mix_CCNOT_CCCNOT, Mix_CNOT_CCNOT, and Mix_all, respectively, each sequentially combining different single pooling techniques. Subfigures (e)–(g) show modulated pooling circuits adapted from [8], where a single entanglement module is repeated across qubits. PoolMod_A (e) implements the simplest module using only one conditional R_Z and R_X gate and an X gate on the second qubit. PoolMod_B (f) introduces two additional R_Y gates and one CNOT per module, while PoolMod_C (g) adds two Hadamard gates, a CZ gate, and more complex parameterization, with two different RX gates. Identical parameter names (e.g., $w1$, $w2$) are reused across modules to reflect shared weights, as proposed in the original design [8]. These circuits aim to encode spatially structured transformations with fewer parameters by repeating entanglement templates.

which are inspired by the work in [21]. These advanced circuits, which we refer to as multi-pooling, are shown in Figure 2.

3.4. Classical Part of the Hybrid Network

The overall architecture is shown in Figure 3 and remains consistent across all tested networks. Following the quanvolutional layer, two linear layers are applied sequentially, with a ReLU activation function applied between them. The output of the second layer is evaluated using the cross-entropy loss function,

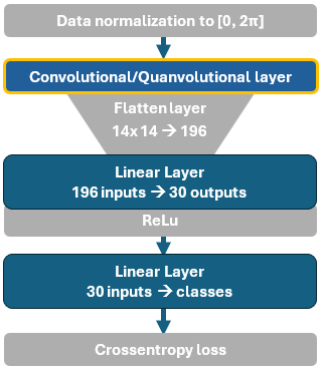


Figure 3: The network architecture illustrates the layers responsible for training on the data. Initially, the 28×28 data arrays are normalized to the range $[0, 2\pi]$, before being processed by either the convolutional or quanvolutional layer to extract features. The resulting output, sized 14×14, is then flattened into a vector of 196 elements, which serves as the input to the first linear layer containing 30 nodes. A Rectified Linear Unit (ReLU) activation function is applied to the 30 outputs before they are passed through the second linear layer, which consists of 2 nodes for classification in Set1 and Set2, or 10 nodes for Set3. Finally, the output is evaluated using the cross-entropy loss function.

depending on the number of classes in the respective dataset. This setup is compared to a standard convolutional network, where a convolutional layer replaces the quanvolutional layer. For consistency, the convolutional layer uses a 2×2 filter kernel and includes a bias term, resulting in a total of five trainable parameters ($2 \times 2 + 1 = 5$).

In contrast, the quanvolutional layers differ in parameter count depending on the pooling type of the circuit:

- **Non-variational circuits** (Non-Variational Circuits in Figure 1, as well as all Mixed Pooling Circuits in Figure 2) contain **0 parameters**.
- **Serially parameterized circuits** (Serial Parametrization in Figure 1) include one parameter per entanglement step, summing up to **3 parameters**.
- **Parallelly parameterized circuits** (Parallel Parametrization in Figure 1) feature one parameter per qubit after the embedding, totaling **4 parameters**.
- **Modulated pooling circuits** (Modular Pooling Circuits in Figure 2) also have **4 parameters**.

Overall, the quanvolutional circuits reduce the number of trainable parameters compared to the classical convolutional layer. However, since all quantum-classical hybrid models are evaluated using the `default.qubit` simulator in PennyLane, the focus of the evaluation is the final classification accuracy of the networks, rather than computational efficiency.

To maintain consistent spatial reduction, both classical and quantum models apply zero padding and a stride of two, reducing the image size from 28×28 to 14×14, analogous to merging a 2×2 patch into a single output value.

3.5. Training Setup

All models were implemented using the PennyLane [28] framework for the quantum circuits and PyTorch [29] for the classical layers. Training was performed using the Adam optimizer [30] with a learning rate of 0.001 and a batch size of 16. Each network was trained for 10 epochs using the cross-entropy loss function. All input images were normalized to the range $[0, 2\pi]$ and mapped to rotation angles via angle encoding.

Circuit parameters were initialized using a normal distribution with a mean of zero and a standard deviation of 1. To ensure that the results are not biased by initial conditions, each experiment was repeated ten times with different random seeds, which were drawn from a tensor initialized with a

fixed master seed (torch.manual_seed(0)) to ensure reproducibility. The reported validation accuracies represent the mean and standard deviation across these ten runs.

To assess the robustness of the results with respect to the batch size, we performed an ablation study using batch sizes of 8, 16, 32, and 64 on a subset of circuits presented in the Appendix section A. The validation accuracy varied only marginally (typically less than 2 percentage points), and the relative ranking of circuit performance remained consistent across batch sizes. This indicates that the observed performance trends are not specific to a particular choice of batch size. A systematic study of the learning rate was not conducted.

4. Results

This chapter presents a detailed analysis of the experimental results obtained from evaluating the various quantum pooling circuits across the chosen datasets. The plots shown highlight the training dynamics, validation performance, and comparative effectiveness of the best, worst, and average pooling strategies compared to their classical counterpart for better visibility. Error bars represent the corresponding standard deviations over the 10 epochs, and the final validation accuracy is depicted with the standard deviation in the plots' legends for better comparison.

4.1. Single Pooling Techniques

As a first step, we evaluate circuits with single pooling configurations to establish a performance baseline. This allows us to isolate and compare the effects of individual pooling strategies before exploring more complex combinations.

4.1.1. Set1

The validation accuracy of the circuits for Set1 with 200 images is shown in Figure 4. Overall, the networks achieve an average classification accuracy of 70.46%, which is only marginally better than the classical convolutional baseline at 70.30%. Hybrid networks, however, exhibit significantly smaller error margins, 0.499 ± 0.055 on average, more than a factor of ten lower in loss variance, and about half the mean loss compared to the classical model (1.093 ± 0.784), although this improvement is not reflected in the final classification accuracy. Notably, the RX-redundant CCCNOT_RX_p starts with the lowest initial mean accuracy but surpasses all other circuits during training, achieving the highest final accuracy (73.94%), albeit with the largest standard deviation (6.94%). In contrast, the CCNOT_RZ_s circuit achieves the lowest final accuracy (65.76%) despite showing the best initial performance. Extending the number of training images from 200 to 400 for Set1 leads to only a minor improvement in validation accuracy for the classical convolutional model, while the average performance of the quanvolutional circuits slightly decreases, as shown in Figure 5. Once again, quantum models show significantly lower validation losses with smaller variance (0.559 ± 0.042) compared to the classical baseline (0.974 ± 0.642), though this advantage does not consistently translate to accuracy. Notably, the standard deviation in accuracy increases for all models over the 10 epochs, whereas the loss deviations remain stable.

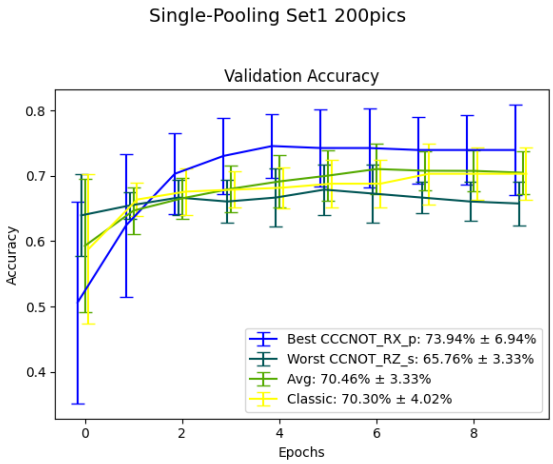


Figure 4: This figure shows the validation accuracy over 10 training epochs for the best, worst, and average single-pooling quanvolutional circuits, along with a classical convolutional baseline, trained on Set1 with 200 images. The best quantum circuit reaches a final validation accuracy of $73.94\% \pm 6.94\%$. The worst-performing circuit ends at an accuracy of $65.76\% \pm 3.33\%$, while the average performance across quantum circuits is $70.46\% \pm 3.33\%$ in accuracy. The classical network achieves $70.30\% \pm 4.02\%$ accuracy. Error bars indicate the standard deviation across 10 runs, showing that quantum circuits can achieve both higher accuracy and lower variance.

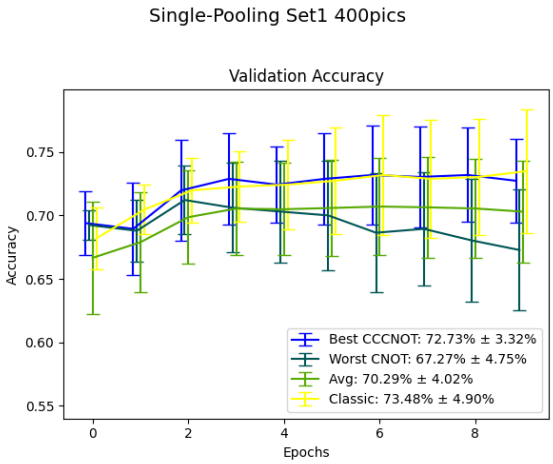


Figure 5: This figure shows the validation accuracy over 10 training epochs for the best, worst, and average single-pooling quanvolutional circuits, along with a classical convolutional baseline, trained on Set1 with 400 images. The best quantum circuit reaches a final validation accuracy of $72.73\% \pm 3.32\%$. The worst-performing circuit ends at an accuracy of $67.27\% \pm 4.75\%$, while the average performance across quantum circuits is $70.29\% \pm 4.02\%$ in accuracy. The classical network achieves $73.48\% \pm 4.90\%$ accuracy. Error bars indicate the standard deviation across 10 runs, showing that quantum circuits can achieve both higher accuracy and lower variance.

The best-performing circuit is the CCCNOT with an accuracy of 72.73% ±3.32%, while the worst circuit (CNOT) ends at 67.27%, despite both starting from similar initial accuracy around 69%. On average, the quantum models reach an accuracy of 70.29% ±4.02%, slightly below the classical result of 73.48% ±4.90%, though the overlapping error margins suggest comparable performance.

4.1.2. Set2

Training the quantum-classical hybrid networks on 200 images from Set2 training on the digits 5 and 7 highlights CCCNOT_RX_s as the best-performing circuit, as shown in Figure 6. Most other pooling methods perform similarly, as reflected by the closely grouped training curves and a high average final accuracy of 88.75%. The RX-redundant circuit CNOT_RX_p yields the lowest performance with a validation accuracy of 76.67% and also exhibits the largest standard deviation among the quanvolutional models at 9.39%. The classical convolution reaches a comparable accuracy of 77.88%, but with a much higher standard deviation of 21.77%, indicating less consistent performance compared to the quantum models.

In contrast to the single-pooling results on Set1, the accuracy dynamics here display a clearer anti-correlation with the loss: circuits with high variance in loss also show larger fluctuations in accuracy, and the relative performance across models is more consistent between both metrics.

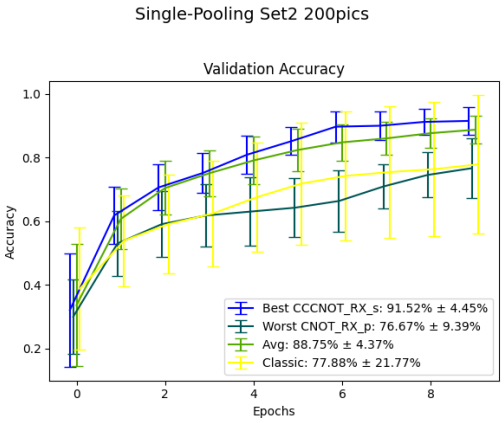


Figure 6: This figure shows the validation accuracy over 10 training epochs for the best, worst, and average single-pooling quanvolutional circuits, along with a classical convolutional baseline, trained on Set2 with 200 images. The best quantum circuit achieves a final validation accuracy of $91.52\% \pm 4.45\%$. The worst-performing circuit ends at an accuracy of $76.67\% \pm 9.39\%$, while the average quantum performance is $88.75\% \pm 4.37\%$ in accuracy. The classical baseline reaches $77.88\% \pm 21.77\%$ accuracy. Error bars indicate the standard deviation across 10 runs, showing that quantum circuits can achieve both higher accuracy and lower variance.

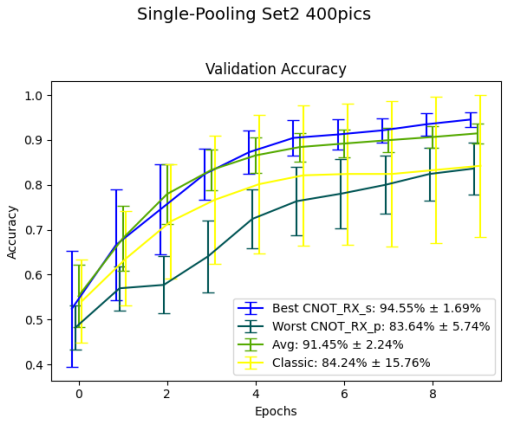


Figure 7: This figure shows the validation accuracy over 10 training epochs for the best, worst, and average single-pooling quanvolutional circuits, along with a classical convolutional baseline, trained on Set2 with 400 images. The best quantum circuit achieves a final validation accuracy of $94.55\% \pm 1.69\%$. The worst-performing circuit reaches $83.64\% \pm 5.74\%$ in accuracy. The average performance of the quantum circuits is $91.45\% \pm 2.24\%$ in accuracy. The classical baseline shows a final validation accuracy of $84.24\% \pm 15.76\%$. Error bars indicate the standard deviation across 10 runs, showing that quantum circuits can achieve both higher accuracy and lower variance.

When increasing the number of training images for Set2 from 200 to 400, a moderate overall gain in validation accuracy is observed as depicted in Figure 7, which is supported by a noticeable drop in validation loss. Once again, the quanvolutional networks outperform the classical convolutional model: their average validation accuracy rises to 91.45%, exceeding the classical result of 84.24% by more than 6 percentage points. All quantum circuits perform better than the classical baseline, except for the RX-redundant CNOT_RX_p circuit, which falls behind at 83.64% and thus marks a clear outlier.

The best-performing model is the CNOT_RX_s circuit, reaching a validation accuracy of 94.55% with the smallest standard deviation of 1.69%. While the classical model shows the strongest relative improvement compared to its result with 200 images, its standard deviation only decreases by about one quarter, whereas the hybrid models reduce their deviation by roughly half.

4.1.3. Set3

With Set3, which contains images of ten digits and 10,000 training images, the network performance changes significantly, as shown in Figure 8. While the feature complexity of Set3 is comparable to Set2, the classification task is considerably more difficult due to the presence of ten distinct classes. To account for this increased complexity, the training set size is raised to 10,000, enabling more robust and stable training outcomes.

The classical convolutional network performs reasonably well, with an average validation accuracy of 73.46%, but suffers from a large standard deviation of 31.09%. This indicates high sensitivity to initialization and unreliable convergence, as the model inconsistently learns relevant features. In contrast, the quanvolutional networks show more consistent results, with standard deviations below 1%, demonstrating high reliability regardless of initial parameters.

The differences in performance among the quantum circuits become more pronounced on this dataset: CCCNOT_RX_s performs substantially worse than its peers with only 59.78% accuracy, while CNOT_RX_s achieves the best result at 77.67%. This clear separation highlights the importance of the

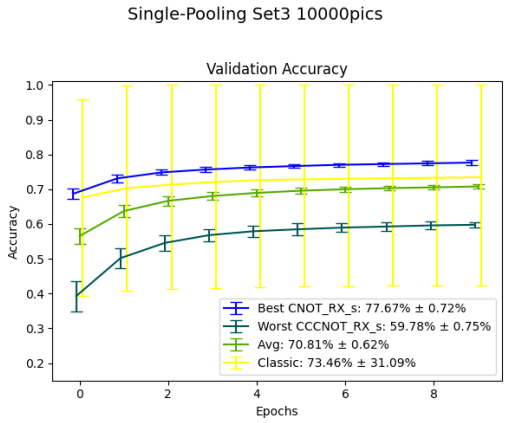


Figure 8: This figure shows the validation accuracy over 10 training epochs for the best, worst, and average single-pooling quanvolutional circuits, along with a classical convolutional baseline, trained on Set3 with 10,000 images. The best quantum circuit achieves a final validation accuracy of $77.67\% \pm 0.72\%$. The worst-performing circuit reaches $59.78\% \pm 0.75\%$ in accuracy. The average performance of the quantum circuits is $70.81\% \pm 0.62\%$ in accuracy. The classical baseline shows a final validation accuracy of $73.46\% \pm 31.09\%$. Error bars indicate the standard deviation across 10 runs, showing that quantum circuits can achieve both higher accuracy and lower variance.

chosen pooling technique. The quanvolutional average accuracy is 70.81%, which is still competitive. A corresponding evaluation of the validation loss curves supported these findings with aligned trends in means and standard deviations.

4.2. Multi Pooling Techniques

When analyzing the performance of the single-pooling methods, we found that the entanglement technique was the most decisive factor in achieving high classification accuracy. In contrast, the choice of angle embedding axis and parameterization had a comparatively minor impact. To preserve computational resources for the more demanding simulation of larger quantum circuits, we therefore focused our multi-pooling study primarily on the effect of combining different single-pooling variations. These combinations are compared against the established parameterized modular circuits proposed in [8] and [21]. A more detailed analysis that led to this design decision is presented in section 4.3.

4.2.1. Set1

When extending the pooling methods from single- to multi-pooling techniques, the average validation accuracy increases to 73.29%, further outperforming the classical convolutional baseline at 70.30%. The best-performing circuit for Set1 with 200 images (Figure 9) is the Mix_CNOT_CCCNOT, reaching 78.79% accuracy, clearly surpassing the best single-pooling method, CCCNOT_RX_p.

However, the Mix_all circuit performs poorly, achieving only 63.64%, which is not only lower than the classical setup and approximately 10% below the quantum average but also worse than the lowest-performing single-pooling circuit, CCNOT_RZ_s (65.76%). This indicates that mixing pooling strategies can both enhance and impair quanvolutional feature extraction, depending on the combination of techniques.

The validation loss follows a similar trend to the single-pooling analysis, with all quanvolutional circuits having slightly lower loss values overall, delivering no extra insights.

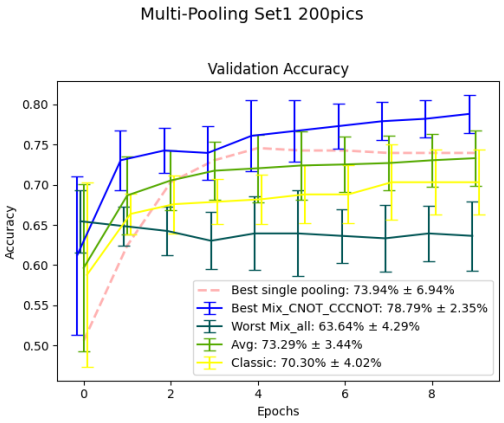


Figure 9: Validation accuracy for the best, worst, and average performing multi-pooling quantum circuits compared to a classical baseline, trained on Set1 using 200 images. The best multi-pooling model (Mix_CNOT_CCCNOT) achieves a final validation accuracy of $78.79\,\% \pm 2.35\,\%$, while the worst-performing circuit (Mix_all) reaches $63.64\,\% \pm 4.29\,\%$. The average across all multi-pooling circuits results in $73.29\,\% \pm 3.44\,\%$. For comparison, the classical model achieves $70.30\,\% \pm 4.02\,\%$. As an additional reference, the best single-pooling model (CCCNOT_RX_p) is shown as a dashed red curve and achieves $73.94\,\% \pm 6.94\,\%$ accuracy. Error bars indicate the standard deviation across 10 runs, showing that quantum circuits can achieve both higher accuracy and lower variance.

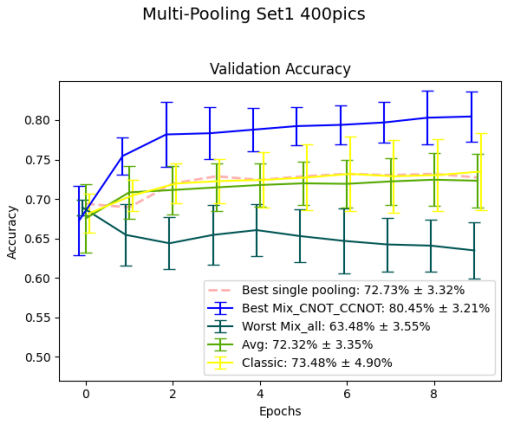


Figure 10: Validation accuracy for the best, worst, and average performing multi-pooling quantum circuits compared to a classical baseline, trained on Set1 using 400 images. The best multi-pooling circuit (Mix_CNOT_CCNOT) achieves a validation accuracy of $80.45\,\% \pm 3.21\,\%$, while the worst-performing circuit (Mix_all) reaches $63.48\,\% \pm 3.55\,\%$. The average performance across all multi-pooling models is $72.32\,\% \pm 3.35\,\%$. The classical baseline achieves $73.48\,\% \pm 4.90\,\%$. The dashed red line shows the best single-pooling circuit (CCCNOT), which reaches $72.73\,\% \pm 3.32\,\%$ accuracy and serves as a reference point for comparison. Error bars indicate the standard deviation across 10 runs, demonstrating variability and robustness in model performance.

Training the networks with 400 instead of 200 images (see Figure 10) leads to Mix_CNOT_CCNOT achieving the best performance with an accuracy of 80.45%, outperforming Mix_CNOT_CCCNOT from the 200-image experiment. In contrast, the average circuit (72.32%) and the worst-performing circuit Mix_all (63.48%) show slightly lower accuracies compared to their 200-image counterparts. Moreover, the mixture of all three single-pooling methods actively hinders feature extraction, as the validation accuracy decreases steadily over the course of ten training epochs, rendering this combination ineffective for this particular dataset.

Standard deviations are comparable to the results obtained with half the training data. The classical convolutional network increases its validation accuracy to 73.48%, which places it marginally above the average of the quanvolutional circuits, but well within the overlapping error margins.

4.2.2. Set2

For Set2 with 200 training images, stronger feature extraction capabilities are observed across all hybrid networks. Mix_CCNOT_CCCNOT achieves a final accuracy of 93.64%, slightly outperforming the best single-pooling technique CCCNOT_RX_s (91.52%). The average multi-pooling circuit reaches 85.19%, which is lower than the average of the single-pooling methods but still clearly surpasses the classical convolutional layer, which reaches 77.88%. The modular pooling method PoolMod_C achieves a similar mean accuracy of 78.48%, but is comparable to the classical counterpart ($\pm$21.77%); its standard deviation of $\pm$17.43% is large, indicating low stability.

Moreover, it should be noted that the modular poolings considerably increase the standard deviation of the average hybrid network to 6.9%, whereas all mixed pooling methods individually maintain error margins below 4%. A similar trend is observed in the loss dynamics, where only PoolMod_A (0.172), PoolMod_B (0.165), and PoolMod_C (0.305) of the multi-pooling circuits exhibit standard deviations

larger than 0.1, while all other hybrid models are below.

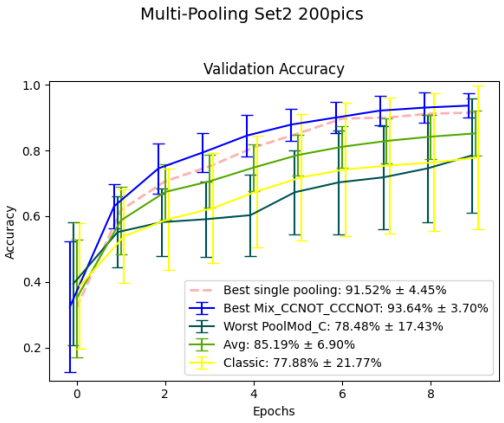


Figure 11: Validation accuracy for the best, worst, and average performing multi-pooling quantum circuits compared to a classical baseline, trained on Set2 using 200 images. The best-performing multi-pooling circuit (Mix_CNOT_CCNOT) achieves a validation accuracy of $93.64\,\% \pm 3.70\,\%$, while the worst-performing circuit (PoolMod_C) reaches $78.48\,\% \pm 17.43\,\%$. The average multi-pooling model reaches $85.19\,\% \pm 6.90\,\%$, and the classical baseline achieves $77.88\,\% \pm 21.77\,\%$. As a reference, the best single-pooling circuit (CCCNOT_RX_s) achieves $91.52\,\% \pm 4.45\,\%$, indicated by the dashed red line. Error bars indicate the standard deviation across 10 runs, reflecting robustness and variability in model performance.

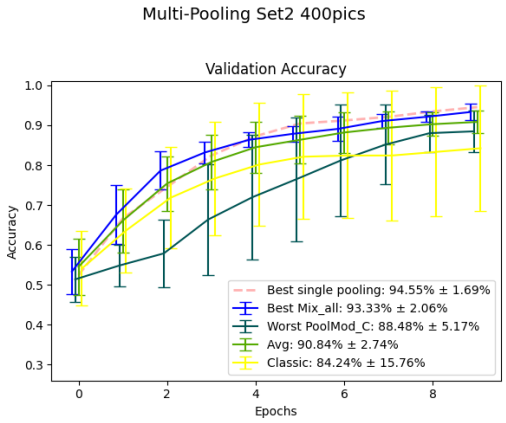


Figure 12: Validation accuracy for the best, worst, and average performing multi-pooling quantum circuits compared to a classical baseline, trained on Set2 using 400 images. The best-performing multi-pooling circuit (Mix_all) achieves a validation accuracy of $93.33\,\% \pm 2.06\,\%$, while the worst-performing circuit (PoolMod_C) reaches $88.48\,\% \pm 5.17\,\%$. The average model reaches $90.84\,\% \pm 2.74\,\%$, and the classical baseline achieves $84.24\,\% \pm 15.76\,\%$. As a reference, the best single-pooling circuit (CNOT_RX_s) achieves $94.55\,\% \pm 1.69\,\%$, shown as a dashed red line. Error bars indicate the standard deviation across 10 runs, reflecting robustness and variability in model performance.

Increasing the number of training images for Set2 from 200 to 400 raises the average accuracy of the multi-pooling circuits to 90.84%.Also, the weakest performing multi-pooling model for this dataset, PoolMod_C, improves by about 10% with the larger training set, reaching a final accuracy of 88.48% and outperforming the classical convolutional model, which achieves 84.24%.

The best-performing model, Mix_all, reaches a final accuracy of 93.33%, but this does not improve upon the Mix_CCNOT_CCCNOT circuit trained on only 200 images. It also performs slightly worse than the best single-pooling circuit for this dataset, CNOT_RX_s, which achieved 94.55%.

Overall, the multi-pooling circuits perform slightly worse than the single-pooling models for this dataset, as the average mean accuracy of multi-pooling circuits (90.84%) is marginally below that of the single-pooling circuits (91.45%).

4.2.3. Set3

Testing the multi-pooling strategies on Set3 with 10,000 training images, all quanvolutional networks approach performance saturation relatively quickly. Mix_all achieves the highest accuracy at 79.72% and the lowest validation loss, outperforming the best single-pooling model. The average accuracy across all multi-pooling circuits increased by approximately 4% compared to the single-pooling average of 74.81%, and even the weakest multi-pooling model, PoolMod_A, improved by about 2% to 61.94%.

Interestingly, the standard deviation decreased only for the Mix_all circuit, while both the average and the worst-performing circuits showed larger error margins than their single-pooling counterparts. Nevertheless, all quanvolutional models remained significantly more stable during training than the classical convolutional network, which achieved a mean accuracy of 73.46% but exhibited a large standard deviation of 31.09%, making its classification performance highly unreliable.

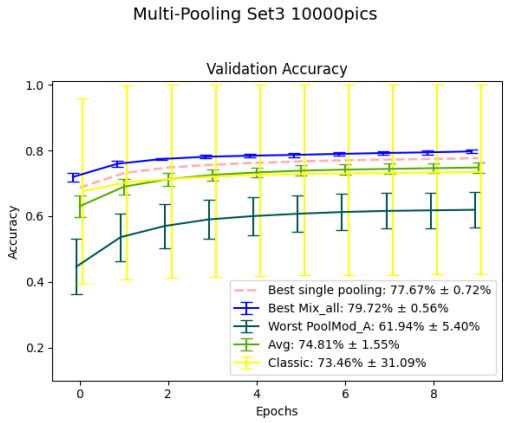


Figure 13: Validation accuracy for the best, worst, and average performing multi-pooling quantum circuits compared to a classical baseline, trained on Set3 using 10.000 images. The best-performing multi-pooling circuit (Mix_all) reaches a validation accuracy of $79.72\,\% \pm 0.56\,\%$, while the worst-performing circuit (PoolMod_A) achieves $61.94\,\% \pm 5.40\,\%$. The average model performs at $74.81\,\% \pm 1.55\,\%$, and the classical baseline reaches $73.46\,\% \pm 31.09\,\%$. As a reference, the best single-pooling circuit (CNOT_RX_s) achieves $77.67\,\% \pm 0.72\,\%$, indicated by the dashed red line. Error bars indicate the standard deviation across 10 runs, reflecting robustness and variability in model performance.

4.3. Comparison

The results of the single-pooling circuits reveal several trends across datasets. For Set2, the validation accuracies range from 75% to 92% for 200 training images and from 83% to 95% for 400 images, showing a clear performance increase with more data. However, not all circuits improve their feature extraction capabilities equally, as the best- and worst-performing quanvolutional models differ when doubling the amount of training data. Notably, in both cases, the classical convolutional baseline exhibits significantly larger standard deviations than any of the quanvolutional circuits, often more than twice as large. This suggests that the classical network is more sensitive to its initial parameters, whereas the quantum-classical hybrids train more reliably and consistently.

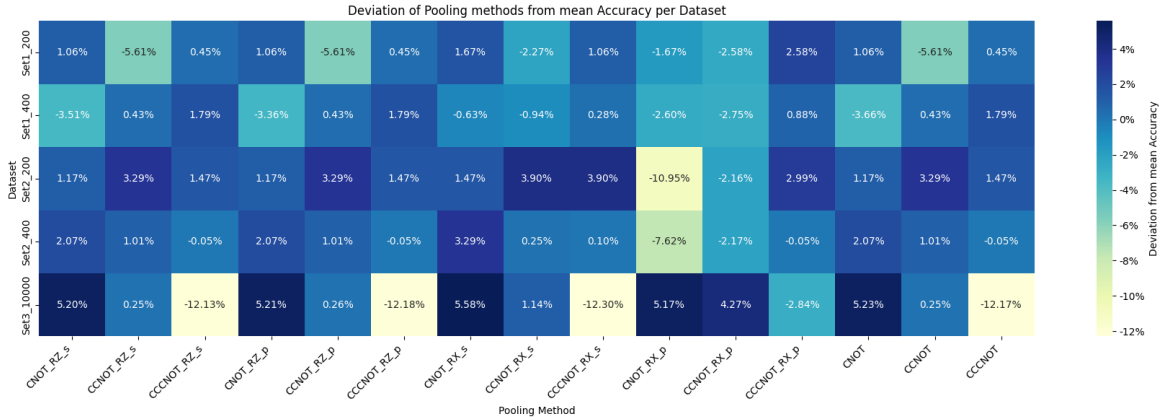


Figure 14: The heatmap presents the percentage point deviation of each single-pooling circuit's accuracy from the mean accuracy computed across all circuits for a given dataset (per row). For each dataset, the accuracies of all pooling methods were averaged, and the individual deviations are shown in color. Blue tones indicate above-average performance, while green to yellow shades mark below-average results. Clear patterns emerge within entanglement types (e.g., CNOT, CCNOT, CCCNOT), while the RX-redundant circuits (e.g., CNOT_RX_p, CCNOT_RX_p, CCCNOT_RX_p) exhibit strong, dataset-dependent deviations. This suggests inconsistent feature encoding caused by redundant parametrization.

This difference in stability becomes even more evident when comparing across datasets. Set1, which

contains relatively dark, low-contrast images, is not well represented by either model type. Nevertheless, quanvolutional networks achieve comparable or better accuracy with lower loss and smaller error margins. In contrast, Set2 and Set3 contain more structured and high-contrast features, which appear to favor the quantum circuits. Especially in Set3, with its 10-class classification task and 10,000 training images, the quanvolutional models not only outperform the classical baseline in accuracy, but also show remarkable robustness, with standard deviations remaining below 1% for all circuits.

Taken together, these results indicate that quanvolutional circuits are particularly effective on datasets with structured and high-contrast features. They generalize better, are less sensitive to initialization, and exhibit more stable training behavior. However, no single circuit consistently outperforms the others across all datasets, suggesting that the optimal choice may depend on the specific data characteristics. In contrast, while the classical convolutional layer can be competitive on average, it exhibits high variance and requires favorable initial conditions to perform well.

Interestingly, as shown in Figure 14, circuits sharing the same entanglement structure exhibit comparable performance, with only minor variations introduced by different parameterizations or angle embeddings. These patterns are visible across datasets, such as for Set1 with 200 training images, where all CCNOT-based circuits consistently perform below average, regardless of the specific embedding or parameterization. This trend becomes even more apparent in Set3, where all CNOT-based circuits perform similarly well, while all CCCNOT-based circuits perform significantly worse.

A notable exception to these entanglement-driven patterns arises with the circuits CNOT_RX_p, CCNOT_RX_p, and CCCNOT_RX_p. Their unusual behavior is attributed to RX-redundancy, where RX embedding is immediately followed by RX parameterization. This redundancy can distort the feature encoding by reinforcing or counteracting the input embedding. In some cases, such as CCNOT_RX_p and CCCNOT_RX_p in Set3, this effect can enhance performance. However, it may also hinder training, as seen with CNOT_RX_p in Set2, where the learning process becomes decoupled from the relevant input features.

While RX-based serial parameterization tends to slightly outperform RZ-based or non-parameterized designs, our results suggest that the choice of pooling operator, CNOT, CCNOT, or CCCNOT, has a more pronounced effect on classification performance. Therefore, in subsequent experiments on multi-pooling architectures, we focus on combinations of plain single-pooling blocks and omit parameterized variants to conserve computational resources without compromising model quality.

When testing more advanced multi-pooling strategies, the dependence on both the specific circuit design and the dataset characteristics becomes more pronounced. In general, beneficial combinations of pooling operations lead to improved feature representation, resulting in higher validation accuracies and lower standard deviations compared to single-pooling circuits as for example Mix_all in Set3. However, some combinations can also reduce classification performance and increase error margins, as observed in Figure 15 for Set1.

The modular pooling circuits (PoolMod_A, PoolMod_B, and PoolMod_C) perform comparably to single-pooling models but are clearly outperformed by the more effective mixed-pooling configurations. In particular, for the more structured datasets Set2 and Set3, the modular poolings exhibit the weakest feature representations, reducing the overall average accuracy of quanvolutional models and increasing their variance. These results suggest that the more complex entanglement and parameterization schemes in the modular designs offer no clear advantage for the classification tasks considered in this study.

Figure 15 presents the deviation of final validation accuracy from the dataset-specific mean for each multi-pooling circuit, including the plain single-pooling methods. In contrast to the patterns observed in the single-pooling setup, no clear or consistent relationship emerges between the mixture of entanglement strategies and model performance. For example, while CCCNOT performs best among single-pooling circuits for Set2 with 400 images, the combination of CNOT and CCNOT (Mix_CNOT_CCNOT) yields higher accuracy than any mixture involving CCCNOT. Furthermore, appending CCCNOT to an already well-performing Mix_CNOT_CCNOT (yielding Mix_all) can disrupt the circuit's capacity to extract relevant features and lead to decreased accuracy.

Overall, the results demonstrate that the benefits of mixing single-pooling methods are not easily predictable. While some combinations result in enhanced accuracy and stability, e.g., Mix_CNOT_CCNOT

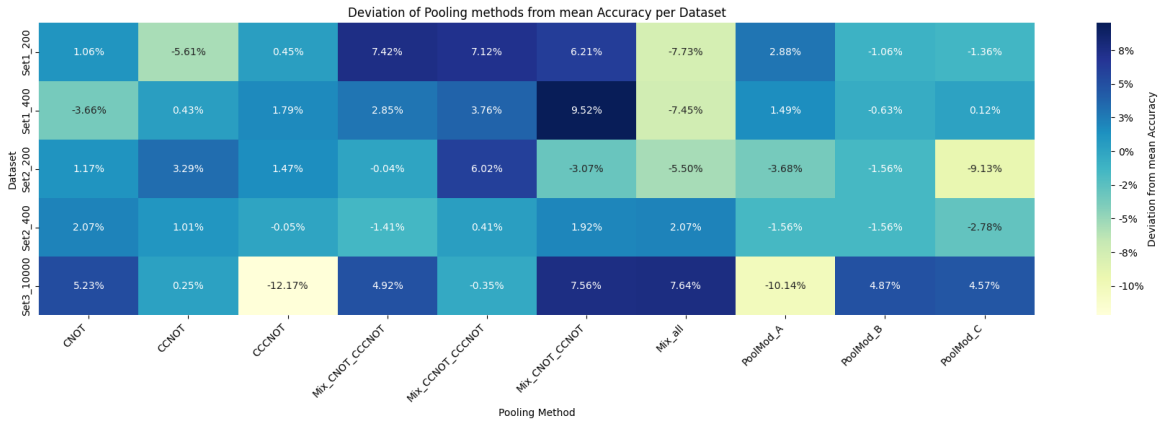


Figure 15: For each dataset (each row), the mean accuracy across the three non-variational single-pooling and all multi-pooling methods was computed. The heatmap displays the deviation (in percentage points) of each circuit's final accuracy from this mean. Blue tones indicate above-average performance, while green to yellow tones represent below-average results. This allows direct visual comparison of each pooling strategy's effectiveness per dataset. The Mix_CNOT_CCNOT circuit consistently performs well across datasets, while PoolMod_A and CCCNOT poolings show larger negative deviations, especially for more complex datasets like Set3.

in Set3, others unexpectedly degrade performance. Consequently, no generalizable criteria can be derived from the data to determine which pooling combinations will be beneficial for a given dataset. In this sense, both the optimal single-pooling strategy and any synergistic effects of their combinations remain dataset-specific and largely empirical.

A pattern does emerge, however, regarding dataset characteristics. Mix_all circuits perform better on datasets with more distinctive or high-contrast features (Set2 and Set3) and with larger training sets. This could suggest that combining multiple pooling types allows for a more comprehensive feature representation, which is particularly advantageous when rich information is available. However, this benefit does not generalize across all tested scenarios, and the combinatorial complexity of possible poolings makes systematic exploration challenging.

These findings also reinforce a broader insight observed throughout our experiments: The entanglement structure of the circuit has a much larger influence on performance than the parameterization method. Our empirical results show that circuits with different parameterizations but identical entanglement patterns tend to perform similarly, whereas changing the entanglement often leads to significant accuracy differences. One possible explanation is that entanglement introduces correlations between qubits, enabling the encoding of higher-order interactions among input features. This greatly enhances the circuit's expressive power, particularly in image-based tasks where spatial correlations are crucial.

In contrast, single-qubit parameterized gates act only locally, modifying individual amplitudes without introducing new correlations. As a result, parameter optimization primarily refines an already existing structure rather than fundamentally enhancing the circuit's capacity.

This observation aligns with the findings of Schuld et al. [7], who demonstrate that data encoding, especially under angle embedding, determines the function class a variational quantum circuit can represent. Since entanglement mediates how input data is distributed across the quantum register, it governs the circuit's ability to capture spatial or contextual relationships within the data. Parameterization, while still valuable, appears secondary in importance relative to entanglement when it comes to learning effective quantum representations. Consistently, Mahmud et al. [24] report that introducing three-qubit Toffoli-based interaction layers improves feature extraction and classification accuracy, which resonates with our observation that pooling circuits employing CCNOT and CCCNOT gates often provide enhanced representational power in multi-pooling schemes.

5. Conclusion & Outlook

The results indicate that single-layer quanvolutional filters in quantum-classical hybrid networks consistently outperform single-layer convolutional filters across all tested datasets. However, it remains uncertain whether this advantage extends to multi-layer quanvolutional networks or if it diminishes with increased complexity. The results also show that the trained rotation parameters used in the quantum circuit do not significantly affect the classification accuracy, regardless of their position in the pooling. Parameter-free pooling methods yielded results similar to both parallel and serial parametrized pooling techniques. However, since all the circuits utilized angle encoding, it is unclear whether parameters would be beneficial in circuits with alternative embedding strategies. Further research is needed to explore whether other encodings could also benefit from these pooling circuits or whether different pooling strategies would be more suitable.

It is important to note that there is no single circuit that consistently achieves the best performance. Rather, the entanglement method must be chosen based on the specific features present in the data. The representation of these features by the networks varies not only by the dataset but also by the number of training images. Increasing the training data induces shifts in the feature representation due to subtle differences in the data. Therefore, different datasets favor different pooling circuits for optimal feature representation, and even small changes in the data can affect the choice of the best circuit. Despite this, when the features change slightly, such as by increasing the training data, the best-performing circuit may switch, but the previously optimal circuit will still perform comparably well. To gain a better understanding of feature-specific entanglements and the advantages of quanvolutional networks for different types of data, further investigations into various pooling methods across diverse datasets are needed.

Finally, mixing simple pooling methods can generally improve the performance of quanvolutional layers, but no clear pattern emerges for which pooling methods should be combined for optimal feature representation. As with single pooling methods, no mixed circuit consistently outperforms others; the best choice depends on the features of the dataset. Comparing the mixed circuits with the three modular poolings reveals that simply increasing entanglement strength does not improve performance. Instead, feature representation is enhanced by combining individual input information (via CNOT) with the relationships between inputs, as achieved by CCNOT and CCCNOT entanglements. Our findings suggest that combining simple pooling methods can enhance the representational power of quanvolutional layers for structured data, yet it remains an open question whether the order in which these methods are applied further influences performance, presenting a potential area for future investigation.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-4o and Grammarly in order to: Grammar and spelling check. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] M. Schuld, I. Sinayskiy, F. Petruccione, An introduction to quantum machine learning, Contemporary Physics 56 (2015) 172–185.

[2] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, Quantum machine learning, Nature 549 (2017) 195–202.

[3] K. O'Shea, An introduction to convolutional neural networks, arXiv preprint arXiv:1511.08458 (2015).

[4] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A.

Fadhel, M. Al-Amidie, L. Farhan, Review of deep learning: concepts, cnn architectures, challenges, applications, future directions, Journal of big Data 8 (2021) 1–74.
[5] M. Henderson, S. Shakya, S. Pradhan, T. Cook, Quanvolutional neural networks: powering image recognition with quantum circuits, Quantum Machine Intelligence 2 (2020) 2.
[6] J. Liu, K. H. Lim, K. L. Wood, W. Huang, C. Guo, H.-L. Huang, Hybrid quantum-classical convolutional neural networks, Science China Physics, Mechanics & Astronomy 64 (2021) 290311.
[7] M. Schuld, R. Sweke, J. J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Physical Review A 103 (2021) 032430.
[8] M. Monnet, H. Gebran, A. Matic-Flierl, F. Kiwit, B. Schachtner, A. Bentellis, J. M. Lorenz, Pooling techniques in hybrid quantum-classical convolutional neural networks, in: 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), volume 1, IEEE, 2023, pp. 601–610.
[9] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (2015) 436–444.
[10] I. Goodfellow, Deep learning, 2016.
[11] S. Sharma, S. Sharma, A. Athaiya, Activation functions in neural networks, Towards Data Sci 6 (2017) 310–316.
[12] M. Rath, H. Date, Quantum data encoding: a comparative analysis of classical-to-quantum mapping techniques and their impact on machine learning accuracy, EPJ Quantum Technology 11 (2024) 72.
[13] A. Matic, M. Monnet, J. M. Lorenz, B. Schachtner, T. Messerer, Quantum-classical convolutional neural networks in radiological image classification, in: 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), IEEE, 2022, pp. 56–66.
[14] J. Bowles, S. Ahmed, M. Schuld, Better than classical? the subtle art of benchmarking quantum machine learning models, arXiv preprint arXiv:2403.07059 (2024).
[15] I. Kerenidis, J. Landman, A. Prakash, Quantum algorithms for deep convolutional neural networks, arXiv preprint arXiv:1911.01117 (2019).
[16] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, P. J. Coles, Absence of barren plateaus in quantum convolutional neural networks, Physical Review X 11 (2021). URL: http://dx.doi.org/10.1103/PhysRevX.11.041011. doi:10.1103/physrevx.11.041011.
[17] H. Gholamalinezhad, H. Khosravi, Pooling methods in deep neural networks, a review, arXiv preprint arXiv:2009.07485 (2020).
[18] I. Cong, S. Choi, M. D. Lukin, Quantum convolutional neural networks, Nature Physics 15 (2019) 1273–1278.
[19] Y. Song, J. Li, Y. Wu, S. Qin, Q. Wen, F. Gao, A resource-efficient quantum convolutional neural network, Frontiers in Physics 12 (2024) 1362690.
[20] K. Chinzei, Q. H. Tran, K. Maruyama, H. Oshima, S. Sato, Splitting and parallelizing of quantum convolutional neural networks for learning translationally symmetric data, Physical Review Research 6 (2024) 023042.
[21] T. Hur, L. Kim, D. K. Park, Quantum convolutional neural network for classical data classification, Quantum Machine Intelligence 4 (2022). URL: http://dx.doi.org/10.1007/s42484-021-00061-x. doi:10.1007/s42484-021-00061-x.
[22] C. Loglisci, D. Malerba, S. Pascazio, Quarta: quantum supervised and unsupervised learning for binary classification in domain-incremental learning, Quantum Machine Intelligence 6 (2024) 68.
[23] S. Das, S. Martina, F. Caruso, The role of data embedding in equivariant quantum convolutional neural networks, Quantum Machine Intelligence 6 (2024) 82.
[24] J. Mahmud, R. Mashtura, S. A. Fattah, M. Saquib, Quantum convolutional neural networks with interaction layers for classification of classical data, Quantum Machine Intelligence 6 (2024) 11.
[25] W. Al-Dhabyani, M. Gomaa, H. Khaled, A. Fahmy, Dataset of breast ultrasound images, Data in Brief 28 (2020) 104863.
[26] J. Yang, R. Shi, B. Ni, Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis, in: IEEE 18th International Symposium on Biomedical Imaging (ISBI), 2021, pp. 191–195.
[27] L. Deng, The mnist database of handwritten digit images for machine learning research [best of the web], IEEE signal processing magazine 29 (2012) 141–142.

[28] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. AkashNarayanan, A. Asadi, et al., Pennylane: Automatic differentiation of hybrid quantum-classical computations, arXiv preprint arXiv:1811.04968 (2018).

[29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in pytorch, in: NIPS 2017 Workshop on Autodiff, 2017. URL: https://openreview.net/forum?id=BJJsrmfCZ.

[30] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017. URL: https://arxiv.org/abs/1412.6980. arXiv:1412.6980.

A. Investigation batch sizes master's thesis

An ablation study on batch size as a hyperparameter was conducted prior to this work as part of a master's thesis. The results showed only minor variations in validation accuracy for batch sizes ranging from 8 to 64, with no consistent preference for a specific value. Consequently, batch size was fixed to 16 in the experiments presented in this paper to reduce computational complexity and simplify hyperparameter tuning.

Figures 16, 17, and 18 illustrate the final validation accuracy after 20 training epochs for Set1 with 546 training images, using CNOT, CCNOT, and CCCNOT pooling, respectively. Additionally, corresponding results for Set3 with 60,000 training images are shown in Figures 19, 20, and 21.

Each plot compares three differently parameterized circuits, basent_RX, strent_rot, and noent_RX, as well as a classical convolutional baseline. Of particular interest is the noent_RX circuit, which corresponds exactly to the unparameterized single-pooling CNOT, CCNOT, and CCCNOT circuits used throughout this work. Across all configurations, the error bars overlap substantially, indicating that batch size does not significantly influence classification performance in the tested setups.

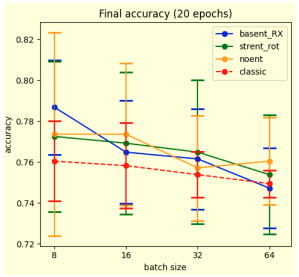


Figure 16: The final validation accuracy of the CNOT pooling circuits shows no meaningful variation across different batch sizes (8–64). While there are minor fluctuations in the mean accuracy, the overlapping error bars indicate that batch size has no significant influence on the classification performance for this setup.

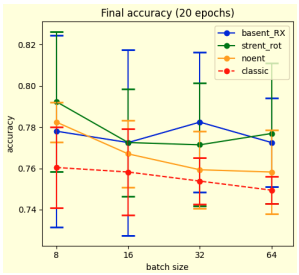


Figure 17: For the CCNOT pooling method, the networks achieve comparable final accuracies across all tested batch sizes. The variations lie well within the standard deviations, suggesting that batch size does not meaningfully affect the training dynamics or generalization performance on Set1.

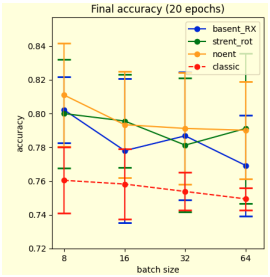


Figure 18: With CCCNOT pooling, the network exhibits slightly different trends across batch sizes, but the broad and overlapping confidence intervals suggest these differences are not statistically significant. Overall, batch size has a negligible effect on final accuracy in this configuration.

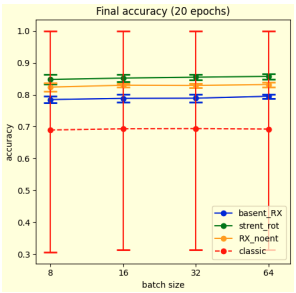


Figure 19: The final validation accuracy of the CNOT pooling circuits on Set3 (60,000 images) shows consistent results across different batch sizes (8–64). Minor shifts in the mean accuracy are visible, but all results lie within overlapping confidence intervals. Batch size has no significant impact on classification performance in this configuration.

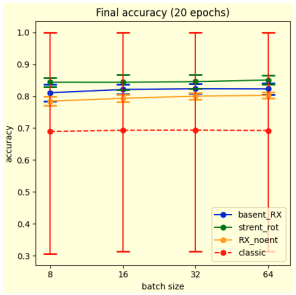


Figure 20: Across all batch sizes tested, the CCNOT pooling circuits yield similar final accuracies on Set3. Despite small absolute differences, the standard deviations largely overlap, indicating that batch size has no meaningful influence on training outcome or generalization in this setup.

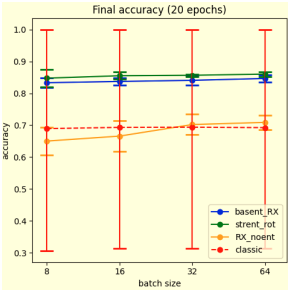


Figure 21: The CCCNOT pooling circuits reach stable classification accuracy regardless of batch size on Set3. The large error bars seen in the classical baseline do not affect the quantum-enhanced models, which perform robustly. No clear advantage of any specific batch size is observed.