SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

The impact of pooling paradigms on quantum convolutional neural networks

Hanady Gebran

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY — INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

The impact of pooling paradigms on quantum convolutional neural networks

Die Auswirkungen von Pooling-Paradigmen auf Quanten Faltungsneuronale Netze

Author:	Hanady Gebran
Supervisor:	Prof. Dr Christian Mendl
Advisor:	PD Dr. habil. Jeanette Miriam Lorenz and Maureen Monnet
Submission Date:	15.06.2023

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Munich, 15.06.2023 Hanady Gebran

Abstract

Quantum machine learning is an emerging field that combines quantum computing and machine learning to harness the power of quantum phenomena, such as superposition, entanglement, and interference, to achieve speedups or improvements over classical computing for certain computational problems.

Efforts to date show that for many of the proposed quantum models, we have promising results in their specific domains, with theoretical studies showing that quantum variants of classical machine learning algorithms can provide good generalization from small training data.

However, there are no strong theoretical ideas about what makes one quantum circuit design better than another, and comparative studies between quantum equivalents have not been performed for all types of classical layers or techniques crucial to classical machine learning. In particular, the pooling layer within convolutional neural networks is a fundamental operation that remains to be explored. Pooling mechanisms significantly improve the performance of classical machine learning algorithms by playing a key role in reducing input dimensionality and extracting eigenfeatures from the input data.

This thesis proposes hybrid quantum-classical convolutional neural networks (QCCNNs) with parameters comparable to those of classical convolutional neural networks and examines the potential application of quantum machine learning in medical imaging classification tasks. These tasks often have limited training data, making it difficult to reliably classify specific diseases such as potentially cancerous lesions.

An in-depth study of pooling techniques in QCCNNs for 2D medical image classification is performed. The performance of three different quantum and hybrid pooling techniques are investigated: mid-circuit measurements, ancillary qubits with controlled gates, and qubit selection with classical post-processing.

By simulating this network with a classical computer using the pennylane framework, we find similar or better performance compared to an equivalent classical model and a QCCNN without pooling and conclude that it is promising to study architectural choices in QCCNNs further for future applications.

Zusammenfassung

Das maschinelle Quantenlernen ist ein aufstrebendes Gebiet, das Quanteninformatik und maschinelles Lernen kombiniert, um die Stärke von Quantenphänomenen wie Superposition, Verschränkung und Interferenz zu nutzen, um bei bestimmten Rechenproblemen Beschleunigungen oder Verbesserungen im Vergleich zur klassischen Informatik zu erzielen.

Die bisherigen Bemühungen zeigen, dass wir für viele der vorgeschlagenen Quantenmodelle vielversprechende Ergebnisse in ihren spezifischen Bereichen haben, wobei theoretische Studien zeigen, dass Quantenvarianten von klassischen Algorithmen des maschinellen Lernens gute Verallgemeinerungen aus kleinen Lerndaten liefern können.

Es gibt jedoch keine starken theoretischen Ideen darüber, was ein Quantenschaltungsdesign besser als ein anderes macht, und vergleichende Studien zwischen Quantenäquivalenten wurden nicht für alle Arten von klassischen Schichten oder Techniken durchgeführt, die für das klassische maschinelle Lernen entscheidend sind. Insbesondere die Pooling-Schicht innerhalb von gefalteten neuronalen Netzen ist eine grundlegende Operation, die noch erforscht werden muss. Pooling-Mechanismen verbessern die Leistung klassischer Algorithmen des maschinellen Lernens erheblich, da sie eine Schlüsselrolle bei der Verringerung der Dimensionalität der Eingabe und der Extraktion eigener Merkmale aus den Eingabedaten spielen.

Diese Dissertation schlägt Quantum-Classic Hybrid Convolutional Neural Networks (QCCNN) mit Parametern vor, die mit denen klassischer Convolutional Neural Networks vergleichbar sind, und untersucht die potenzielle Anwendung von Quantum Machine Learning bei Klassifikationsaufgaben in der medizinischen Bildgebung. Diese Aufgaben verfügen oft über begrenzte Lerndaten, was die zuverlässige Klassifizierung spezifischer Krankheiten, wie z. B. potenziell krebsartiger Läsionen, erschwert.

Es wird eine eingehende Untersuchung von Pooling-Techniken in QCCNN zur Klassifizierung von medizinischen 2D-Bildern durchgeführt. Die Leistung von drei verschiedenen Quanten- und Hybrid-Pooling-Techniken wird untersucht: Messungen in der Mitte des Schaltkreises, Ancillary-Qubits mit kontrollierten Gattern und Qubit-Auswahl mit herkömmlicher Nachbearbeitung.

Bei der Simulation dieses Netzwerks mit einem klassischen Computer unter Verwendung des Pennylane-Rahmens finden wir eine ähnliche oder bessere Leistung im Vergleich zu einem äquivalenten klassischen Modell und einem QCCNN ohne Pooling

und kommen zu dem Schluss, dass es vielversprechend ist, die architektonischen Entscheidungen in QCCNNs für zukünftige Anwendungen genauer zu untersuchen.

Contents

1 Introduction

Machine learning is a form of artificial intelligence (AI) that aims to create systems that learn or improve their performance based on the data they process. Some of the areas where machine learning is becoming increasingly prevalent include science, healthcare (the subject of this thesis), education, manufacturing (e.g. damage detection), financial modelling (future profits), cybersecurity, data governance and even marketing. This can result in significant time savings and enable tasks that would otherwise require extensive specialisation. For example, doctors may need years of study to detect diseases, but machine learning can help them with this task. [Mit97]

One of the most notable and influential subfields of machine learning is deep learning, which uses multiple layers of artificial neural networks. This learning often requires massive databases of information to be trained. Deep learning has achieved promising results in image, video, speech and audio processing with deep convolutional neural networks, and in modelling sequential data such as text and speech by using recurrent networks [Bro+21]. With their impressive efficiency and performance, the field of research is very dynamic. [GBC16][LBH15]

Deep Convolutional Neural Networks (DCNNs) are one of the most common and widely used varieties of recurrent networks. In the past, feature engineering required human intervention, whereas today DCNNs have multiple layers of neurons that apply local filters to the input data and create feature maps that capture the relevant information. They therefore learn without human supervision or prior knowledge, which makes them effective for image recognition tasks [KSH12]. The hierarchical structure of the human visual system inspired the idea of DCNNs and, since their introduction in the 1980s, they have achieved peak performance in various computer vision tasks, such as object detection, segmentation, tracking, face recognition, etc.

Quantum computing is an emerging technology that uses quantum bits (qubits) and can hence exploit the advantages of quantum phenomena, such as entanglement, superposition, and interference, to achieve speed-ups or improvements over classical computing for certain computational problems. For example, quantum computing can help solve cryptographic problems that are difficult or impossible to solve with classical computing. Factorization problems, in particular, are excellent examples of problems that can be solved with a quantum computer [DR22]. Although some reservations are expressed as to whether the idea of "beating" classical machine learning with quantum

gain should continue to dominate the literature [SK22b], the hope is that QC can improve AI systems, which would ultimately combine two of the most currently active topics [NC00].

Indeed, by using quantum parallelism or interference effects, QC could process smaller datasets and discover new patterns in the data that otherwise might be hidden or inaccessible to conventional algorithms, which is important given the frequent lack of labeled data for supervised learning. It may also achieve better results with fewer learning steps as good generalization is guaranteed from few training data [Car+22a].

In theory, QC has already shown its potential for several use cases. However, current noisy intermediate-scale quantum computers (NISQs) have a limited number of $\mathcal{O}(100)$ qubits, with limited connectivity and gate fidelities. As a result, some of the most interesting algorithms developed are still far from being accessible. For example, Shor's algorithm can efficiently factor large numbers on a quantum computer, breaking some widely used public key cryptosystems due to the hardness of the factoring problem. However, breaking the current 2048-bit RSA key requires a circuit of about 4000 logical qubits, knowing that with error correction, 60 physical qubits are equivalent to about 1 logical qubit, for this use case we would need a NISQ computer of about $\mathcal{O}(100000)$ qubits. Therefore, in the near term, hybrid QML algorithms, which use a limited number of qubits, are one of the most promising use cases for QC. These algorithms involve iterative interactions between classical and quantum computers, taking advantage of their respective strengths.

In Section 2, the basics of machine learning and convolutional neural networks are reintroduced. In Section 3, quantum computing and some of its applications to machine learning are discussed. In Section 4, the focus is on quantum-classical convolutional neural networks (QCCNN) and the gaps in the literature that this thesis aims to fill. In Section 5, an experimental setup using QCCNN with comparable parameters is presented and the approaches using 1) mid-circuit measurements, 2) auxiliary qubits with controlled gate operations, and lastly 3) qubit selection with a non-linear classical postprocessing function are explained.In Section 6, the performance of all tested quantum data pooling architectures is presented and interpreted, and the correlation between the effective dimension and learning performance is investigated. In Section 7, the thesis is concluded and future work in the field of quantum machine learning is discussed.

2 Introduction to deep learning in radiological imaging

2.1 Machine learning and deep learning

Deep learning is a subset of machine learning that uses artificial neural networks (ANNs) inspired by the structure of the brain itself. ANNs consist of interconnected processing nodes that receive inputs from other neural nodes and calculate a weighted sum of the inputs. This sum is often passed through an activation function to determine the response of the neural node. The connection weights between neurons are adjusted by backpropagation to minimize the error between the expected and actual results [Bro+21] [GBC16] [JZH21].

A single layer Artificial Neural Network (ANN) has a limit on the number of input representations it can handle. This is why deep neural networks (DNNs) were introduced. DNNs are composed of several layers, and while each individual layer does not have much complexity or discriminating power, the combination of several layers allows a high level of pattern recognition to be achieved. In fact, in this type of architecture, each layer corresponds to a different level of abstraction: the lower levels learn simple features such as lines or circles, and the higher levels learn more complex features such as noses, hair or lips. The advantage of this approach is that engineers do not need to hard-code the relevant features to accomplish the task at hand, as it is the different levels of abstraction that generate the correct discriminating feature sensors to predict the class of an image. As manual feature engineering is minimal here, the use of such an architecture has quickly become popular and widespread. However, training DNNs usually presents some difficulties such as over-fitting, gradient explosion, class imbalance and the need for large datasets.

While DNNs have achieved state-of-the-art performance in many applications, they have some limitations. One of the main ones is their poor ability to generalize and extrapolate beyond the training data. They also rely heavily on large amounts of labeled data, making them extremely data-dependent and unsuitable for applications where data availability is limited. These models also have difficulty operating in open environments, adapting to outliers, perceiving hierarchical structures, distinguishing between correlation and causation, and providing meaningful explanations for their

decisions. Finally, deep neural networks lack the capacity for ontological inference, an essential ability to understand abstract concepts [Tsi20].

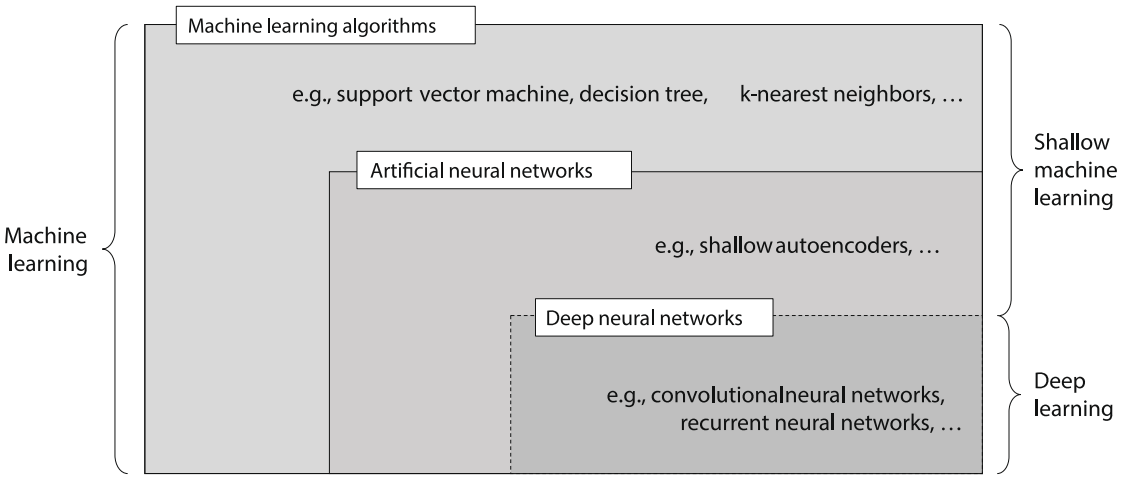


Figure 2.1: Machine learning= Shallow machine learning + Deep learning [JZH21]

2.2 Convolutional Neural Networks

2.2.1 High level overview

A CNN replicates the visual cortex's simple and complex cells, which respond to local edges, directions, and more complex patterns and shapes, to learn increasingly complex features from input images [AG17]. CNNs outperform other neural networks when given inputs such as images, speech, or audio, which sets them apart from other neural networks.

Convolutional, pooling, and fully-connected (FC) layers are the three primary types of layers used in CNNs. The convolutional layer is the first layer of a convolutional network and the fundamental building block, where the majority of computation takes place. A pixel matrix makes up the input data. The feature detector, also known as a kernel or filter, moves across the image's receptive fields and looks to see if the feature is present. A feature hierarchy is produced in the CNN by the fact that the later layers can see the pixels in the earlier layers' receptive fields.

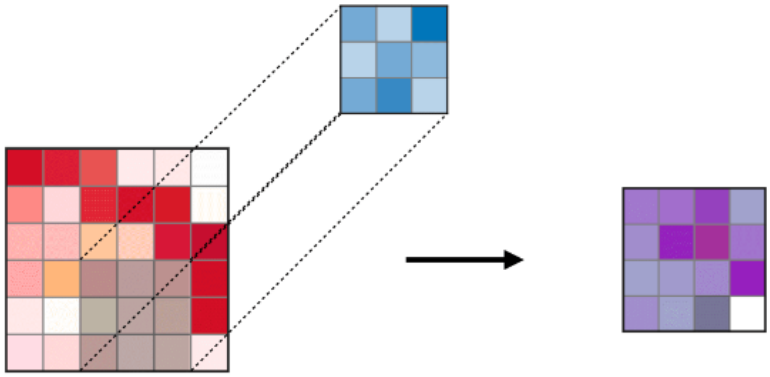


Figure 2.2: Visual representation of a convolutional layer [Sta23].

The pooling layer is a subsampling operation typically applied after a convolutional layer. In particular, the most popular types of pooling are max and average pooling, where the maximum and average values are taken, respectively.

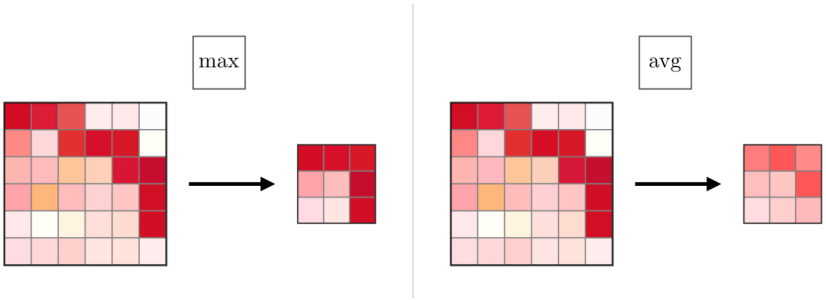


Figure 2.3: Visual representation of a max pooling layer and an avg pooling layer [Sta23].

A feature map, which is the convolution's output, is passed through the ReLU activation function to add nonlinearity. $ReLU : g(z) = \max(0, z)$
The final layer of a CNN is the fully-connected layer. Its name refers to the fact that every neuron in the layer is linked to every neuron in the layer above. The fully-connected layer, which maps the extracted features to the desired output, receives flattened and fed output from the final pooling layer.

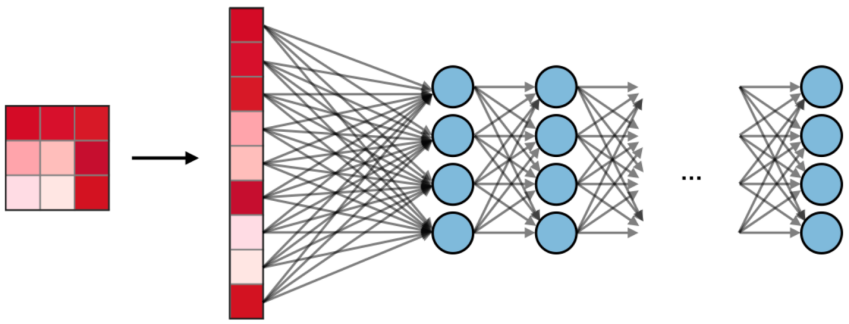


Figure 2.4: Visual representation of a fully connected layer [Sta23].

Recent developments in CNNs include the use of attention mechanisms to help the network focus on task-specific areas of input images and residue connections to learn residue features that capture the difference between an input and an output of a layer. Because they are able to learn hierarchical representations of input images, process high-dimensional inputs, and recognize complex features in images, CNNs are effective tools for computer vision tasks.

2.2.2 Architecture and training

A convolutional neural network (CNN) typically accepts as input a three-dimensional tensor, typically an image with dimensions H rows, W columns, and 3 channels (R, G, B). The input is then subjected to a sequential chain of calculations.

$$\boldsymbol{x}^1 \longrightarrow \boldsymbol{w}^1 \longrightarrow \boldsymbol{x}^2 \longrightarrow \boldsymbol{x}^{L-1} \longrightarrow \boldsymbol{w}^{L-1} \longrightarrow \boldsymbol{x}^L \longrightarrow \boldsymbol{w}^L \longrightarrow z$$

The input is processed in the initial layer. The tensor $\boldsymbol{w}^1$ represents the parameters involved in processing the first layer. The result of the first layer is denoted by $\boldsymbol{x}^2$, which simultaneously serves as the input for the processing of the next layer.

This sequential processing continues until all layers of the CNN have been traversed, resulting in the $\boldsymbol{x}^L$. Assuming that the given problem concerns image classification involving C classes, it is usual to present $\boldsymbol{x}^L$ as a C dimensional vector. The i-th entry of this vector encodes the posterior probability that $\boldsymbol{x}^1$ belongs to the i-th class. To transform $\boldsymbol{x}^L$ into a probability mass function, the treatment in the $(L-1)$th layer can be performed as a softmax transformation of $\boldsymbol{x}^{L-1}$.

The last layer is called the loss layer. Suppose that t represents the corresponding target value (ground truth) for the input $\boldsymbol{x}^1$. In this context, a cost or loss function can

be used to evaluate the dissimilarity between the CNN prediction $\boldsymbol{x}^L$ and the target value $\boldsymbol{t}$. For example, a simple loss function could be defined as follows:

$$z = \frac{1}{2} \left\| \boldsymbol{t} - \boldsymbol{x}^L \right\|^2$$

In the case of a classification problem, cross-entropy loss is often used. In such a scenario, the ground truth is represented by a categorical variable t. This categorical variable t is then transformed into a C dimensional vector $\boldsymbol{t}$. At this stage, $\boldsymbol{t}$ and $\boldsymbol{x}^L$ both take the form of probability mass functions, with the cross-entropy loss serving as a measure of dissimilarity between them. Therefore, minimising the cross-entropy loss allows the CNN to be formed efficiently.

The parameters of the CNN model are optimized to minimize the loss function z and match the desired labels. Learning consists of going through the network to obtain predictions and comparing them to the targets to calculate the loss. The parameters are updated gradient descent methods that adjusts the parameters based on the partial derivative of the loss after each parameter. By updating the parameters in the opposite direction of the gradient, the loss function is minimized. Suppose all the parameters of a CNN model $\boldsymbol{w}^1, \ldots, \boldsymbol{w}^{L-1}$ have been learned, we can output the CNN prediction as $argmax_i \boldsymbol{x}_i^L$ [Wu17].

2.2.3 Loss landscape and accuracy

The loss landscape of a deep neural network refers to the multidimensional structure of the loss function that minimizes the network during learning. The loss function, represented by $f(\theta)$, compares the network's expected output to the actual output and is generally non-convex, which means it has many minima and can be difficult to optimize [Li+18][ZLZ21]. Despite the complexity of the loss function, neural networks are often able to identify optimal minimizers that achieve high accuracy on training and test data. Current research has focused on understanding the shape of the loss landscape and its impact on the optimization process. The loss landscape can have various characteristics such as: flat regions, steep cliffs and narrow valleys, which can affect the optimizer's ability to find good solutions.

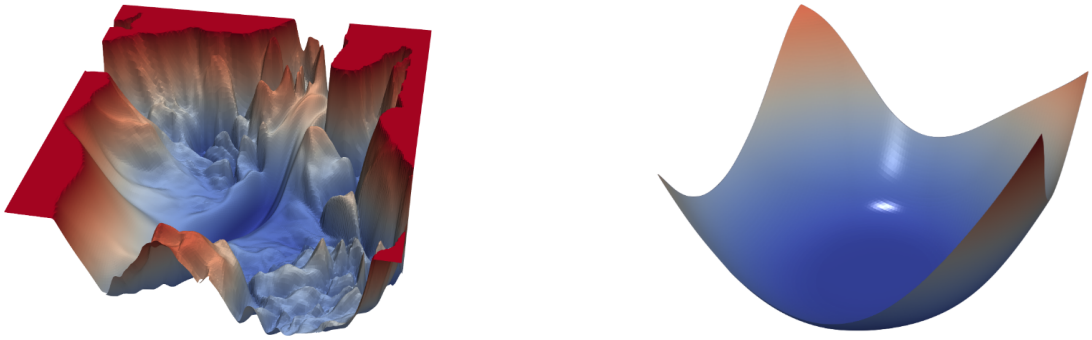


Figure 2.5: Comparative plot of loss landscape regions: chaotic vs smooth. [Tho]

2.2.4 Convolution layer

In the convolution layer, using multiple convolution kernels is standard. Assuming the use of D kernels, each having a spatial dimension of $H \times W$, the collective representation of these kernels can be denoted as $\boldsymbol{f}$. Represented as an order 4 tensor in $\mathbb{R}^{H \times W \times D^l \times D}$ and encompasses the attributes of the kernels.

As illustrated in Figure 2.2, the spatial dimensions of the output are smaller compared to those of the input, especially when the convolution kernel size exceeds 1×1. Given an input size of $H^l \times W^l \times D^l$ and a kernel size of $H \times W \times D^l \times D$, the resulting convolution yields a size of $\left(H^l - H + 1\right) \times \left(W^l - W + 1\right) \times D$.

Stride, an equally significant concept in convolution, dictates the movement of the convolution process. As depicted in Figure 2.2, the kernel convolves with the input at every possible spatial location, corresponding to a stride value of $s = 1$. However, when $s > 1$, the kernel skips $s - 1$ pixel locations with each movement, leading to the creation of $\boldsymbol{y}$ (or $\boldsymbol{x}^{l+1}$) in $\mathbb{R}^{H^{l+1} \times W^{l+1} \times D^{l+1}}$, where $H^{l+1} = H^l - H + 1$, $W^{l+1} = W^l - W + 1$, and $D^{l+1} = D$.

Hence, the convolution process can be expressed as the following equation:

$$y_{i^{l+1},j^{l+1},d} = \sum_{i=0}^{H} \sum_{j=0}^{W} \sum_{d^l=0}^{D^l} f_{i,j,d^l,d} \times x^l_{i^{l+1}+i,j^{l+1}+j,d^l}.$$

[Wu17].

2.2.5 Pooling layer

The pooling layer addresses concerns such as overfitting, computation time, and recognition accuracy. It performs a downsampling operation on the feature maps, extracting relevant information while discarding irrelevant details. There are different types of pooling methods commonly used in CNNs, including max pooling and average pooling.

In the pooling layer, the input $\boldsymbol{x}^l \in \mathbb{R}^{H^l \times W^l \times D^l}$ is processed channel by channel independently. The spatial extent of the pooling, specified by the dimensions $H \times W$, is determined during the design of the CNN structure. The most commonly used setup is $H = W = 2$ with a stride of 2. Assuming that H divides H^l and W divides W^l, and the stride equals the pooling spatial extent, the output $\boldsymbol{y}$ (or equivalently $\boldsymbol{x}^{l+1}$) of the pooling layer will be an order 3 tensor of size $H^{l+1} \times W^{l+1} \times D^{l+1}$, where

$$H^{l+1} = \frac{H^l}{H}, \quad W^{l+1} = \frac{W^l}{W}, \quad D^{l+1} = D^l$$

The pooling operation divides the $H^l \times W^l$ matrix within each channel into non-overlapping subregions of size $H \times W$. Each subregion is then mapped to a single number using a pooling operator [Wu17].

In max pooling, the pooling operator maps a subregion to its maximum value, while the average pooling maps a subregion to its average value. Precisely,

Max Pooling:

$$y_{i^{l+1},j^{l+1},d} = \max_{0 \le i < H, 0 \le j < W} x^l_{i^{l+1} \times H + i, j^{l+1} \times W + j, d}$$

Average Pooling:

$$y_{i^{l+1},j^{l+1},d} = \frac{1}{HW} \sum_{0 \le i < H, 0 \le j < W} x^l_{i^{l+1} \times H + i, j^{l+1} \times W + j, d}$$

Here, $0 \le i^{l+1} < H^{l+1}$, $0 \le j^{l+1} < W^{l+1}$, and $0 \le d < D^{l+1} = D^l$.

Max pooling selects the maximum value within each pooling subregion, providing translation invariance and reducing the dimension of intermediate feature maps. On the other hand, average pooling calculates the average value of each subregion, providing a downscaled representation of the input. Pooling can also be performed on overlapping regions, which can be beneficial in situations where weak spatial information surrounding dominant regions is useful [Anw+18].

The pooling layer's primary objectives are to reduce the number of parameters, thus minimizing computational overhead and avoiding overfitting. By extracting important information and discarding irrelevant details, pooling assists in transforming general feature descriptions into actionable information. The choice of pooling method is

crucial in solving computer vision challenges since it aims to capture meaningful representations of combined visual features obtained through convolution.

2.3 Application in radiological imaging

Convolutional neural networks revolutionized medical imaging by enabling fast and precise identification of abnormalities in various medical images [ZS19]. CNNs could surpass or match human radiologists in detecting lung nodules, breast cancer, and brain tumors, thereby increasing productivity and advancing medical imaging technologies.The scope of this thesis is on the breast cancer detection application.This topic is prominent as it is a global challenge, given that this type of cancer is common and affects 10% of women globally. Early detection and prompt treatment are critical in improving the chances of survival [Kly+14]. One of the most effective techniques for detecting breast cancer in its early stages is radiological imaging, such as mammography. Mammography can identify small calcifications and masses that may be indicative of cancer, making it a highly effective technique for breast cancer detection [SK22a].

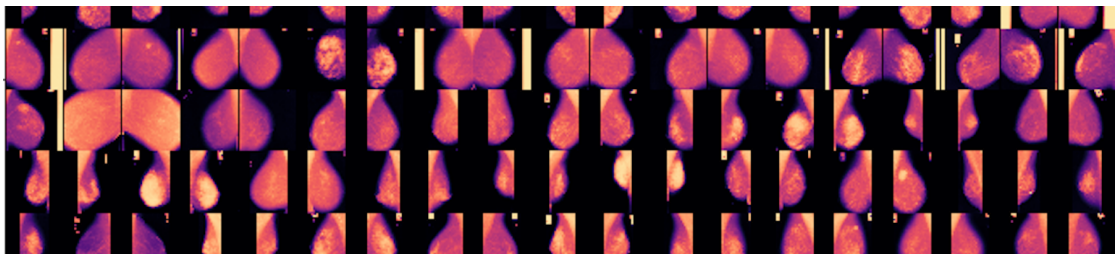


Figure 2.6: Mammography dataset [Ya21][Aa20].

In the domain of radiological imaging, the widespread adoption of convolutional neural networks faces numerous challenges that need to be addressed.

One significant obstacle is the interpretability of CNN predictions. Despite their impressive performance, CNNs often generate results that are difficult to comprehend and trust. This lack of transparency impedes their application in critical scenarios, as healthcare professionals such as clinicians and radiologists require explanations and insights to confidently rely on CNN outcomes [Pre+19]. To overcome this challenge and enhance the accuracy and interpretability of CNNs in radiological imaging, researchers are actively exploring innovative solutions. One such approach is "pruning by explanation," which draws inspiration from the field of explainable artificial intelligence (XAI). This method aims to identify the most relevant components within a CNN by assigning relevance scores based on concepts from XAI. By eliminating the

less relevant components, the model can be streamlined and made more interpretable without compromising its performance. Furthermore, intrinsic XAI models, including Interpretable CNN, CNN Explainer, XCNN, and FCM, have been developed to improve the interpretation of CNNs. These models incorporate additional elements such as loss functions, autoencoders, cluster algorithms, and decision trees to provide interpretable insights into the decision-making process of CNNs. However, caution must be exercised when integrating these XAI models into critical applications like medical diagnosis, as the introduction of these components may impact the overall accuracy of CNN classification [IS23].

Another issue, though less prevalent in medical imaging CNNs, is intentional manipulation of input data aimed at deceiving CNNs and causing misclassification. A promising strategy to mitigate such attacks is training with perturbed samples. By combining a convolutional denoising autoencoder with a classifier, a defensive structure can be constructed to enhance the CNN's robustness and reliability. This allows the model to learn to differentiate between genuine and perturbed samples, thus reducing its vulnerability to negative attacks [AK22].

Furthermore, one of the major challenges in medical image classification is the lack of sufficient data, expert commentary, high imaging costs, and data privacy concerns. As a result, there is a limited availability of large, comprehensive, and high-quality datasets in radiology. This scarcity makes it difficult to effectively train and evaluate convolutional neural networks in this domain. In the work of [Car+22b], it has been shown that certain QML algorithms may generalize better in the presence of only a small amount of training data. In the team where I conducted my thesis at the Fraunhofer Institute, they have previously researched quantum-classical convolutional neural networks and their performance on medical data. This research has shown promising results, indicating that quantum-variants of CNNs have the potential to outperform classical CNNs in this domain [Mat+22]. Building upon this previous work, my thesis aimed to further enhance the results by incorporating a pooling mechanism.

While advancements in computer hardware and software have simplified the training and deployment of CNNs in medical imaging, effective data selection and interpretation remain crucial factors for successful utilization. Proper curation, preprocessing, and annotation of data are essential to ensure that CNNs receive high-quality input that accurately represents real-world scenarios encountered in radiological imaging. Additionally, efforts to ensure the ethical and responsible use of CNNs in the healthcare sector are of utmost importance to avoid undesirable consequences, including potential biases or discriminatory outcomes.

3 Introduction to quantum computing

3.1 Quantum bits

The qubit, or quantum bit, is a fundamental concept in quantum computation and quantum information. Unlike classical bits, which have only two possible states, qubits are in superposition states that are linear combinations of $|0\rangle$ and $|1\rangle$. Mathematically, qubits are abstract entities that can be represented as vectors in a two-dimensional complex vector space. The two special states $|0\rangle$ and $|1\rangle$ are the computational basis states and form an orthonormal basis for this vector space [NC02].

The properties of qubits can be visualized geometrically using the Bloch sphere. The Bloch sphere is a unit sphere that represents the state space of a single qubit. The north and south poles of the sphere correspond to the states $|0\rangle$ and $|1\rangle$, respectively, while all other points on the surface of the sphere correspond to superposition states:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle$$

The range of values for θ and φ such that they cover the whole sphere is $\theta \in [0, \pi)$ and $\varphi \in [0, 2\pi)$. Angle θ corresponds to lattitude and angle φ corresponds to longitude.

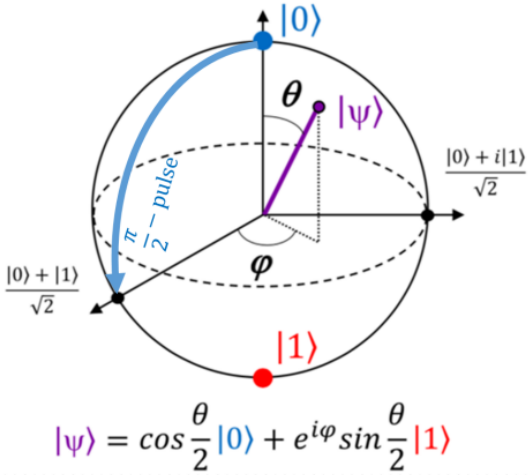


Figure 3.1: Bloch sphere representation of a qubit [Jaz+19].

This establishes a direct mapping between the qubit state and the position on the sphere, ensuring a one-to-one relationship. The Bloch sphere, however, does not account for the global phase since states $|\psi\rangle$ and $e^{i\lambda}|\psi\rangle$ are considered equivalent. Therefore, it can be used to visualize the effect of quantum gates and operations on a qubit's state, which can be represented as a rotation of the sphere around different axes.

The state of a qubit can be manipulated and transformed in ways that lead to measurement outcomes that depend distinctly on the different properties of the state. When a qubit is measured, it collapses into either the $|0\rangle$ or $|1\rangle$ poles with probabilities determined by the magnitudes of the coefficients of the corresponding state in the superposition. In the context of the Bloch representation, the probability of a quantum state collapsing to a particular pole is directly proportional to the distance of the corresponding arrow from that pole. Specifically, a state whose arrow is closer to the north pole has a higher probability of collapsing to the north pole, and vice versa for the south pole. The angle θ between the arrow and the vertical axis represents the magnitude of this probability.

The power of quantum computation and quantum information lies in the ability to manipulate and transform qubits and their states in ways that allow for efficient and powerful algorithms for solving problems that are intractable for classical computers.

3.1.1 Multiple qubits

The behavior of qubits, becomes intriguing when we consider multiple qubits, indeed while a single qubit quantum computer may not be as interesting, as it can be seen as a way of performing probabilistic computations, the introduction of multiple qubits opens up new possibilities. For a two-qubit system, there are four computational basis states: $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. A quantum state of two qubits involves associating a complex coefficient, or amplitude, with each computational basis state. The state vector describing a two-qubit system is then given by

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

where α_x is the amplitude associated with basis state $|x\rangle$. Similar to a single qubit, the probability of measuring a particular outcome x is given by $|\alpha_x|^2$, and the post-measurement state collapses to $|x\rangle$. The normalization condition requires that the sum of the probabilities for all possible outcomes is equal to one, i.e., $\sum_{x\in[0,1]^2} |\alpha_x|^2 = 1$, where $[0,1]^2$ denotes the set of all binary strings of length two.

When measuring only one qubit in a two-qubit system, the post-measurement state is re-normalized by a factor to ensure the normalization condition is satisfied. An important two-qubit state is the Bell state given by

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}.$$

The Bell state is responsible for many surprising phenomena in quantum computation and quantum information, such as quantum teleportation that uses two classical bits to transmit a single qubit in an unknown quantum state and superdense coding that allows the transmission of two classical bits of information using only one qubit . Upon measuring the first qubit of a Bell state, there is a 50% chance of obtaining 0, leaving the post-measurement state $|\varphi'\rangle = |00\rangle$, and a 50% chance of obtaining 1, leaving $|\varphi'\rangle = |11\rangle$.

Therefore, measuring the second qubit always gives the same result as measuring the first qubit, suggesting that the measurement results are related. Correlations in the Bell state have been a topic of intense interest since the famous work of Einstein, Podolski, and Rosen in which they first drew attention to the strange properties of these states.

A quantum circuit can accept an input of n qubits, where each qubit can exist in a superposition of $|0\rangle$ and $|1\rangle$.Therefore the total number of possible states is 2^n and ranges from $|0\ldots0\rangle$ to $|1\ldots1\rangle$. Each quantum superposition of n qubits can be represented by:$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle$.

Quantum computers possess the ability to process data at significantly higher speeds compared to classical computers and supercomputers. A notable illustration of this is the simulation of a fully entangled 500-qubit system, which would necessitate a computer with a storage capacity of 2^{500} bits, surpassing the estimated number of atoms in the universe. This aspect holds great potential for simulation purposes. Researchers are actively working towards leveraging this immense computational power to simulate molecules. Notably, a breakthrough was achieved by scientists from Harvard and Google, who demonstrated the utilization of a quantum computer for modeling electron interactions in a complex molecule [Har21]. As early as 1981, physicist Richard Feynman, a Nobel laureate, predicted that quantum computers based on quantum mechanics could accurately simulate large molecules []. Quantum computers offer the possibility of directly simulating systems governed by quantum principles, such as molecules or materials, owing to the quantum nature of their constituent quantum bits. Recent experiments have exhibited the remarkable capabilities of these devices when performing carefully chosen tasks [].

Nevertheless, manipulating and controlling the quantum states of multiple qubits to execute useful calculations remains challenging and prone to errors. Consequently, researchers are exploring two primary avenues: increasing the number of qubits in a system and implementing error correction or mitigation techniques.

3.2 Quantum computation

3.2.1 Single qubit gates

Single qubit gates are an essential building block in quantum circuits, just as classical logic gates are in classical circuits. They are used to manipulate the state of a single qubit, which can be represented as a superposition of the basis states $|0\rangle$ and $|1\rangle$. The action of a single qubit gate is defined by its effect on the amplitudes of the basis states [NC02].

For example, a quantum NOT gate, which is analogous to the classical NOT gate, maps $|0\rangle$ to $|1\rangle$ and vice versa. However, it also has a nontrivial effect on superpositions of $|0\rangle$ and $|1\rangle$. Specifically, a quantum NOT gate maps the state

$$\alpha|0\rangle + \beta|1\rangle$$

to the state

$$\alpha|1\rangle + \beta|0\rangle.$$

This action is linear and reversible, which is a general property of quantum mechanics. It can be represented by a 2x2 matrix, where the first column corresponds to the output when the input is $|0\rangle$ and the second column corresponds to the output when the input is $|1\rangle$. In the case of the quantum NOT gate, this matrix is given by

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The matrix representation of a single qubit gate must satisfy certain constraints. In particular, the matrix must be unitary, meaning that its conjugate transpose times itself is equal to the identity matrix:

$$U^\dagger U = I,$$

where $U^\dagger$ is the conjugate transpose of U and I is the 2x2 identity matrix. This condition ensures that the gate preserves the normalization of the quantum state, which requires that the sum of the squared magnitudes of the amplitudes is equal to 1. In other words, the total probability of measuring a qubit in any state must be 1.

Any 2x2 unitary matrix can be used as a valid single qubit gate. This means that there are many nontrivial single qubit gates beyond the quantum NOT gate. A non-trivial single qubit gates is the Z gate:

$$Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

which leaves $|0\rangle$ unchanged, and flips the sign of $|1\rangle$ to give $-|1\rangle$. Another important gate is the Hadamard gate, which maps $|0\rangle$ to a superposition of $|0\rangle$ and $|1\rangle$ and $|1\rangle$ to a different superposition of $|0\rangle$ and $|1\rangle$:

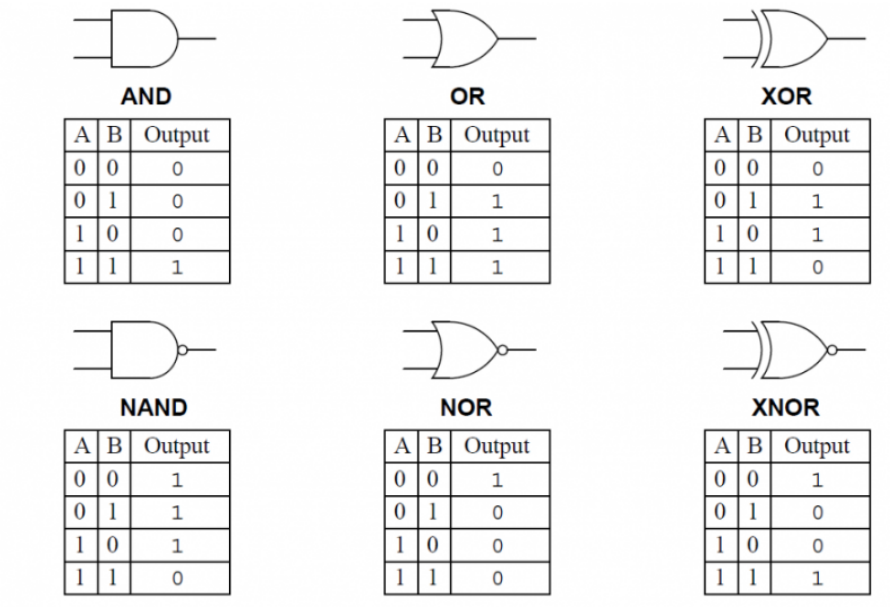
$$H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

This gate is often used to create superposition states, which are important for many quantum algorithms. Single-qubit gates are indispensable for manipulating the state of a single qubit in quantum circuits, enabling arbitrary unitary operations and playing a vital role in practical quantum circuit design, as well as being a fundamental requirement for universal quantum computing.

Figure 3.2: Visualization of the action of the Hadamard gate on the Bloch sphere on a qubit in state $|0\rangle$ [Uni].

3.2.2 Multiple qubit gates

Multiple bit gates are essential components in classical and quantum computing circuits. In classical computing, six notable multiple bit gates are the AND, OR, XOR, NAND, NOR and XNOR gates.



AND

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

XOR

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	0

NAND

A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

NOR

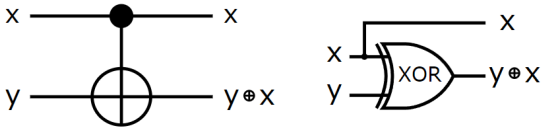
A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

XNOR

A	B	Output
0	0	1
0	1	0
1	0	0
1	1	1

Figure 3.3: Classical multiple bit gates: AND, OR, XOR, NAND, NOR, XNOR [SII21]

In quantum computing, the prototypical multi-qubit logic gate is the controlled-NOT or CNOT gate, which has two input qubits, the control qubit and the target qubit. The CNOT gate can be seen as a generalization of the classical XOR gate since the control qubit and the target qubit are XORed and stored in the target qubit.



input		output	
x	y	x	y+x
|0⟩	|0⟩	|0⟩	|0⟩
|0⟩	|1⟩	|0⟩	|1⟩
|1⟩	|0⟩	|1⟩	|1⟩
|1⟩	|1⟩	|1⟩	|0⟩

input		output	
x	y	x	y+x
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Figure 3.4: XOR classical = CNOT quantum [Wik21]

3.2.3 Common one-qubit and two-qubits gates

The most commonly used 1-bit gates and their effect on the input qubits are as follow:

Name	Symbol	Matrix Form	Action upon the Qubit
Identity	I	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	No action
Pauli-X	X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	Rotation by π around the X axis
Pauli-Y	Y	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	Rotation by π around the Y axis
Pauli-Z	Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	Rotation by π around the Z axis
Hadamard	H	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	Creates a 50% chance superpositior
Rotational X	$R_X(\theta)$	$\begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$	Rotation by θ around the X axis
Rotational Y	$R_Y(\theta)$	$\begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{bmatrix}$	Rotation by θ around the Y axis
Rotational Z	$R_Z(\theta)$	$\begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$	Rotation by θ around the Z axis

The most commonly used 2-bit gates and their effect on the input qubits are as follow:

Name	Symbol	Matrix Form	Action upon the Qubit
Controlled-X gate	CX	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$	Apply X gate on target if control=$\|1\rangle$.
Controlled-Y gate	CY	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{pmatrix}$	Apply Y gate on target if control=$\|1\rangle$.
Controlled-Z gate	CZ	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$	Apply Z gate on target if control=$\|1\rangle$.

3.3 NISQ devices

NISQ is an acronym for "Noisy Intermediate-Scale Quantum" computing systems that comprise a limited number of qubits, typically on the order of about 100. These devices exhibit limited connectivity and gate fidelity, which affects their overall performance[12].

In the realm of quantum computing, he deleterious effects of noise and decoherence complicate the full realization of the computational potential of these systems.The presence of errors is a fundamental limitation across all quantum technologies. To address this challenge, the scientific community is actively engaged in researching quantum error correction and mitigation techniques aimed at mitigating the effects of errors on quantum computations.

Currently, there are three major categories of quantum processing devices: circuit-based, annealing-based, and analog-based. Each of these technologies has unique advantages and limitations, leading researchers to further improve their functionalities.

In particular, quantum algorithms such as VQE (Variational Quantum Eigensolver) take advantage of the comparatively modest computational requirements of NISQ computers. They enable the solution of optimization problems in industries such as finance, logistics, and chemistry. This approach minimizes the burden on quantum hardware while still leveraging the power of quantum computation.

Quantum Computer Challenges		
QC Requirement	Why Challenging?	Potential Solutions
Entanglement	-Key quantum mechanics -Gives QCs their calculation power -Maintaining entanglement is the challenge	- Isolate qubits from environmental noise - Fault-tolerant system architecture - QC technology with less noise vulnerability
Decoherence	- Irreversible loss of qubit information in QCs	- Superconducting circuits - Quantum dots - Color center in diamonds or other crystals
Qubit error correction	-Quantum information cannot be copied, which prevents normal error-correction techniques - QC error correction is feasible, more complex	- Logical qubits in addition to physical qubits - Multiple error-prone physical qubits work together to mimic a stable single qubit
Low data transfer rate	- QCs need faster 1/0 data rate to feed apps - Lowers QC utilization rate and usage value	- Not clear how to solve this issue yet - Increasing need as deployment takes off
Deployment	Why Challenging?	Potential Solutions
Low-temp operation	- Most QC technologies need extremely low temperature - Needed to minimize decoherence	- Superconducting tech advances - QC tech operating near room temperature
Multiple QC technologies	- QC technology battles market uncertainty - Fractured investments, expertise, and resources	- One QC tech becomes dominant - Maybe two QC techs become clear leaders
Qubit fabrication	- Minimal capacity to manufacture qubit systems - Minimal qubit parts supply chain	- Qubit manufacturing investments - Qubit parts supply chain investments
Software ecosystem	-Many software platforms:QCalgorithms, QCsoftware apps, QC SDKs, QC control software - PC software platforms to interface and control QCs	- Open-source software platforms - Hardware abstraction and APIs for multiple QCs: Across generations and QC technologies
High cost and complexity	Rapid changes, lab production volume, QC tech competition, limited workforce	- More investments and higher volumes - More planning and cooperation
(Source: [GAO19])		

3.4 Quantum Machine Learning

Quantum Machine Learning (QML) is an emerging field that sits at the intersection of quantum computing and machine learning. Its primary goal is to harness the unique properties of quantum computing to enhance and accelerate machine learning algorithms. QML researchers are working towards developing faster and more accurate learning algorithms than what is currently achievable with classical computing [Bia+17].

During the current NISQ era, with limited connectivity and relatively low gate fidelities, QML research is primarily focused on two directions for near term applications. The first direction involves using quantum computing-based subroutines to accelerate traditional machine learning techniques.These subroutines could take advantage of the quantum computer's large parallel computing power to perform calculations faster than a conventional computer might for the same task. The second direction involves studying parameterized quantum circuits, also called variational quantum circuits (VQC). These circuits can be trained using classical optimizers. The goal is to achieve quantum advantages, where quantum algorithms outperform classical algorithms on certain tasks.

Quantum machine learning (QML) research is driven by the extensive state space and computational potential offered by quantum systems, as well as their ability to represent mappings that are challenging or computationally intractable in classical settings.

However, training quantum neural networks and optimizing them pose challenges due to the non-convex nature of the loss landscape and the presence of barren plateaus. Barren plateaus refer to regions in the parameter space where gradients diminish exponentially as the number of qubits increases. Consequently, finding optimal parameters that minimize the loss function and improve algorithm performance becomes exceedingly difficult. To address this issue, various approaches have been explored, including introducing noise during training, employing gradient rescaling or resampling, and developing optimization algorithms specifically tailored for quantum systems [McC+18].

In their work [Abb+21], Abbas et al. investigated the expressiveness and learnability of quantum models, demonstrating that well-designed quantum neural networks can surpass classical neural networks in terms of higher effective dimensions and faster learning capabilities. The authors utilized information geometry tools to define the concept of expressibility for both quantum and classical models, introducing a novel generalization limit based on effective dimension, which relies on Fisher information. Additionally, they established a connection between the Fisher information spectrum and barren plateaus, highlighting that certain quantum neural networks exhibit resilience to this problem and can achieve faster training compared to classical

models. This advantage arises from their optimization landscapes, characterized by a more uniformly distributed Fisher information spectrum.

3.4.1 Quantum Data Encoding

In quantum machine learning, a feature map is a unitary transformation that embeds classical data into a quantum state of Hilbert space. The choice of the feature map is essential as it determines the expressive power of parametrized quantum circuits as function approximators. Indeed, the encoding influences the available frequencies and the richness of the frequency spectrum that the model can access. By using different encoding strategies, quantum models can therefore learn different types of functions, and this can affect the generalization performance of the model [SSM21]. Thus, selecting an appropriate data encoding strategy is crucial to ensure that the model can accurately learn from the input data and generalize to new data. Several methods for data embedding exist, including basis encoding, amplitude encoding, and higher order encoding.

Threshold encoding Using a certain threshold t, an input x is encoded to the quantum state $|0\rangle$ if $x < t$, and otherwise to the state $|1\rangle$.

Basis encoding associates the input, represented as a natural number $N \in \mathbb{N}$, with a quantum basis state. The input is converted into its binary string representation $b_1 b_2 \dots b_n$, where n is the number of bits necessary to represent the number in binary, $2^{n-1} \leq N \leq 2^n$., and each qubit in the quantum state passes through a sequence of $R_X(\theta)$ and $R_Z(\theta)$ gates.If $b_i = |0\rangle$, then $\theta = 0$, the qubit remains the same. If $b_i = |1\rangle$, then $\theta = \pi$, the qubit is negated. The basis encoding needs $n = \lceil \log(N) \rceil$ qubits.

Amplitude encoding, on the other hand, maps the input vector $x = [x_1, x_2, \dots, x_{2^n}]^T$ into a quantum state using the amplitudes of the basis states of the system. The vector is first normalized to ensure that its elements are in the complex field and sum to 1: $\sum_{i=0}^{2^n} |x_i|^2 = 1$. The resulting quantum state is $|\phi_x\rangle = \sum_{i=1}^{2^n} x_i |i\rangle$, allowing for the encoding of 2^n features into n qubits. If the number of features is not a power of 2, the vector is padded with constant values to reach the closest higher power of 2.

The **higher order encoding** method is the one that was used. This method uses two-qubit gates, in addition to single-qubit gates, leading to an entangled encoding feature map. This technique starts with qubits in state $|0\rangle$ and applies a Hadamard gate and a Z-axis rotation $R_Z(x_n)$, where x_n denotes the n-th input. An entangling operation $R_{ZZ}(\phi_{ij})$ is then applied to every qubit pair i and j, consisting of a CNOT gate, a rotation $R_Z(\phi_{ij})$, and another CNOT gate. The rotation $R_Z(\phi_{ij})$ is applied to the j-th qubit, using $\phi_{ij} = x_i * x_j$, where x_i and x_j are inputs.

The higher order encoding technique is particularly interesting for quantum machine learning models, as it was shown that for support vector machines, quantum advantage

can only be achieved if the encoding feature map is difficult to simulate classically [Hav+19].

3.4.2 Trainable variational circuits layer

The trainable quantum layer is a fundamental component in many quantum machine learning models, which consists of various rotational gates that transform the input quantum state into the desired output state. The trainable quantum layer can be expressed as:

$$U_\Theta = e^{-i\theta_m U_m} e^{-i\theta_{m-1} U_{m-1}} ... e^{-i\theta_1 U_1}$$

In this representation, U_i represents the i-th rotational gate, and θ_i represents its corresponding trainable parameter. The set of trainable parameters is denoted as $\Theta = \{\theta_1, \theta_2, ..., \theta_m\}$. These parameters are learned during the training process to optimize the output of the quantum circuit.

Each term $e^{-i\theta_k U_k}$ in the expression corresponds to a rotation of the quantum state around the unitary operator U_k by an angle θ_k. By applying these successive rotations, the trainable quantum layer transforms the input quantum state according to the learned rotation angles to produce the desired output.

The trainable quantum layer includes various rotational gates, including RX, RY, RZ, and U gates, along with their controlled variants. These gates are applied to the input quantum state in a specific order, which can be arbitrary. The placement of these gates can be optimized during the training process to obtain the best performance of the quantum circuit.

Ideally, the experiments are simulated using software libraries such as PennyLane and Qiskit to present the results. The purpose behind this choice is to overcome limitations present in current quantum hardware, such as long waiting queues and concerns regarding reproducibility. Furthermore, simulation allows for more efficient exploration of different architectures and optimization techniques, while providing a flexible and accessible environment for experimentation. The input to the trainable quantum layer is the quantum state prepared by the embedding circuit, and the output is obtained by applying the U_Θ ensemble of gates to this state.

3.4.3 Quantum measurement

In VQCs, a measurement is performed to extract classical information from the quantum state generated by running a quantum circuit on n qubits. After the measurement, the output can be represented as a set of 2^n probabilities of each basis state of the entire n-qubit output or as n expectation values of each qubit.

To compute the expectation value of a qubit, we use the formula:

$$\langle \psi | Z_j | \psi \rangle = \mathrm{Tr}(Z_j \rho)$$

where $|\psi\rangle$ is the quantum state generated by the quantum circuit, Z_j is the Pauli Z operator acting on the j-th qubit, and ρ is the density matrix of the quantum state. This formula gives us the average value of the observable Z_j in the state $|\psi\rangle$.

To obtain the expectation values of all n qubits, we apply the formula for each qubit index j. This gives us a set of n expectation values $\langle Z_0 \rangle, \langle Z_1 \rangle, ..., \langle Z_{n-1} \rangle$.

3.4.4 Classical Optimization Loop

Optimizing a VQC is crucial to achieve accurate quantum results. The VQC generates an output that is compared to the expected output in supervised learning. This comparison results in a loss function, which is sent to a classical optimizer. The optimizer updates the quantum rotational parameters based on the loss value and optimization strategy. Although the classical optimizer does not use quantum technology, it is vital for optimizing the VQC. This optimization process is iterative, meaning that the parameters are repeatedly adjusted and measurements taken until the VQC output is optimized. This occurs either when the maximum epoch is reached or the desired accuracy is achieved.

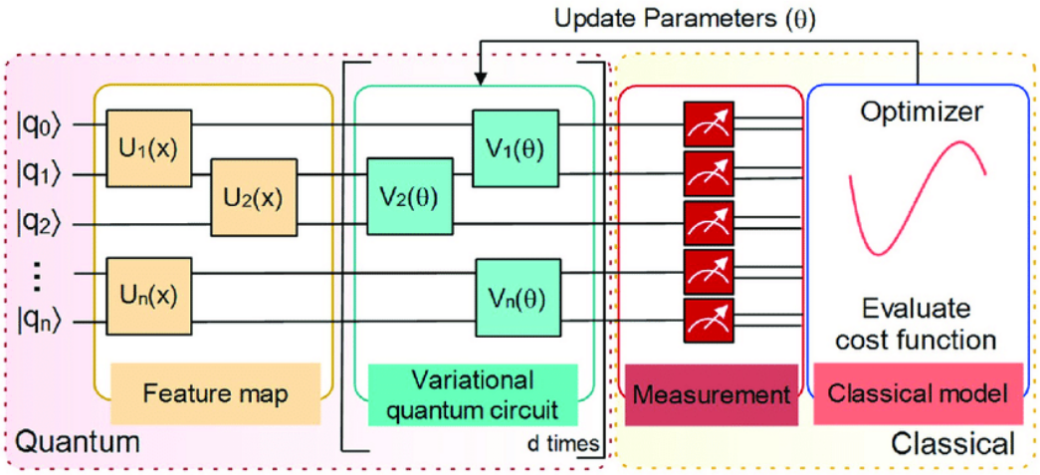


Figure 3.5: Typical VQC based QML pipeline [Sen+22]

3.4.5 The Effective Dimension: A Measure of Model Complexity

The effective dimension (ED) serves as a valuable metric for assessing the generalization power and fitting ability of classical and quantum machine learning models. Derived from the Fisher Information Matrix, a statistical technique that measures the impact of parameter variances on output probabilities, it offers an alternative complexity measure inspired by information geometry.

The primary objective of the effective dimension is to estimate the model's size within the expansive model space, which represents the set of all possible functions for a specific model class. In this context, the Fisher information matrix acts as the underlying metric. By leveraging the number of data observations, the ED incorporates a natural scale or resolution for observing the model space. This is particularly advantageous when working with limited data, as it sheds light on how data availability influences the accurate assessment of model complexity.

The effective dimension can be mathematically defined as follows:

$$d_{\gamma,n}(\mathcal{M}\Theta) = 2\frac{\log\left(\frac{1}{V\Theta}\int_\Theta \sqrt{\det(\mathrm{id}_d + \frac{\gamma n}{2\pi\log n}\hat{F}(\theta))}d\theta\right)}{\log\left(\frac{\gamma n}{2\pi\log n}\right)},$$

where $\mathcal{M}\Theta = p(\cdot,\cdot;\theta) : \theta \in \Theta$ is a statistical model with a d-dimensional parameter space $\Theta \subset \mathbb{R}^d$ and n data samples. $V\Theta = \int_\Theta d\theta$ is the volume of the parameter space, and $\hat{F}(\theta) \in \mathbb{R}^{d\times d}$ is the normalized Fisher information matrix defined as:

$$\hat{F}ij(\theta) = d\frac{V\Theta}{\int_\Theta \mathrm{tr}(F(\theta))d\theta}F_{ij}(\theta),$$

where $F_{ij}(\theta)$ is the Fisher information matrix evaluated at θ. The normalization ensures that

$$\frac{1}{V_\Theta}\int_\Theta \mathrm{tr}(\hat{F}(\theta))d\theta = d$$

.

In the above equation, $\gamma \in (0,1]$ and $\log n$ are constant factors proposed by [Ber+20a] and extended by [Abb+21] establish a generalization bound. The regularization parameter γ is employed to prevent overfitting by limiting the model's complexity. It controls the trade-off between model complexity and accuracy, where a larger value of γ corresponds to a simpler model with lower accuracy, and a smaller value of γ leads to a more complex model with higher accuracy. The inclusion of the $\log n$ term ensures that the effective dimension scales logarithmically with the number of data samples, effectively avoiding an overestimation of the model's complexity.

4 Quantum classical convolutional neural networks: SOTA and motivation

Quantum variants of convolutional neural networks (CNNs) can be realized in different ways. One possibility is to transfer all the components of a CNN onto a quantum computer (QC), resulting in what is known as a quantum convolutional neural network (QCNN) [CCL19]. However, this approach generally relies on the availability of quantum random access memory (QRAM), which is currently not accessible on near-term intermediate-scale quantum (NISQ) devices.

QRAM is a quantum algorithm that allows classical data to be converted into corresponding quantum states. Unfortunately, implementing the QRAM approach requires a significant number of qubits to convert classical data into quantum states, making it impractical for use in current NISQ devices.

Some studies have explored the potential of executing specific parts of a CNN on a QC, considering that it may be feasible with existing hardware due to its lower qubit requirements. For instance, the initial classical convolutional layer in a CNN can be substituted with either an untrainable or trainable quantum convolutional layer. By adopting this architecture, previous works have achieved promising performance on the MNIST dataset [Mat+22].

4.1 QCNN architecture

In [CCL19] the authors propose a novel quantum machine learning model called Quantum Convolutional Neural Network (QCNN) inspired by convolutional neural networks (CNNs). QCNN efficiently handles quantum machine learning tasks with O(log(N)) variational parameters for N qubits, enabling efficient training and implementation on near-term quantum devices. The QCNN architecture combines Multi-scale Entanglement Renormalization Ansatz (MERA) a representation of a quantum state generated by a series of unitary and isometry layers and quantum error correction (QEC), making it powerful for solving quantum many-body problems. The paper demonstrates the potential of QCNN through two examples. Firstly, accurately recognizing quantum states associated with 1D symmetry-protected topological phases using a QCNN trained on a small set of solvable points. Secondly, optimizing a quantum

error correction scheme for a given error model by proposing a generic framework for encoding and decoding procedures, outperforming known quantum codes. The QCNN circuit model is inspired by classical CNNs and includes convolution and pooling layers implemented using quantum operations. It takes an unknown quantum state, applies convolution and pooling operations to reduce system size, and obtains output through measurements. The unitaries in the circuit are learned through training on labeled data. The authors relate QCNN to MERA and QEC, highlighting the underlying mechanisms. QCNN combines MERA and nested QEC, with pooling layers acting as syndrome measurements for error correction. This enables QCNN to mimic renormalization group flow and efficiently classify quantum states or optimize error correction schemes. However, due to the relatively large number of variational parameters, the current implementation is on classical computers rather than short-term quantum devices.

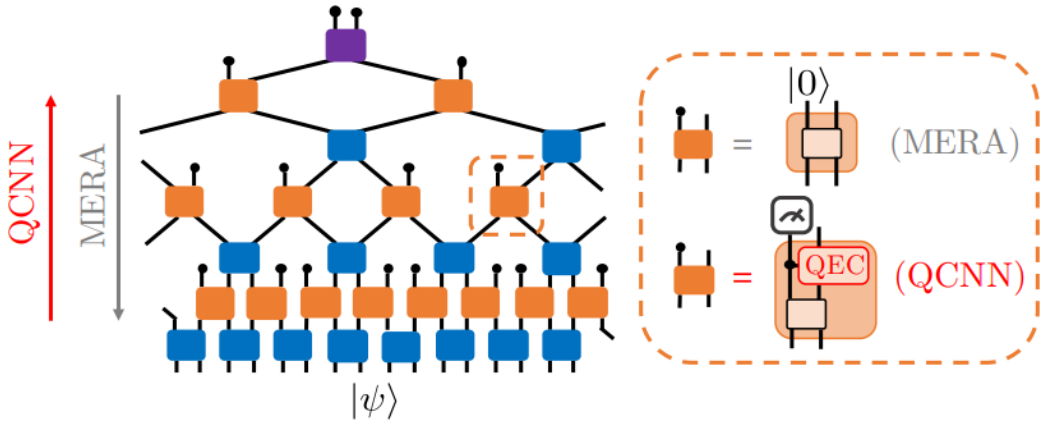


Figure 4.1: QCNN and MERA share the same circuit structure, but run in reverse directions. [CCL19]

In a rigorous analysis of gradient scaling for QCNN parameters, it has been discovered that QCNNs do not suffer from exponentially vanishing gradients, also known as barren plateau landscapes, unlike many other QNN architectures. The gradient variance of QCNNs decreases polynomially, ensuring their trainability even under random initialization. This result provides an analytical guarantee and highlights the unique characteristics of QCNNs [Pes+21].

Another study benchmarked QCNNs for classification tasks on classical data, specifically focusing on pattern recognition. Various aspects of the QCNN algorithm were explored, including the structure of parameterized quantum circuits, quantum and classical data encoding methods, pre-processing techniques, cost functions, and optimizers. The experiments demonstrated that QCNNs, with a small number of free

parameters, achieved high classification accuracy comparable to or even surpassing traditional convolutional neural networks under similar training conditions. The advantage of QCNNs was attributed to their ability to capture global correlations through entanglement, while CNNs can only capture local correlations [HKP22].

In a different approach, a novel variational circuit ansatz called branching QCNN (bQCNN) was introduced. It leverages mid-circuit measurements and classical control flow capabilities of quantum devices, allowing for the execution of subsequent quantum operations based on measurement results. The bQCNN was found to be significantly more expressive than the original QCNN, indicating its potential for a wider range of quantum machine learning tasks on near-term devices. The utility of mid-circuit measurements in near-term quantum applications was emphasized, and possibilities for future hybrid quantum-classical approaches were discussed [Mac+22].

Overall, while QCNNs have attracted attention for their potential in quantum data analysis, there has been relatively limited emphasis on pooling operations. Only a few papers have explored the utilization of certain pooling techniques, such as mid-circuit measurement. However, the exploration of different types of pooling methods in QCNN research has been relatively scarce.

4.2 QCCNN architecture

In the QCCNN architecture, the classical convolutional layer found in a conventional CNN is replaced with a quantum convolutional layer. The classical CNN typically comprises a single convolutional layer utilizing a 2×2 filter size, followed directly by a fully connected layer. In contrast, the QCCNN architecture incorporates a quantum convolutional layer using the same 2×2 filter size. The QCCNN employs the higher order encoding technique, which is expected to yield optimal performance for quantum convolutional layers. Within the quantum convolutional layer, a trainable component known as the Variational Quantum Circuit (VQC) is utilized. Subsequently, each qubit is measured in the Z-basis, and the resulting measurements are stored individually in feature maps. With a filter size of 2×2 and a one-to-one mapping of input values to qubits, the QCCNN employs four qubits and generates four feature maps.

The QCCNN architecture effectively tackles the input and output challenges faced by current quantum machine learning algorithms. The input problem pertains to the computational complexity involved in encoding classical data into quantum states, while the output problem relates to the limited expressive power of traditional quantum circuits. By employing a higher order encoding technique and storing measurements in feature maps, the QCCNN successfully overcomes these obstacles.

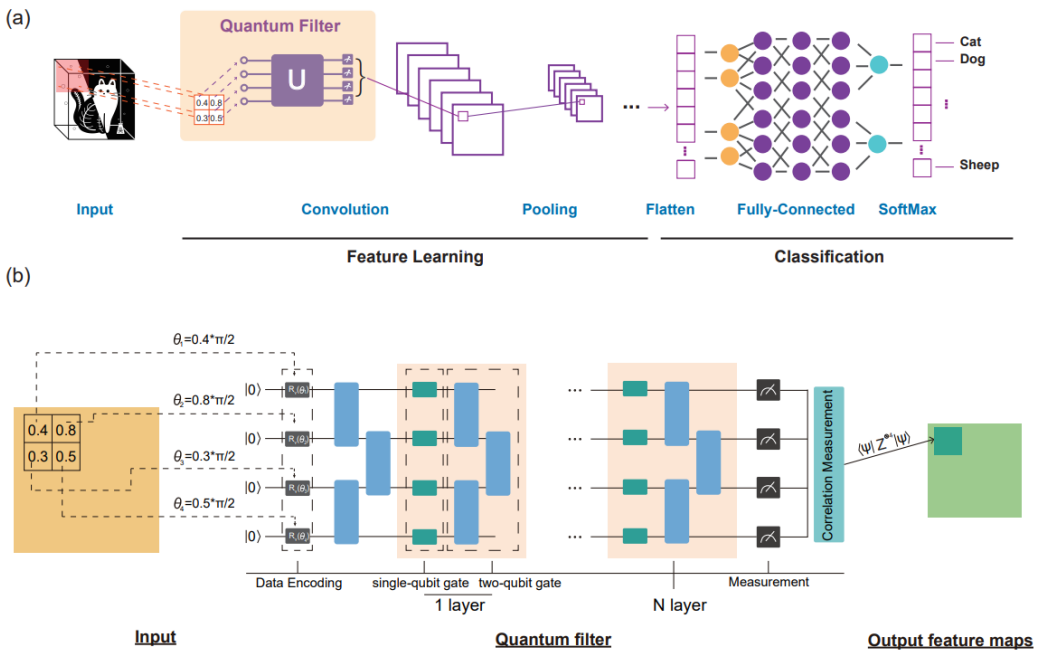


Figure 4.2: Hybrid Quantum-Classical Convolutional Neural Network (QCCNN): (a) Overall Architecture, (b) VQC Architecture [Liu+21]

5 QCCNNS with a quantum pooling layer

5.1 Motivation

Hybrid quantum-classical structures have gained increasing popularity in the field of quantum machine learning due to their potential for solving complex problems beyond classical computers' capabilities. However, limited research exists on hybrid classical-quantum structures, as most attention has been focused on fully quantum solutions. The pooling mechanism within these architectures, which plays a crucial role in classical machine learning, has been relatively understudied in the context of quantum machine learning.

Pooling mechanisms are instrumental in extracting relevant features, reducing feature map dimensionality, minimizing variance, and accelerating the learning process in classical machine learning. Maximum pooling is effective for capturing low-level objects such as edges and points, while medium pooling is suitable for extracting smooth objects. Therefore, a comprehensive investigation of the pooling mechanism in hybrid quantum-classical structures could be valuable.

This thesis aims to conduct a comprehensive investigation into the pooling mechanism within hybrid quantum-classical structures. The primary focus is to explore the effects of four distinct pooling techniques on both model accuracy and effective dimensionality. While three of the four pooling techniques were implemented by myself, the fourth technique, namely modular quantum pooling blocks, was implemented by another researcher within the lab.

The objectives of this thesis are as follows:

1. Designing and evaluating four distinct quantum pooling architectures for hybrid quantum-classical Convolutional Neural Networks (QCCNNs) tailored to ultrasound image analysis for malign lesion identification.

2. Comparing the performance of the proposed quantum pooling architectures with classical convolutional neural networks and quantum-classical CNNs without pooling. Models with similar numbers of trainable parameters were considered to highlight the advantages of pooling within the hybrid structures.

3. Providing an explanation for the performance of the developed models using a quantum-relevant metric. This analysis sheds light on the effectiveness of the employed pooling mechanisms.

It is noteworthy to acknowledge that the initial step in the implementation process involved utilizing the QCCNN code developed by [Mat+22] as a starting point and the incorporation of the effective dimension metric into the QCCNN framework was executed by one of my supervisors.

5.2 Experimental Setup

5.2.1 Dataset

In this study, we use the BreastMNIST dataset, which is an integral component of the MedMNIST datasets [Aa20][Ya21]. Despite its relatively modest size, with 546 training images, 156 test images, and 78 validation images, the BreastMNIST dataset contributes to the MedMNIST collection. It serves as a good resource for calibrating and comparing deep learning models, particularly in scenarios where limited data and low-resolution images are prevalent, as commonly encountered in medical imaging tasks. The BreastMNIST dataset comprises breast ultrasound images collected from a cohort of 600 patients, encompassing normal, benign, and malignant lesions. These images have been downscaled to a low-resolution of 28×28 pixels to facilitate benchmarking and comparison of deep learning models. Additionally, to enable binary classification of breast cancer, the dataset merges the normal and benign states into a non-malignant class, facilitating a clear distinction between non-malignant and malignant lesions.

To mitigate potential issues during model training, such as the occurrence of exploding gradients, the dataset was preprocessed by normalizing the data to have a mean of 0 and a standard deviation of 1. This normalization step ensures that the input data is centered and scaled appropriately, thereby facilitating stable and effective training of quantum convolutional neural networks (QCCNNs).

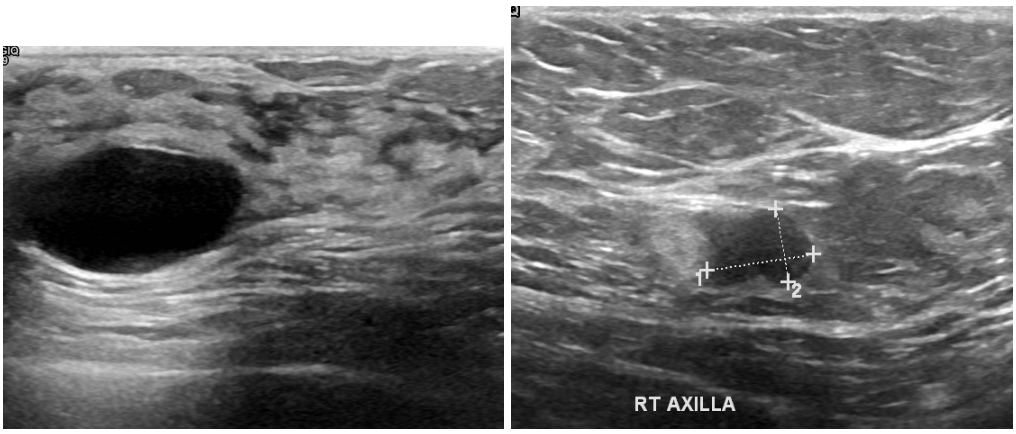


Figure 5.1: Benign vs Malignant breast ultrasound [Aa20]

5.2.2 Pennylane

Hybrid computing is at the heart of variational circuit optimization, where a quantum algorithm is optimized using a classical coprocessor. We chose PennyLane when coding and training our model, it is a Python 3 software framework for differentiable programming for quantum computers. The library provides a unified architecture for short-range quantum computing devices, supporting both qubit and continuous variable paradigms. It allows users to easily distribute quantum circuits to different quantum devices and seamlessly integrates classical machine learning libraries with quantum simulators and hardware, giving users the ability to train quantum circuits [Ber+18].

5.3 Problem statement

We consider a dataset consisting of N medical images ${x^{(i)}}_{i=1}^{N}$, each with dimensions 28×28, along with corresponding binary labels $y^{(i)} \in 0, 1$ indicating whether a tumor in the image is benign or malignant.

To classify the tumors, we propose a quantum-classical convolutional neural network (QCCNN) with parameters $\theta = \theta_q, \theta_c$, where θ_q and θ_c respectively represent the quantum and classical parameters of the model. Given an input image $x^{(i)}$, the QCCNN produces a probability distribution over the two classes, $p_\theta(y^{(i)}|x^{(i)}) = p_\theta(y^{(i)} = 0|x^{(i)}), p_\theta(y^{(i)} = 1|x^{(i)})$.

The network architecture starts with a quantum convolutional layer composed of one to four quantum circuits. These circuits are parameterized by a set of variables θ.

Each circuit processes the input image and generates one to four output feature maps. The total number of feature maps produced by this layer is four for all configurations explored in this thesis. The output from the quantum convolutional layer is then fed into a fully connected layer that uses another set of variables θ_c. This layer integrates information from the feature maps and outputs a classical probability distribution over two classes.

The QCCNN is trained by minimizing the cross-entropy loss between predicted and true labels for the entire dataset.

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N} y^{(i)} \log p_\theta(y^{(i)} = 1|x^{(i)}) + (1 - y^{(i)}) \log(1 - p_\theta(y^{(i)} = 1|x^{(i)}))$$

To optimize the parameters θ, we use the ADAM optimizer and update the parameters via gradient descent:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$$

where η is the learning rate and $\nabla_\theta \mathcal{L}(\theta)$ is the gradient of the loss with respect to the learnable parameters θ.

In contrast to the quantum convolution operation that yields four values, the quantum convolution + pooling operation, produces a single value. In order to establish a fair basis for comparison with the classical CNN and the QCCNN baseline, we utilize four quantum kernels concurrently to construct the quantum pooling architecture. This ensures that the classical layers maintain a consistent parameter count across various configurations. Moreover, this approach presents the benefit of extracting unique features from the image by employing four parallel quantum kernels. It is worth noting that the quantum pooling operation can be made trainable, for instance, through the utilization of mid-circuit measurements during the pooling process.

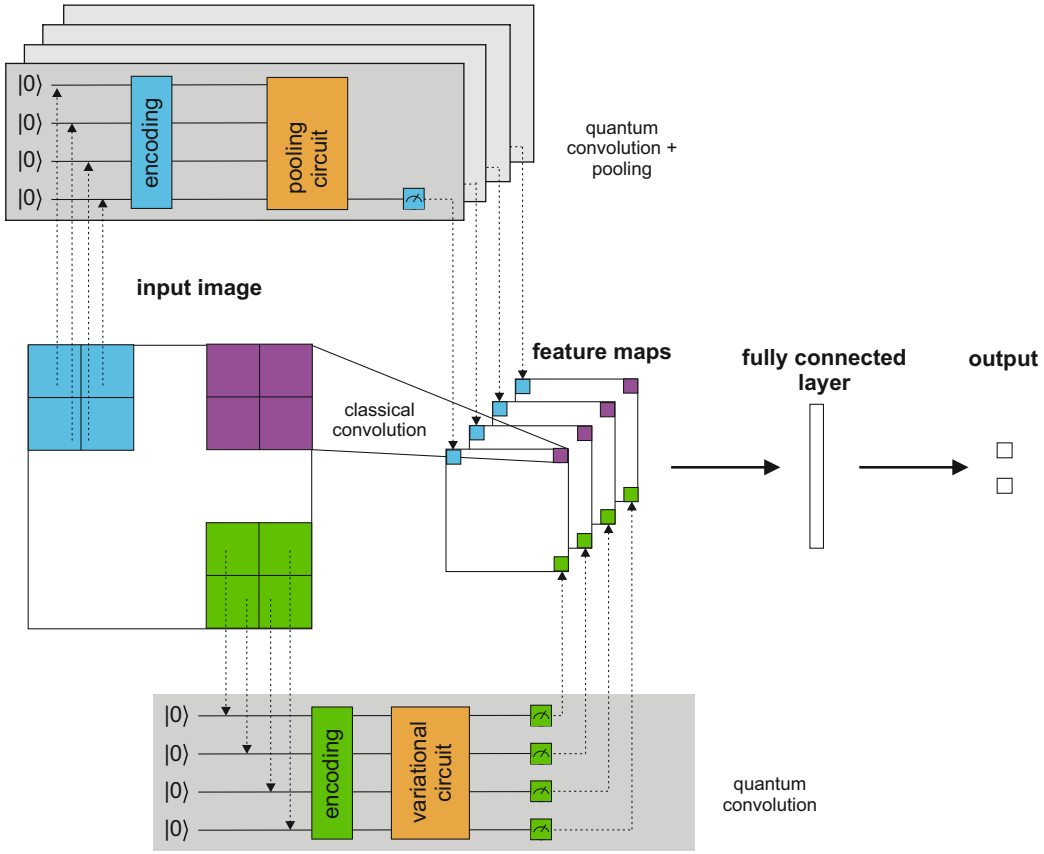


Figure 5.2: Architecture sketch of the CNN and QCCNNs with quantum convolutional layers with and without pooling.

5.4 Pooling methods

The performance of three different quantum and hybrid pooling techniques is studied: mid-circuit measurements, ancilla qubits with controlled gates, and qubit selection with classical postprocessing.

5.4.1 Mid circuit measurement

In the realm of quantum computing, the concept of mid-circuit measurements has garnered attention as a means to improve the efficiency and practicality of quantum circuits. By strategically measuring a portion of the quantum system during an inter-

mediate stage of the circuit, additional parameters can be introduced while reducing circuit complexity. This approach strikes a favorable balance between the number of parameters involved and the intricacy of the circuit.

Although the introduction of mid-circuit measurement in a quantum system can potentially disrupt the interference between quantum states and lead to computational errors, modern IBM quantum computers have made progress in mitigating this issue. Indeed, the use of mid-circuit measurements to change circuit composition, also known as dynamic circuits, has progressed to the point where it is now effectively implemented in hardware. This development was highlighted at the Qiskit Spring Challenge, where dynamic circuits were the main topic of the challenge and were successfully tested on real hardware machines, such as the 127-qubit Sherbrooke system. This is the largest qubit capacity made available to the general public to date [23].

In this master's thesis, mid-circuit measurements are employed to exert control over multiple qubits simultaneously, similar to a multi-qubit controlled gate. This technique proves advantageous as practical implementations of multi-qubit controlled gates often face hardware limitations [Chu+23].

The use of mid-circuit measurements facilitates the implementation of diverse paths within the circuit based on measurement outcomes. In a scenario involving four qubits labeled as q_0, q_1, q_2, and q_3, the state of qubit q_0 is measured. Subsequently, if the measurement outcome of q_0 is 1, rotation gates R_{θ_0}, R_{θ_1}, and R_{θ_2} are applied to qubits q_1, q_2, and q_3, respectively. Qubit q_1 is then measured, and if its measurement yields 1, rotation gates R_{θ_3} and R_{θ_4} are applied to qubits q_2 and q_3. Finally, a CNOT gate is applied between q_2 and q_3, the state of q_2 is measured, and if the measurement outcome is 1, rotation R_{θ_5} is applied to qubit q_3. The last qubit, q_3, undergoes measurement, and its value is fed into the classical fully-connected layer. Within this pooling method, two different alternatives are tested, employing rotation gates with trainable angles R_X and R_Y respectively.

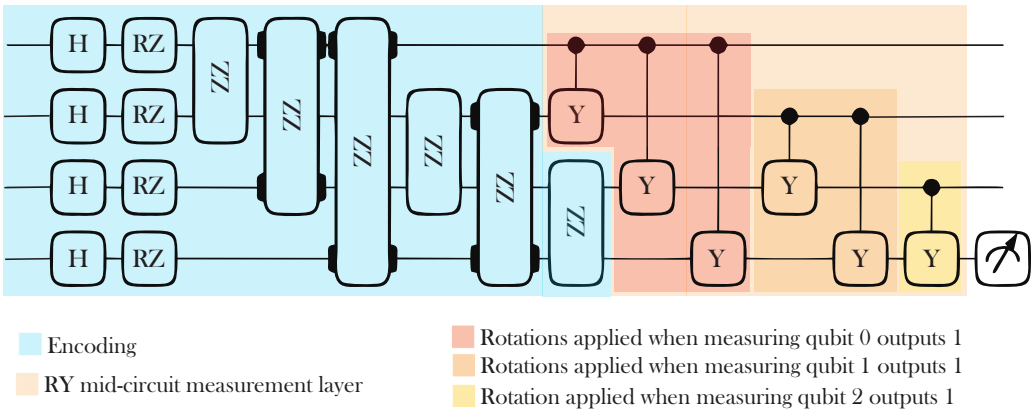


Figure 5.3: Mid-circuit measurement pooling example. This circuit consists of a higher-order encoding, followed by a R_Y mid-circuit measurement layer.

5.4.2 Ancilla qubit and controlled gates

The field of quantum machine learning has gained significant attention due to its potential for solving complex computational problems. However, models with no inductive bias often face challenges in terms of trainability and generalization. To address these issues, researchers have proposed group-invariant quantum machine learning models, which incorporate certain design principles to enhance performance and robustness [Lar+22].

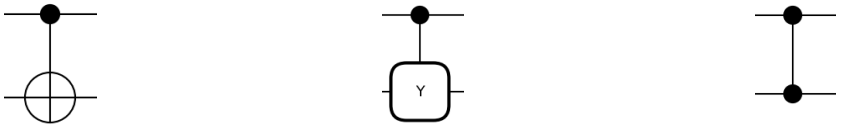
Ancilla Qubits

We employ an ancilla qubit as the target of measurement within the variational circuit layer, thereby achieving a direct reduction in input dimensionality from 5 (comprising 4 data qubits and 1 ancilla qubit) to 1 (solely the ancilla qubit).

Controlled Gates

In quantum computing, controlled gates are quantum gates that act on one qubit (called the target qubit) only when another qubit (called the control qubit) is in a particular state. In a quantum pooling layer, a sequence of controlled gates can be applied to the data qubits using the ancilla qubit as the control qubit. This allows the computation to depend on all possible states of the ancilla qubit, instead of just a single state. In our setup the controlled gates are either CNOT, CY or CZ, with

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{CY} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{pmatrix} \quad \text{CZ} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$



(a) CNOT gate representation (b) CY gate representation (c) CZ gate representation

Computation for a 4-qubits system and 4-CNOT

The system is initiated by preparing a quantum state ψ composed of four data qubits and one ancilla (control) qubit. The ancilla qubit is acted upon by the Hadamard gate H, resulting in the state ψ_1". Subsequently, a sequence of four controlled gates, specifically CNOT gates, is employed. In each case, the ancilla qubit serves as the control and one of the data qubits functions as the target. This process is repeated for qubit 1, qubit 2, and so on. Once the controlled gates have been executed, the Hadamard gate H is applied once again to the ancilla qubit. Following this, the PauliZ gate is performed on the same ancilla qubit. Finally, the expectation value of the ancilla qubit is extracted, serving as the output measurement of the experiment.

To represent this circuit mathematically, we can use the following notation:

- $|0\rangle^{\otimes 5}$ denotes the initial state of the 5 qubits, where each qubit is in the $|0\rangle$ state.
- H denotes the Hadamard gate.
- $CNOT$ denotes the controlled-X gate.
- Z denotes the Pauli-Z gate.

In our experimental setup, we initially prepare all qubits in the $|0\rangle$ state within the computational basis. However, before executing the pooling operation, we subject the qubits to data encoding, inducing non-trivial quantum states. Also some of the architectures evaluated in this thesis involve adding CNOT and rotations before the pooling step. Despite this, we can simplify the calculations by approximating all data qubits to be in the $|0\rangle$ state during pooling. Conducting computations under these simplified conditions, with CNOT serving as the control gate, can provide valuable

insights into the fundamental mechanisms that govern the pooling process.We keep the other parameters identical to the original experimental design.

Given a 5-qubit system initially in state $|00000\rangle$, let's apply the mentioned operations sequentially:

Apply Hadamard on the third qubit:

$$|00000\rangle \rightarrow (H|0\rangle) \otimes |0000\rangle = (|0\rangle + |1\rangle)/\sqrt{2} \otimes |0000\rangle = (|00000\rangle + |10000\rangle)/\sqrt{2}$$

Apply CNOT between the first and fifth qubits, knowing ancilla is the control qubit:

$$(|00000\rangle + |10000\rangle)/\sqrt{2} \rightarrow (|00000\rangle + |10001\rangle)/\sqrt{2}$$

Apply CNOT between the second and fifth qubits:

$$(|00000\rangle + |10001\rangle)/\sqrt{2} \rightarrow (|00000\rangle + |10011\rangle)/\sqrt{2}$$

Apply CNOT between the third and fifth qubits:

$$(|00000\rangle + |10011\rangle)/\sqrt{2} \rightarrow (|00000\rangle + |10111\rangle)/\sqrt{2}$$

Apply CNOT between the fourth and fifth qubits:

$$(|00000\rangle + |10111\rangle)/\sqrt{2} \rightarrow (|00000\rangle + |11111\rangle)/\sqrt{2}$$

Apply the Hadamard gate on the third qubit:

$$(|00000\rangle + |11111\rangle)/\sqrt{2} \rightarrow (H|0\rangle) \otimes |0000\rangle + (H|1\rangle) \otimes |1111\rangle$$

$$= (|0\rangle + |1\rangle)/\sqrt{2}) \otimes (|0000\rangle + \otimes(|0\rangle - |1\rangle)/\sqrt{2}) \otimes (|1111\rangle$$

$$= (|00000\rangle + |10000\rangle)/2 + (|01111\rangle - |11111\rangle)/2$$

Apply the Pauli-Z gate on the third qubit:

$$(|00000\rangle + |10000\rangle)/2 + (|01111\rangle - |11111\rangle)/2 \rightarrow (Z|0\rangle) \otimes |0000\rangle + (Z|1\rangle) \otimes |0000\rangle + (Z|0\rangle) \otimes |1111\rangle - (Z$$

$= |00000\rangle - |10000\rangle + |01111\rangle + |11111\rangle$

The final state of the 5-qubit system after a series of operations is given by the normalized linear combination $= |00000\rangle - |10000\rangle + |01111\rangle + |11111\rangle$.

To determine the probability of measuring the ancilla qubit in a particular state, we can compute the expectation value of the fifth qubit by using the projection operator onto the state where the fifth qubit is either 0 or 1. Since the coefficient of each of the four basis states that include the fifth qubit is the same, we can simplify the calculation by observing that there are two possible states with qubit 5 in the state 0, and two possible states with qubit 5 in the state 1. Hence, the probability of measuring qubit 5 in either state 0 or 1 is equal, and is 50%.

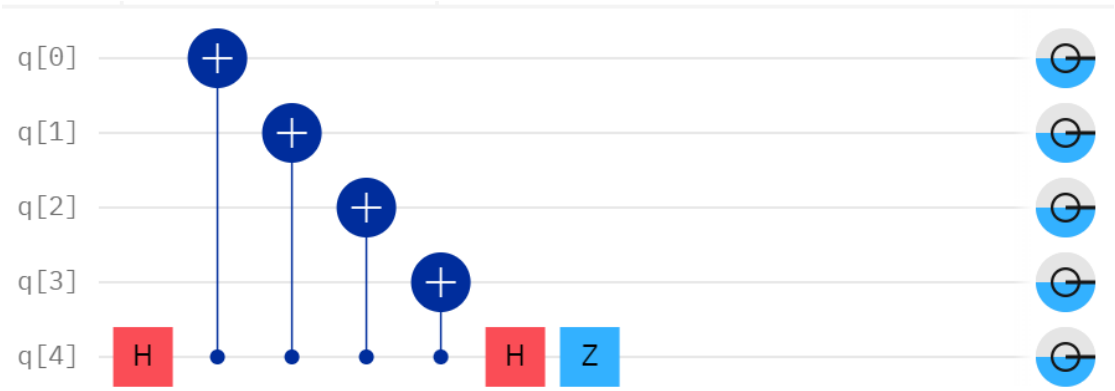


Figure 5.5: Pooling circuit described H-CNOT-H

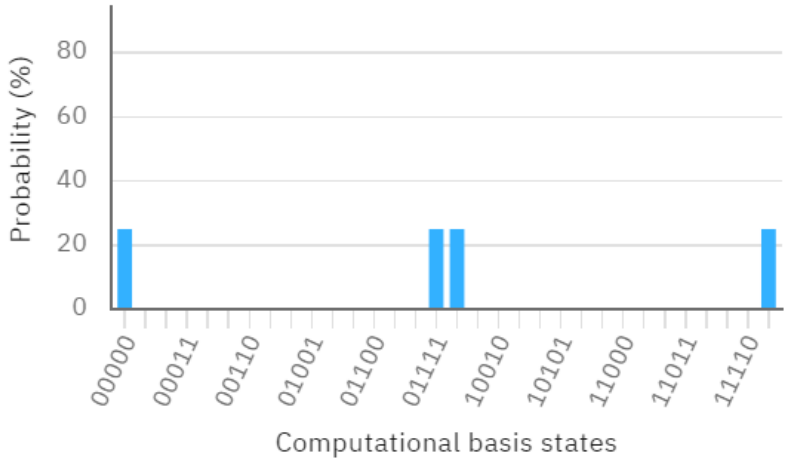


Figure 5.6: Pooling circuit computational basis

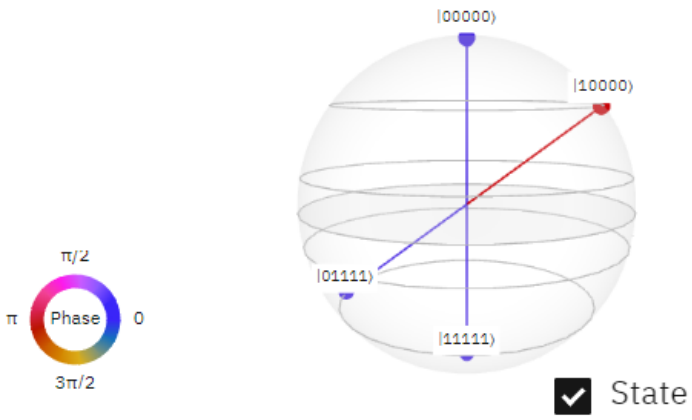


Figure 5.7: Bloch sphere

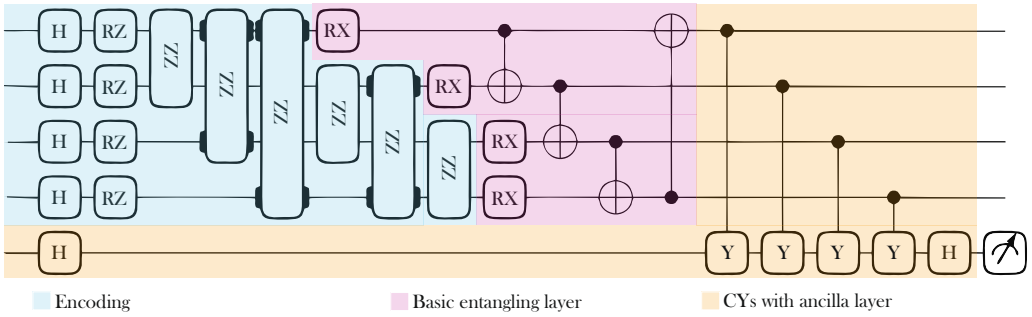


Figure 5.8: Ancilla qubit with controlled gates pooling example. This circuit consists of a higher-order encoding, followed by a basic entangling layer and CYs with ancilla qubit.

5.4.3 Qubit selection with classical postprocessing

In this experiment, we were inspired by the work of [Sch+20] in the field of classical post-processing. After encoding the data, we randomly selected one qubit from our 4-qubit system, which could either remain fixed throughout the computation or be randomly selected for each iteration. The selected qubit was then measured, and the measurement outcome was used as the output.

Additionally, we applied a post-processing step similar to an activation function, using either the sign or tanh function as our activation function. The sign function,

denoted as $Sign(x)$, is defined as follows:

$$f(x) = Sign(x) = 2H(x) - 1 = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases},$$

where $H(x)$ represents the Heaviside step function.

Moreover, the tanh function, denoted as $Tanh(x)$, is defined as:

$$Tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}.$$

By applying these activation functions, we explored whether quantum circuits benefit from non-linear activation functions in a similar manner to classical algorithms. As the output of all tested circuits falls within the range of -1 to 1, the $Sign(x)$ and $Tanh(x)$ non-linear activation functions were employed to ensure that the output of the quantum circuit remains within the same range.

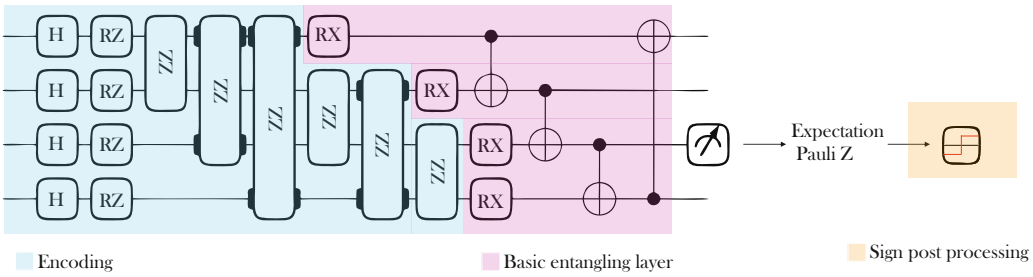


Figure 5.9: Qubit selection with classical postprocessing pooling example. This circuit consists of a higher-order encoding, followed by a basic entangling layer and a qubit selection pooling with $Sign(x)$ postprocessing

5.4.4 Modular quantum pooling blocks

To offer a thorough analysis, this thesis also examines an alternative pooling method explored by the research team, which I did not directly contribute to. By including this alternative method, valuable insight is gained into the performance of pooling methods, especially considering that one of the Mod used is considered a top performer, and impacts the conclusions drawn from the effective dimensionality analysis.

The proposed approach inspired by [HKP22] utilizes modular blocks that operate on pairs of adjacent qubits, with the utilization of either quantum pooling operations

(Mod A) or a combination of quantum convolutional circuits and pooling operations (Mod B and C). Initially, the application of two pooling blocks to the qubits results in a dimensionality reduction from four to two qubits. During the pooling process, the remaining qubits are traced out. Subsequently, the two remaining qubits undergo another round of convolutional and pooling operations, employing the same architecture as the preceding blocks, further reducing the dimensionality from four to one qubit.

The circuit architecture of this quantum pooling layer is illustrated in figures 5.10 to 5.13. The design of this pooling method incorporates modular blocks that act on pairs of neighboring qubits (the first and second qubits, and the third and fourth qubits). Each of these blocks consists of either a quantum pooling operation alone (Mod A) or a combination of a quantum convolutional circuit and a quantum pooling operation. Two different architectures are investigated for the convolutional part, with the first variant being inspired by tree tensor network ideas (Mod B), while the second circuit variant is known for its favorable entanglement properties (Mod C) [SJA19]. Consequently, the initial two pooling blocks yield a dimensionality reduction from four to two qubits, as the other qubits are traced out during the pooling operation. The remaining two qubits undergo an additional round of convolutional and pooling operations, employing the same architecture as the earlier blocks, resulting in a reduction in dimensionality from four to one qubit. The remaining qubit is measured and subsequently incorporated into the following classical parts of the network. The modular design of this circuit architecture facilitates its easy extension and adaptability to more complex scenarios.

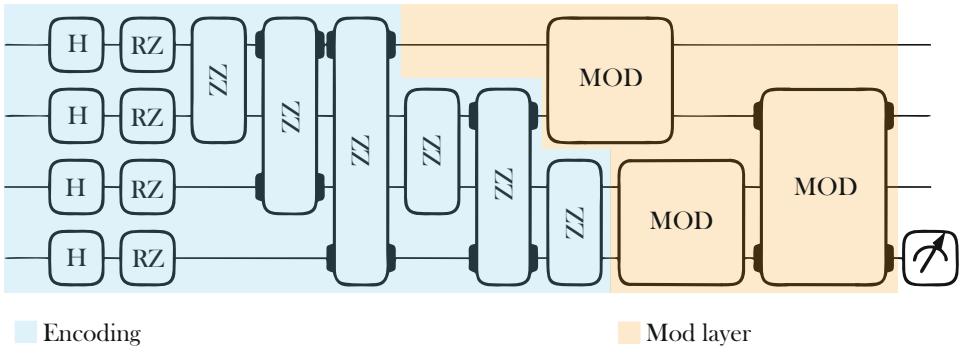


Figure 5.10: Modular quantum pooling blocks pooling example. This circuit consists of a higher-order encoding, followed by modular quantum pooling blocks.

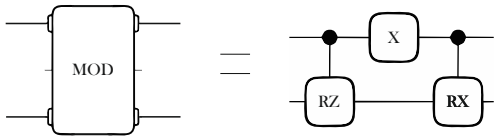


Figure 5.11: Mod-a

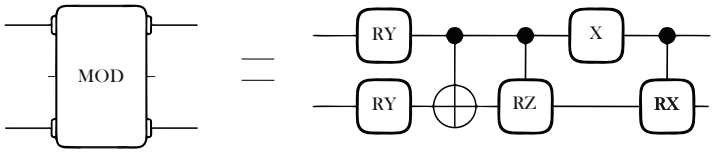


Figure 5.12: Mod-b

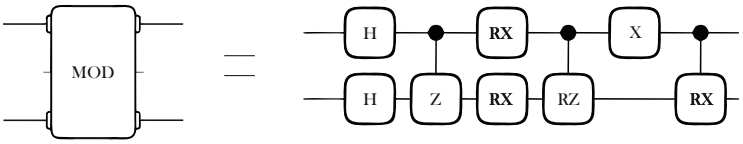


Figure 5.13: Mod-c

6 Results and quantum metrics

To ensure result integrity and minimize the impact of chance, three runs were performed per architecture, each with different initial parameters in the network. These runs were conducted for each solution and architecture type using different random seeds. The averaged results from these three runs were used to calculate uncertainty bands, representing the variation. The uncertainty bands were determined by calculating the mean minus the standard deviation and the mean plus the standard deviation. PyTorch [Pa19] was used for conducting the experiments, while Pennylane [Ber+20b] was employed for simulating the quantum circuits without considering noise effects.

For training the networks, both classical and hybrid, a fixed set of parameters was used: 20 epochs, the Adam optimizer, a learning rate of 0.001, and a batch size of 8. No extensive hyperparameter tuning was performed due to the lengthy training times of QCCNNs. The focus of the study was not on encoding paradigms; hence, only one variant, specifically higher-order encoding, was utilized.

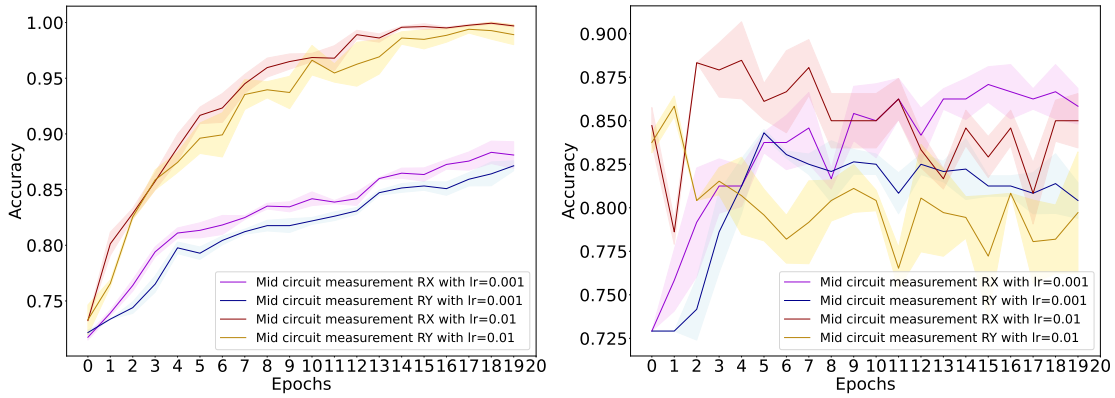
To assess the model's performance, three metrics were employed: traditional machine learning metrics such as loss and accuracy on the training dataset, and a metric commonly used in quantum computing, namely the effective dimension. Additionally, an in-depth analysis of the mid-circuit measurement method was conducted, which involved studying rotation parameters and the loss landscape.

6.1 Classical CNN and QCCNN baseline

The architectures presented in the previous work of the research team in [Mat+22] are used as the baseline. The classical CNN baseline consists of a convolutional layer with four filters of size 2×2, which is directly followed by a fully connected layer. In the QCCNN baseline, the convolutional layer is replaced by a quantum convolution operation to represent filters of size 2×2 moving over the image, as shown in figure 5.2. I did not contribute to the implementation of these baseline models, but I implemented the pooling counterparts, with the exception of the modular pooling.

6.2 Mid-circuit measurement

This section considers two hyperparameters: the type of rotation utilized (RX or RY) and the learning rate (0.001 or 0.01). This leads to a total of four possible combinations. However, our primary focus is on the learning rate of 0.001. This choice is supported by the observations depicted in Figure 6.1, where the choice of a learning rate of 0.01 leads to overfitting. In particular, it is observed that the model attains nearly 100% accuracy on the training dataset within fewer than 20 epochs. However, the validation accuracy initially reaches its highest point within the initial few epochs but subsequently deteriorates. This issue was consistent across most of the attempted pooling methods. However, the method of using mid-circuit measurements with RX rotations presented in this thesis is the only one where the model perfectly learned the training data, reaching a training accuracy of 1.0 (with a validation accuracy of at most 88.47%) within 20 epochs. Hence, unless explicitly stated otherwise, it can be assumed that the learning rate is set to 0.001.



(a) Accuracy on the training dataset

(b) Accuracy on the testing dataset

Figure 6.1: Accuracy comparison of 4 QCCNNs with mid-circuit measurement pooling depending on learning rate.

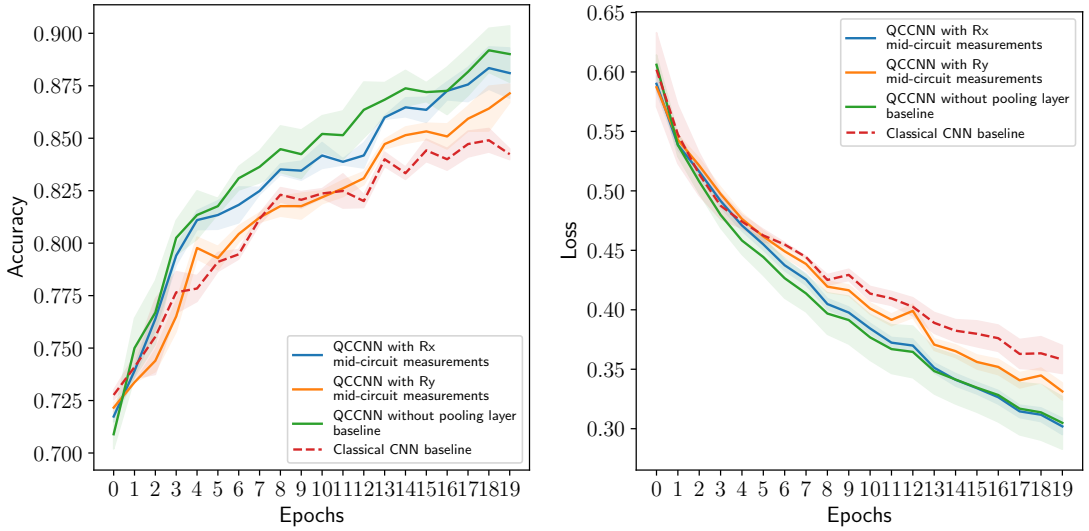
6.2.1 Accuracy and loss

In Figures 6.2 and 6.3, the training and validation accuracy and loss curves are depicted, showcasing a comprehensive comparison between the hybrid quantum-classical mid circuit measurement variants, the classical CNN baseline, and the non-pooling QCCNN baseline. Notably, the learning accuracy of the hybrid quantum-classical variants consistently surpasses that of the classical model on the training dataset. Furthermore, in terms of learning loss, despite the initial lower loss exhibited by the QCCNN baseline

in the first half of the learning process, comparable loss is achieved by the QCCNN with RX mid circuit measurement method starting epoch 12. This QCCNN with RX mid circuit measurement model also shows a narrower uncertainty band, indicating reduced sensitivity to the choice of random seed. This characteristic could indicate a superior stability and reliability.

When assessing generalization on the validation set, the architecture utilizing R_X measurements of the intermediate circuit outperforms the architecture utilizing R_Y measurements of the intermediate circuit quite obviously. Moreover, the former surpasses both baselines in terms of maximum validation accuracy. Encouragingly, this performance improvement is sustained until the end of the training, reflecting efficient progress in the training process. This observation is further supported by the validation loss of that model, which falls below that of the other methods after episode 10.

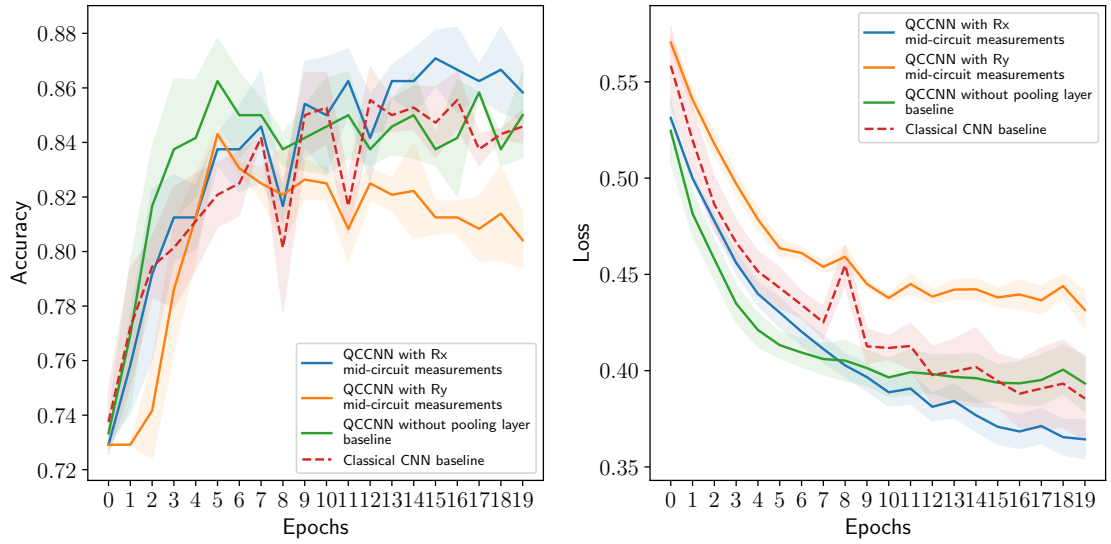
While the QCCNN with RX mid circuit measurement model appears to be the most effective approach, it is important to acknowledge that the overlapping variance bands with the other methods prevent definitive conclusions regarding its superiority.



(a) Accuracy

(b) Loss

Figure 6.2: Performance comparison on the training dataset of 4 models: 2 QCCNNs with mid-circuit measurement pooling, a basic QCNN and a classical CNN.



(a) Accuracy

(b) Loss

Figure 6.3: Performance comparison on the validation dataset of 4 models: 2 QCCNNs with mid-circuit measurement pooling, a basic QCNN and a classical CNN.

6.2.2 Quantum weights

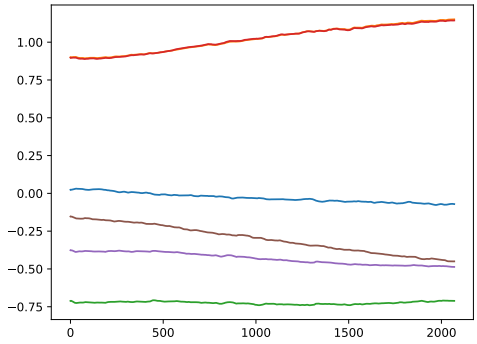
The rotation angles of RX gates and RY gates trained in a mid-circuit measurement pooling circuit for a QCCCNs model are depicted in figures 6.4 and 6.5 respectively. A total of six rotation gates can be trained in this type of VQC, resulting in the representation of six angles. These angles should fall within the range of -$\pi/2$ to +$\pi/2$.

When using a learning rate of 10-3, the angles of the six rotation gates do not converge significantly. Among the six weights, two maintain their values throughout all epochs, while the remaining weights exhibit minor fluctuations without displaying clear convergence.

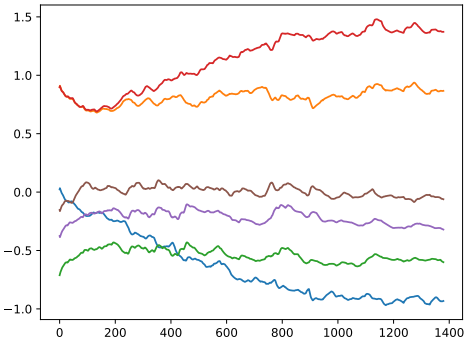
When using a learning rate of 10-2, the updates to the angles in the training process can be described as relatively abrupt. Nevertheless, it should be noted that the movement in the unsuccessful pooling technique is even more pronounced, highlighting a general learning rate that is too high.

However, it is crucial to consider the shape of the loss landscape before drawing definitive conclusions. In the event that the loss landscape is flat, it would be reasonable to observe significant differences in angle values. This behavior can be attributed to a good model's tendency to explore longer distances in an effort to escape local minima. Therefore, a comprehensive evaluation of these results should take into account the

characteristics of the loss landscape.

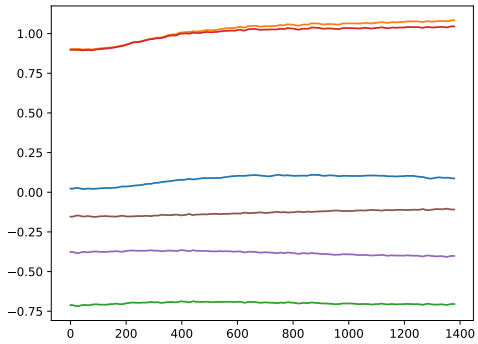


(a) Learning rate 0.001

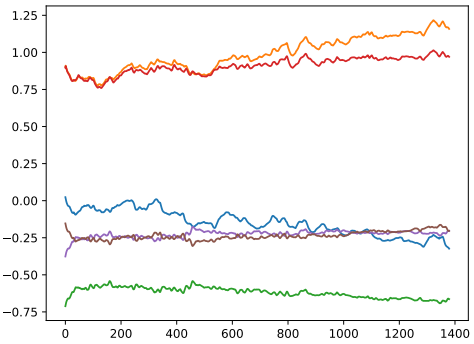


(b) Learning rate 0.01

Figure 6.4: Angle update for mid circuit measurement with RX

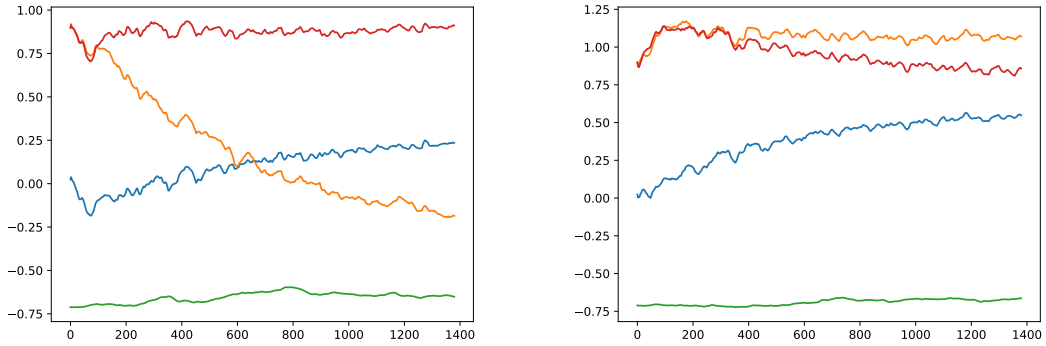


(a) Learning rate 0.001



(b) Learning rate 0.01

Figure 6.5: Angle update for mid circuit measurement with RY



(a) CY (b) CZ

Figure 6.6: Angle update for ancilla qubit with 3 inversed controlled gates and learning rate 0.01

6.2.3 Loss landscape

The loss landscape was visualized on a three-dimensional graph, with the z-axis representing the loss value. The aim was to determine the most appropriate approach by considering two options based on the parameters of the x and y axes.

The first option was to use Principal Component Analysis (PCA) to perform dimensionality reduction. The aim was to reduce the feature space from 6 to 2 dimensions. However, the axes resulting from PCA were not very interpretable, making it difficult to obtain meaningful explanations from the plot.

To improve interpretability, the second approach was chosen. In this approach, two carefully selected angles were used as x and y axes. Specifically, two sets of angles were selected: 0 and 5, and 3 and 4. The aim of using these angles was to highlight differences in the loss landscape for angles that depend on the same measurement and for those that represent the most different dependence. Indeed, Angles 3 and 4 corresponded to the rotation gates affected by the measurement of the second qubit. Angle 0 represented a rotation gate impacted by the measurement of the first qubit, while angle 5 represented the only rotation gate reliant on the third qubit measurement.

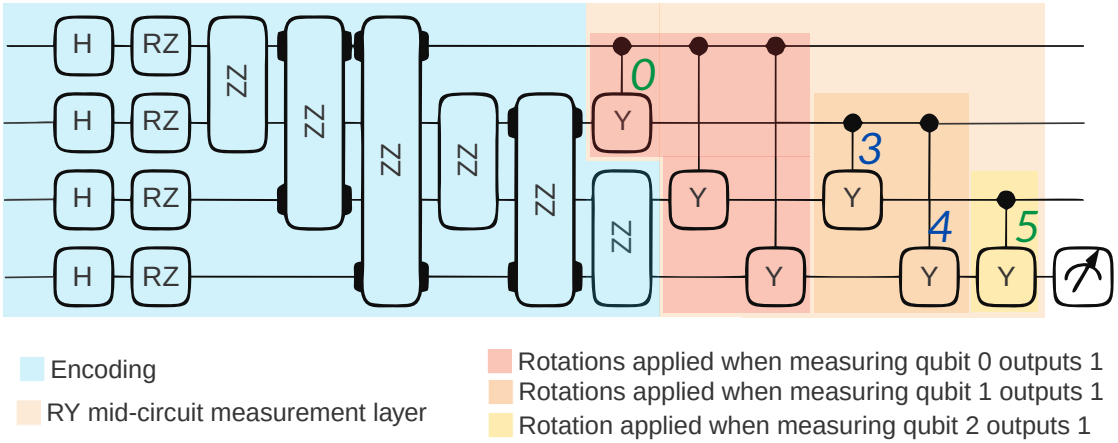


Figure 6.7: Mid-circuit measurement pooling example with rotations numbered.

The analysis of the loss landscape involves the utilization of angle values denoted as alpha and beta, which are used along the x and y axes, respectively. These angles are obtained throughout the training process. The mean and standard deviation of these angles are calculated by considering the 20 values over 20 training epochs. The examination of the loss with respect to variations in angle values around two specific points, namely the angle at epoch 0 (representing the angle at the beginning of training, i.e., the cold start) and the angle at epoch 20 (following training), is a part of our approach. The range of angles within three standard deviations from these points is explored, resulting in a total of 100 loss points.

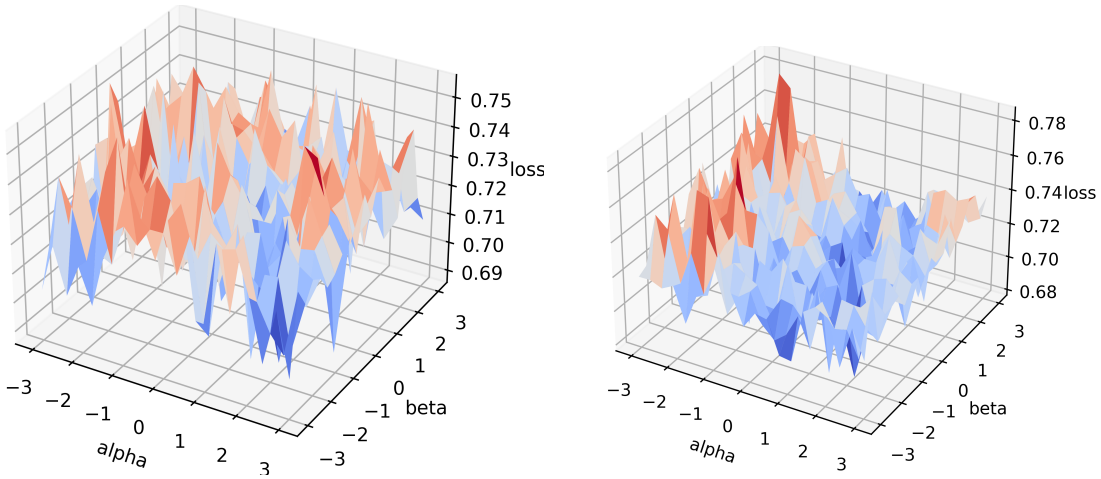
During the cold start, it is observed that the loss does not appear to depend significantly on the quantum parameters before training the overall model. This lack of dependence can be attributed to the relatively smaller number of quantum parameters (6) compared to the classical parameters (a few hundred). As a result, even when the quantum parameters are varied, the influence of the randomness in other classical parameters impedes the achievement of a desirable loss.

In contrast, after training, the angles of the pooling layers exhibit a significant impact on the loss. Just a single adjustment of an angle can cause the loss to either decrease or increase by a factor of 5. It is worth noting that certain angles have a greater importance than others. For instance, as depicted in Figure 1, angle 0 has a more pronounced effect on the loss landscape compared to angle 5.

Furthermore, this analysis provides deeper insights into the behavior of the weights. When using a learning rate of 10^-3, the weights do not undergo substantial changes and do not seem to converge. Multiple local minima exist in close proximity, often with similar values. The proximity of these minima facilitates transitions between

them, as the values around the final angle fall within a similar range, even for local maxima. A sharp increase in loss is only observed when moving beyond one standard deviation from the final angle. On the other hand, when using a learning rate of 10^2, as depicted in Figure 1, the analysis reveals that the model is situated within a relatively flat minimum surface. Even when the learning rate is increased by one standard deviation, similar values are obtained. This observation may also explain the non-convergence of the angles.

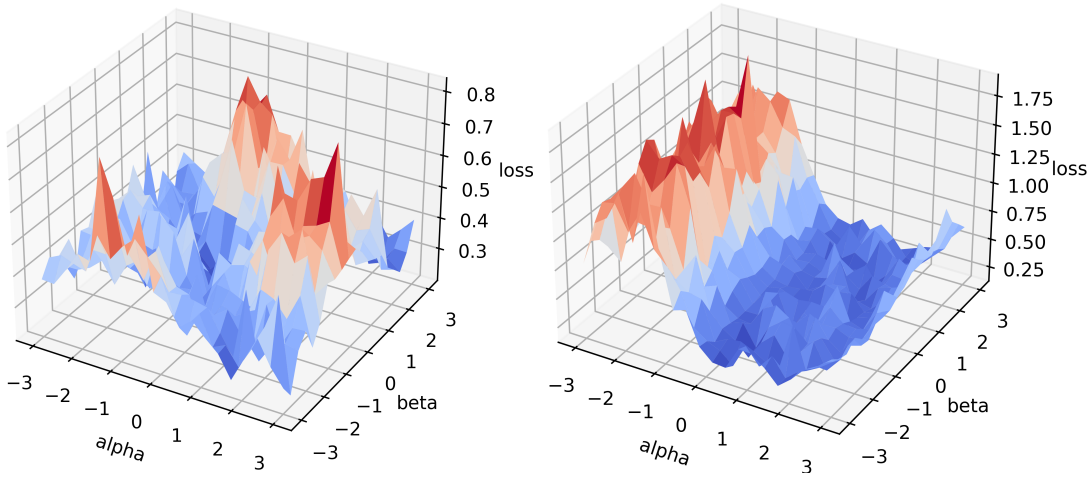
Consequently, the loss landscape analysis reveals that the quantum parameters have limited influence during the initial stages of training due to the overwhelming number of classical parameters. However, after training, the angles of the pooling layers play a crucial role in determining the loss. Additionally, the presence of multiple local minima in close proximity hinders the convergence of these angles.



(a) Angle 3, 4

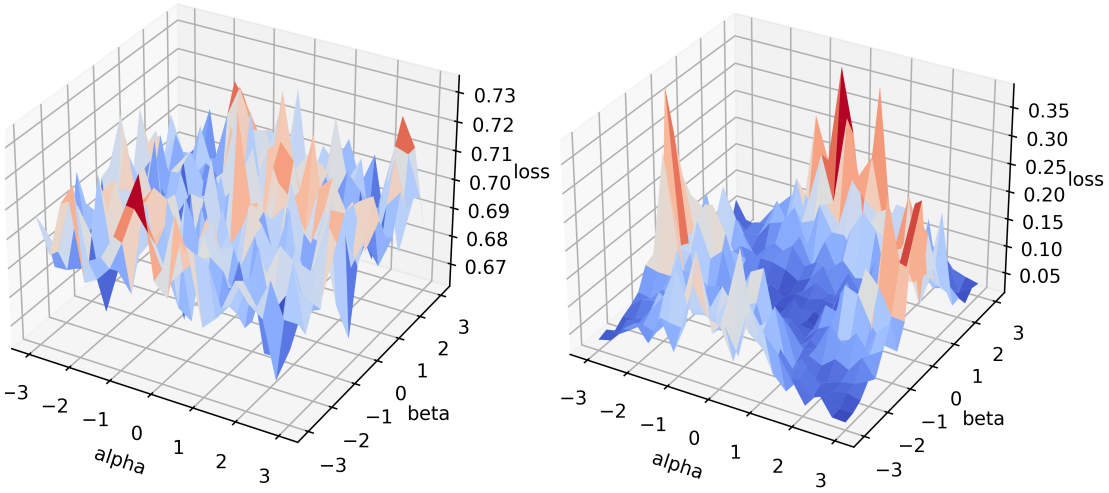
(b) Angle 0, 5

Figure 6.8: Visualization of the loss landscape of mid-circuit measurement with RY gates, cold start, and a learning rate of 0.001 for angles alpha and beta.



(a) Angle 3, 4 (b) Angle 0, 5

Figure 6.9: Visualization of the loss landscape of mid-circuit measurement with RY gates, following training and a learning rate of 0.001 for angles alpha and beta.



(a) Cold start (b) Following training

Figure 6.10: Visualization of the loss landscape of mid-circuit measurement with RX gates, and a learning rate of 0.01 for angles 3 and 4.

6.2.4 Effective dimension

The effective dimension is associated with three significant implications that contribute to our understanding of a model's performance.

Firstly, a measure of the model's information capacity is provided, potentially enhancing confidence in its ability to generalize on other datasets. When a combination of high effective dimension, high accuracy, and low loss is observed, it may suggest that these results are a consequence of the model architecture rather than simply coincidental compatibility with the dataset. However, achieving the same accuracy and loss with a low effective dimension might reduce the significance of the model architecture in contributing to these outcomes.

Secondly, the correlation between accuracy and effective dimension, or the presence of low accuracy with low effective dimension, could be used as a potential screening tool. By efficiently calculating the effective dimension, which depends solely on the architecture and not on training, it is possible to make a preliminary assessment of whether it is worthwhile to invest extensive computational resources and time in training a model. This approach may help in prioritizing runs more effectively.

Lastly, an important aspect is the ability to explore why a specific architecture is performing well or not. By seeking justifications for the observed performance, researchers can gain valuable insights into the underlying mechanisms and make informed decisions for further improving the model.

Note that the effective dimension being examined is for a variational quantum circuit (VQC) and not a complete layer. A pooling layer consists of four VQCs of a specific type. Since pooling layers typically consist of 4 to 12 angles and the basic entangling layer has 4 angles, a normalized effective dimension is employed that considers the number of parameters (angles).

In our analysis, we find that the Basic entangling layer without pooling has a normalized effective dimension of approximately 0.933, compared to 0.909 for mid-circuit measurement. However, it would be incorrect to conclude that the basic entangling layer is superior in regards to that metric based solely on these values. The effective dimension accounts for the capacity of a single circuit, and although the basic entangling layer has a higher one, it outputs four measurements without tracing out any qubits, meanwhile mid circuit circuit trace out 3 out of 4 qubits. In addition, the normalization penalizes mid-circuit measurement, which has six parameters compared to the four of the entangling. Hence, it is rather intriguing and in a positive manner that a single mid-circuit VQC demonstrates comparable complexity to the basic entangling layer, despite the inclusion of four mid-circuit VQCs in the QCCNN model as opposed to just one basic entangling VQC.

Table 6.1: Effective dimension (ED) and highest training/validation accuracy of QCCNN models with mid-circuit measurement pooling and the QCCNN baseline.

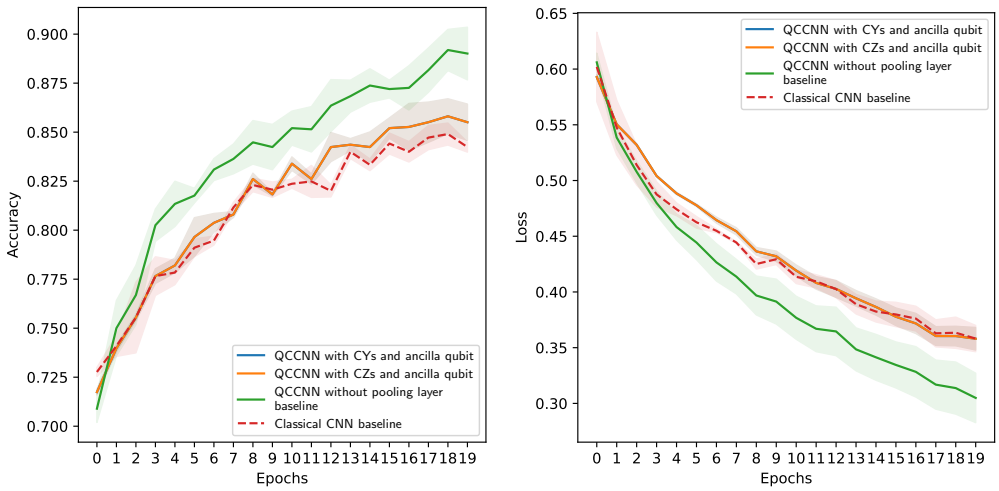
VQC	Normalized ED	Max train acc	Max val acc
R_X mid-circuit measurements	0.909 ± 0.016	92.81 ± 0.37	87.08 ± 1.02
R_Y mid-circuit measurements	0.906 ± 0.008	87.14 ± 0.44	84.31 ± 0.20
Basic entangling layer no pooling	0.933 ± 0.019	89.19 ± 1.05	86.25 ± 1.56

6.3 Ancilla qubit and controlled gates

6.3.1 Accuracy and loss

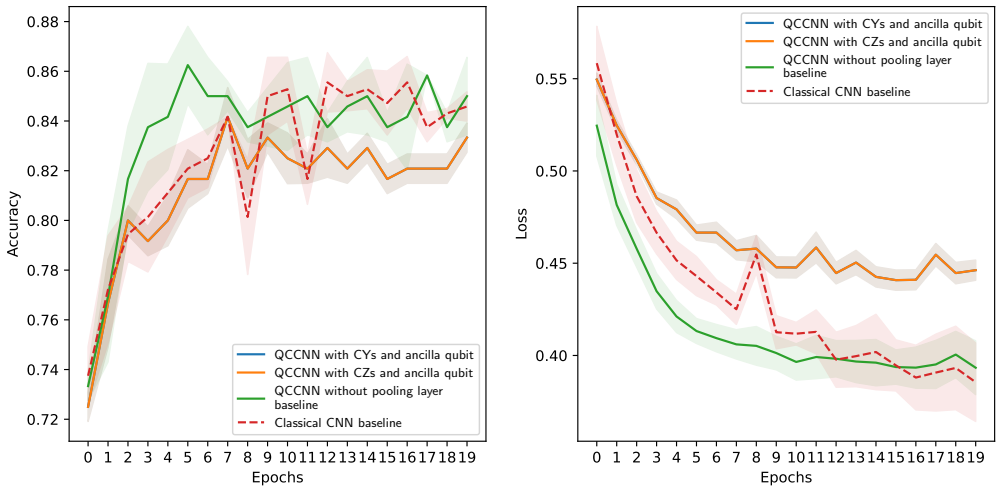
Figures 6.11 and 6.12 present the training and validation accuracy and loss curves, showcasing a comprehensive comparison between the hybrid quantum-classical ancilla qubit and controlled gates models, the classical CNN baseline, and the non-pooling QCCNN baseline. The results suggest that the overall performance of the QCCNN does not appear to be significantly affected by the inclusion of CY or CZ controlled gates, as indicated by the similarity in the accuracy and loss curves for both variants.

It is observed that the employed ancilla qubit with controlled gates pooling method demonstrates satisfactory training performance. This method achieves a maximum accuracy slightly higher than that of the classical CNN and maintains a similar loss to it during the latter half of the training. However, when examining the validation accuracy, this method does not perform as well as both the classical CNN and the QCCNN without pooling baselines, as its maximum validation accuracy is lower than that of the aforementioned baselines.



(a) Accuracy (b) Loss

Figure 6.11: Performance comparison on the training dataset of 4 models: 2 QCCNNs with ancilla qubit and controlled gates pooling, a basic QCNN and a classical CNN.



(a) Accuracy (b) Loss

Figure 6.12: Performance comparison on the validation dataset of 4 models: 2 QCCNNs with ancilla qubit and controlled gates pooling, a basic QCNN and a classical CNN.

6.3.2 Effective dimension

Drawing a conclusive assessment of the effectiveness of the controlled gate and ancilla qubit method based on the effective dimension is a challenging task. Although the results obtained from both CY and CZ Variational Quantum Circuits (VQCs) were nearly identical, there is a noticeable difference in the effective dimensions of the CY and CZ gates.

Furthermore, the VQCs that utilize this pooling method employ four parameters, which is the same as the number of parameters in the basic entangling layer. Hence, the reduction in effective dimension from 0.933 to 0.801 or 0.772 can be solely attributed to the tracing out of four out of five qubits in the controlled gates method and the choice of VQC architecture. Therefore, despite this decrease, the effective dimension is not sufficiently low to explain the predominantly subpar results of the QCCNN when utilizing this pooling method.

Table 6.2: Effective dimension (ED) and highest training/validation accuracy of QCCNN models with ancilla qubit and controlled gates pooling and the QCCNN baseline.

VQC	Normalized ED	Max train acc	Max val acc
Ancilla qubit with CY gates	0.772 ± 0.041	85.81 ± 0.89	84.17 ± 1.18
Ancilla qubit with CZ gates	0.801 ± 0.042	85.81 ± 0.89	84.17 ± 1.18
Basic entangling layer no pooling	0.933 ± 0.019	89.19 ± 1.05	86.25 ± 1.56

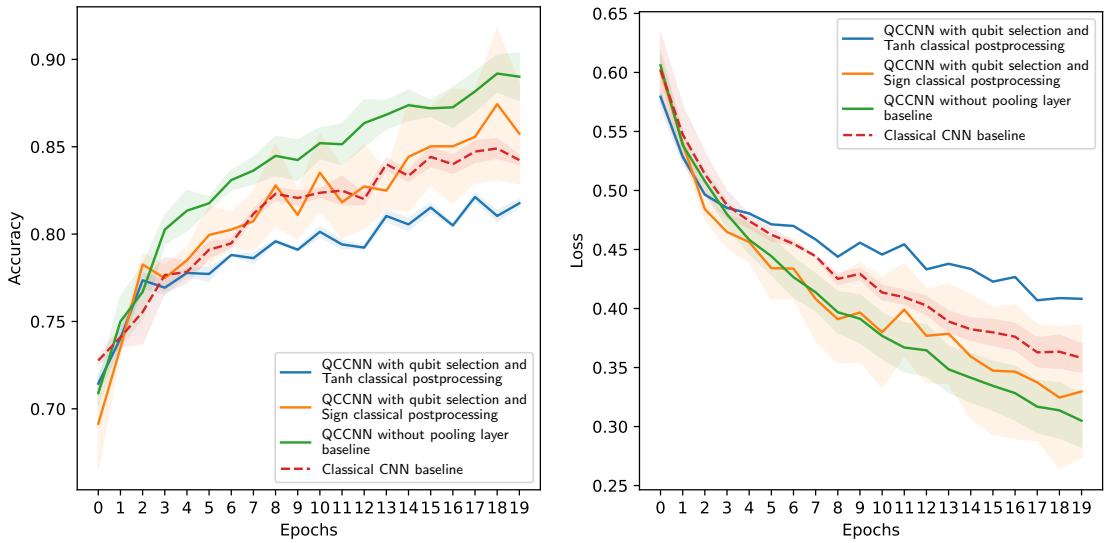
6.4 Qubit selection with classical postprocessing

6.4.1 Accuracy and loss

Figures 6.13 and 6.14 depict the training and validation accuracy and loss curves, allowing for a comprehensive comparison among the hybrid quantum-classical qubit selection with classical postprocessing models, the classical CNN baseline, and the non-pooling QCCNN baseline. The results suggest that the non-pooling QCCNN performs better in terms of both accuracy and loss on the training dataset.

However, it is worth noting that the architecture utilizing the $Tanh(x)$ activation function exhibits promising generalization and fast learning on the validation data. It tends to outperform both the classical baseline and the non-pooling QCCNN, on average, in terms of validation accuracy starting from the third epoch, and achieve the highest accuracy on that epoch. Although this architecture consistently performs well, the variability in validation accuracy makes it challenging to conclusively determine its

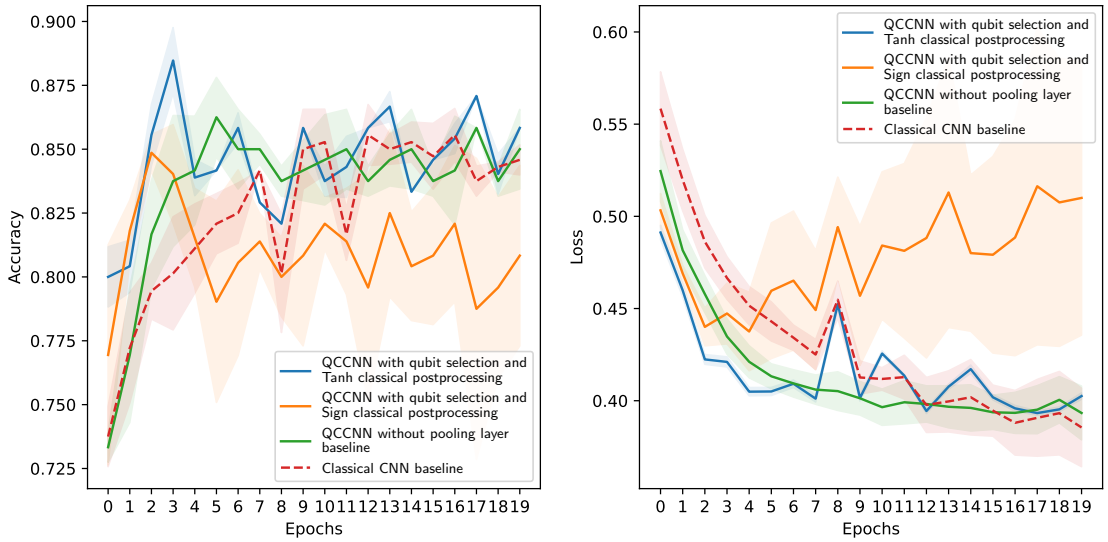
superiority over the non-pooling QCCNN. Conversely, the architecture employing the $Sign(x)$ activation function struggles to generalize effectively on the validation data, showing notable variance bands. The inconsistency is particularly evident in the loss, where only one of the three runs achieves satisfactory results. This behavior may be attributed to potential information loss associated with the application of the $Sign(x)$ function.



(a) Accuracy

(b) Loss

Figure 6.13: Performance comparison of 4 models on the training dataset: 2 QCCNNs with qubit selection with classical postprocessing pooling, a basic QCNN and a classical CNN.



(a) Accuracy

(b) Loss

Figure 6.14: Performance comparison of 4 models on the validation dataset: 2 QCCNNs with qubit selection with classical postprocessing pooling, a basic QCNN and a classical CNN.

6.4.2 Effective dimension

The primary focus of this pooling method lies in the type of classical post-processing applied. However, it should be noted that a decrease in effective dimension from 0.933 to 0.666 is observed when the effective dimension of the Variational Quantum Classifier (VQC) is calculated before classical postprocessing. This decrease occurs specifically when three out of four qubits are traced out, and only the third qubit (q2) is retained.

Table 6.3: Effective dimension (ED) and highest training/validation accuracy of QCCNN models with qubit selection for classical postprocessing pooling and the QCCNN baseline.

VQC	Normalized ED	Max train acc	Max val acc
Qubit selection	0.666 ± 0.048	82.13 ± 0.23	88.47 ± 1.29
Basic entangling layer no pooling	0.933 ± 0.019	89.19 ± 1.05	86.25 ± 1.56

6.5 Modular quantum pooling blocks

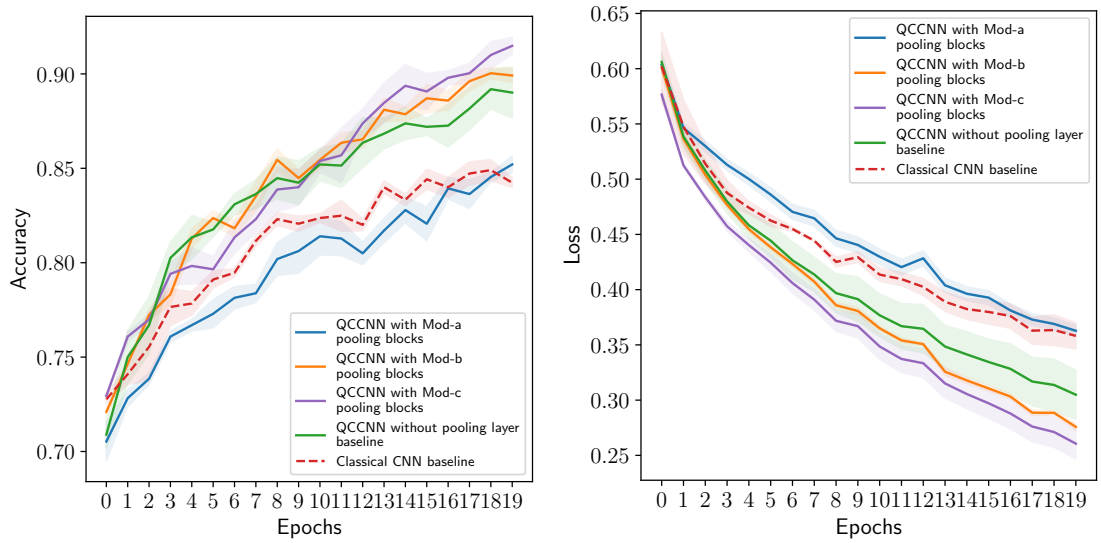
6.5.1 Accuracy and loss

Figures 6.15 and 6.16 depict the training and validation accuracy and loss curves, allowing for a comprehensive comparison among the hybrid quantum-classical models with modular pooling blocks, the classical CNN baseline, and the non-pooling QCCNN baseline.

The performance in terms of validation accuracy shows promise for the modular pooling methods. Out of the three methods considered (Mod-a, Mod-b, and Mod-c), the classical baseline is outperformed by Mod-a and Mod-c. This suggests that the incorporation of modular pooling has the potential to enhance accuracy in classifying breast ultrasound images. Furthermore, all three methods end up achieving higher training accuracy than the classical baseline.

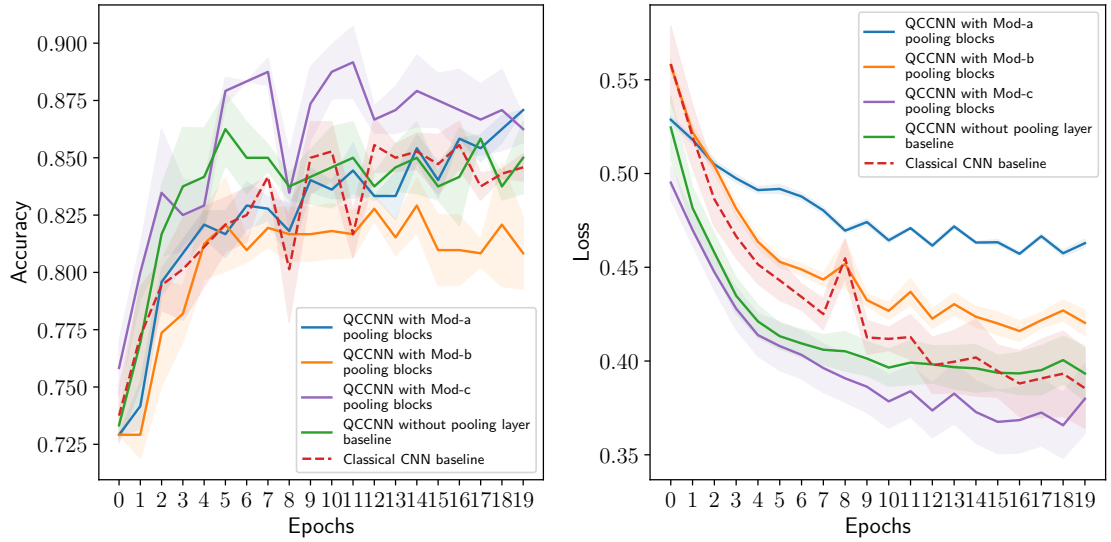
Notably, Mod-c stands out for its substantial performance gain compared to the classical baseline. It exhibits rapid and steep convergence early in the training process and demonstrates clear and distinct error bands that do not overlap with other models. This suggests a more confident and stable learning process for Mod-c. However, the comparison based on loss is inconclusive due to the presence of overlapping error bands, making it difficult to declare Mod-c as superior.

Conversely, Mod-a consistently exhibits the highest loss for both the training and validation datasets. This indicates higher levels of error despite achieving higher accuracy in validation. On the other hand, Mod-b demonstrates lower loss during training compared to both the classical CNN and QCCNN without pooling. However, on the validation dataset, Mod-b tends to exhibit higher loss compared to the baselines.



(a) Accuracy (b) Loss

Figure 6.15: Performance comparison of 5 models on the training dataset: 3 QCCNNs with modular quantum pooling blocks, a basic QCNN and a classical CNN.



(a) Accuracy (b) Loss

Figure 6.16: Performance comparison of 5 models on the validation dataset: 3 QCCNNs with modular quantum pooling blocks, a basic QCNN and a classical CNN.

6.5.2 Effective dimension

The experimental setup involved three different methods: Mod A, Mod B, and Mod C.

Mod A uses 2*3 angles, resulting in a total of 6 parameters. In contrast, both Mod B and Mod C use 4*3 angles, leading to 12 parameters. The experimental results showed significant variations among these three methods.

When comparing Mod A pooling to the basic entangling method, it was observed that Mod A exhibited a lower normalized effective dimension of 0.726, while the basic entangling method had a value of 0.933. However, drawing definitive conclusions from this comparison proved inconclusive due to the fact that modular pooling traced out 3 out of 4 qubits and had a different parameter count of 6, in contrast to the basic entangling method's 4 parameters.

Regarding Mod B and Mod C, both methods showed low effective dimensions. Mod B had an effective dimension of 0.149, while Mod C had a value of 0.249. Both methods shared similar conditions, including 12 parameters and 3 traced-out qubits. Interestingly, the higher effective dimension observed in Mod C suggested a potential correlation between effective dimension and accuracy. This was supported by the fact that Mod C demonstrated notably better results compared to Mod B. However, it is noteworthy that Mod C exhibited a significant reduction in effective dimension compared to the basic entangling method, making it challenging to establish a clear correlation between effective dimension and accuracy since Mod C had better results than using the basic entangling method.

Table 6.4: Effective dimension (ED) and highest training/validation accuracy of QCCNN models with modular quantum pooling and the QCCNN baseline.

VQC	Normalized ED	Max train acc	Max val acc
Modular pooling Mod-a	0.726 ± 0.006	85.21 ± 0.45	87.08 ± 0.00
Modular pooling Mod-b	0.149 ± 0.003	90.04 ± 0.30	82.92 ± 1.18
Modular pooling Mod-c	0.249 ± 0.016	91.49 ± 0.44	89.17 ± 1.56
Basic entangling layer no pooling	0.933 ± 0.019	89.19 ± 1.05	86.25 ± 1.56

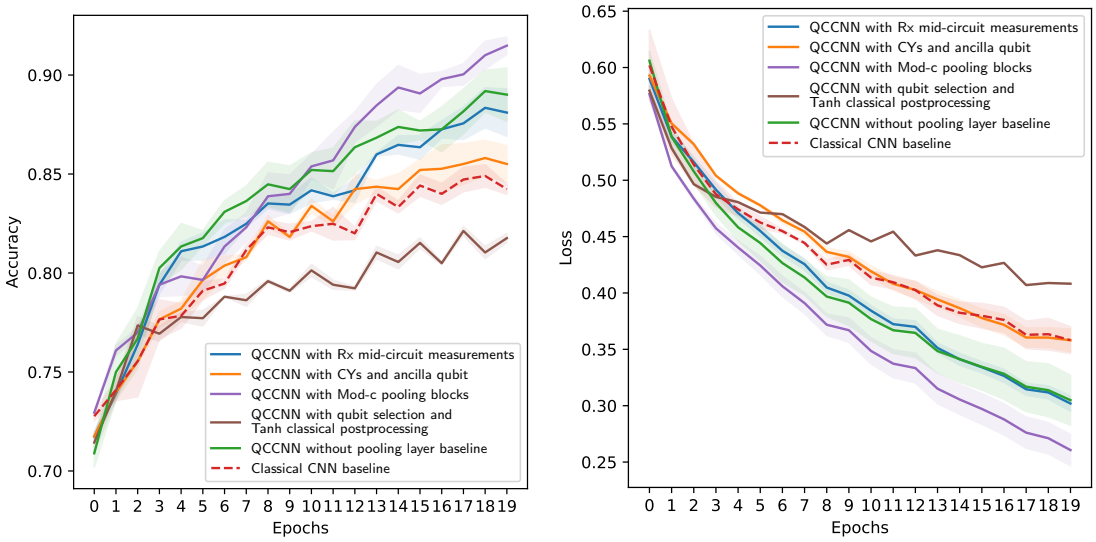
6.6 Comparison of the best models

6.6.1 Accuracy and loss

Figures 6.17 and 6.18 presents the performance comparison of the best performing configuration for each pooling option.

The QCCNN with RX mid-circuit measurement consistently outperforms the classical CNN baseline in terms of training accuracy. It achieves similar loss values while demonstrating improved stability and reliability. Among the modular pooling methods, Mod-c shows significant performance improvement. It converges quickly and exhibits distinct error bands. Additionally, the architecture that utilizes the Tanh(x) activation function shows promising generalization and fast learning. The inclusion of ancillary qubit and CY gates contributes to satisfactory training performance.

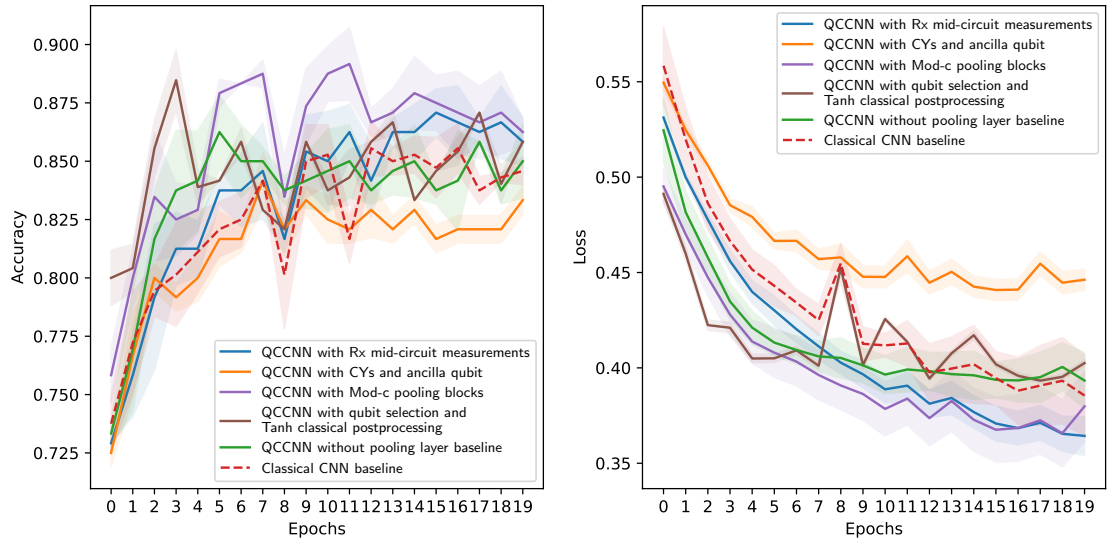
In general, pooling architectures consistently achieve better results than the traditional CNN baseline in terms of learning and validation accuracy. QCCNNs also show faster convergence in terms of validation accuracy. Among the different pooling methods, the QCCNN using Mod-c modular pooling achieves the highest levels of learning and validation accuracy and the lowest loss. It is followed by the QCCNN incorporating mid-circuit RX measurements, and finally, the QCCNN using qubit selection and classical Tanh(x) post-processing. However, the QCCNN employing ancillary qubit pooling performs less effectively than the basic QCCNN without pooling.



(a) Accuracy

(b) Loss

Figure 6.17: Performance comparison of the best performing pooling models on the training dataset with a basic QCNN and a classical CNN.



(a) Accuracy

(b) Loss

Figure 6.18: Performance comparison of the best performing pooling models on the validation dataset with a basic QCNN and a classical CNN.

6.6.2 Effective dimension

The master's thesis identified the best performing pooling method from each category. The ED, which measures the model's information capacity, is expected to have a positive correlation with training accuracy. However, it is not possible to draw a definitive conclusion within the scope of this thesis. Mid-circuit measurements exhibit a relatively high normalized ED and perform well in terms of training accuracy. Interestingly, Mod-c, despite having one of the lowest normalized ED values, is actually the best performing variant. Controlled gates with ancillary qubits show an average ED, even though they were among the least successful pooling methods. Excluding modular pooling options, a positive correlation between ED and VQC performance was observed. This suggests that further investigation is needed to explore this relationship. For future research, it would be valuable to explore alternative normalization methods that are less stringent, one possible approach could involve dividing by the square root of the number of parameters, instead of the absolute number, and statistically evaluating its suitability.

Table 6.5: Effective dimension (ED) and highest training/validation accuracy of of all the best performing QCCNN models with pooling and the QCCNN baseline.

VQC	Normalized ED	Max train acc	Max val acc
R_X mid-circuit measurements	0.909 ± 0.016	92.81 ± 0.37	87.08 ± 1.02
Ancilla qubit with CY gates	0.772 ± 0.041	85.81 ± 0.89	84.17 ± 1.18
Modular pooling Mod-c	0.249 ± 0.016	91.49 ± 0.44	89.17 ± 1.56
Qubit selection	0.666 ± 0.048	82.13 ± 0.23	88.47 ± 1.29
Basic entangling layer no pooling	0.933 ± 0.019	89.19 ± 1.05	86.25 ± 1.56

7 Conclusion and further work

This master's thesis delved into the investigation of various quantum-classical pooling techniques. Specifically, the thesis explored the application of mid-circuit measurements, ancilla qubits with controlled gates, qubit selection with classical postprocessing, and modular pooling for hybrid Quantum-Classical Convolutional Neural Networks (QCCNNs). The aim was to replace the classical convolutional layer in a traditional CNN with a quantum convolution and pooling layer. The use of QCCNNs shows promise, especially in scenarios where training data is limited, which is often the case in the classification of medical images. The evaluation of these techniques was conducted on a small dataset of breast ultrasound images, with the objective of classifying lesions as either benign or malignant.

Among the pooling techniques, the modular pooling approach exhibited the best performance on the selected dataset. The mid-circuit measurements pooling showed promise in terms of training ability, while the qubit selection with classical postprocessing demonstrated excellent generalization ability. However, no pooling architecture stood out as significantly superior to the others in terms of classification performance.

The research was constrained by an accuracy limitation in the selected dataset, which is also observed in traditional cutting-edge architectures for that specific application. Nevertheless, certain hybrid approaches demonstrated similar or slightly enhanced accuracy while requiring fewer training iterations and trainable parameters. This faster learning potential could be particularly beneficial for tasks prioritizing efficiency in the short term, given that only 4 logical qubits are required.

To evaluate the effectiveness of the proposed techniques, a noteworthy measure called the effective dimension was utilized. The goal was to examine the relationship between the highest achieved effective dimension and the validation accuracy. However, a strong correlation between the two was not established. Additionally, the pooling process employed in the circuit led to differences in dimensions between the input and output, making it impossible to calculate other important measures such as expressivity and entanglement due to the tracing out of qubits.

Moving forward, future research should consider several avenues. Firstly, testing the proposed architectures on diverse datasets with varying precision levels beyond BreastMNIST would provide valuable insights into the potential quantum advantages across different domains. Secondly, exploring alternative quantum indicators that

can predict accuracy before training would be intriguing. While effective dimension was considered in this study, a comprehensive understanding of quantum circuits' capabilities would require investigating measures such as expressivity and entanglement. Lastly, conducting experiments on quantum hardware is crucial to account for real-world noise and size limitations inherent in practical implementations. This would offer valuable insights into the feasibility and performance of the proposed pooling techniques, bridging the gap between simulation-based findings and actual quantum hardware.

List of Figures

List of Tables

Bibliography

[] "Quantum Computer Simulates Largest Molecule Yet, Sparking Hope for Future Drug Discoveries." In: *Science* ().

[] "Quantum Simulation of a 45-Atom Molecule." In: *APS Physics* 15 (175).

[23] *IBM Quantum Computing Challenge - Spring 2023*. `https://challenges.quantum-computing.ibm.com/spring-2023`. Accessed: May 29, 2023. 2023.

[Aa20] W. Al-Dhabyani and et al. "Dataset of breast ultrasound images." In: *Data in Brief* 28 (2020), p. 104863. DOI: `10.1016/j.dib.2019.104863`.

[Abb+21] A. Abbas, S. Alexander, M. Cerezo, L. Cincio, P. J. Coles, A. Datta, A. J. Gray, T. H. Hsieh, I. Kerenidis, and N. M. Linke. "The power of quantum neural networks." In: *Nature Communications* 12.1 (2021), pp. 1–10.

[AG17] N. Aloysius and M. Geetha. "A review on deep convolutional neural networks." In: *2017 international conference on communication and signal processing (ICCSP)*. IEEE. 2017, pp. 0588–0592.

[AK22] A. Amirkhani and M. P. Karimi. "Adversarial defenses for object detectors based on Gabor convolutional layers." In: *The Visual Computer* 38.6 (2022), pp. 1929–1944.

[Anw+18] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami, and M. K. Khan. "Medical image analysis using convolutional neural networks: a review." In: *Journal of medical systems* 42 (2018), pp. 1–13.

[Ber+18] 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." In: *arXiv preprint arXiv:1811.04968* (2018).

[Ber+20a] O. Berezniuk, A. Figalli, R. Ghigliazza, and K. Musaelian. "A scale-dependent notion of effective dimension." In: *arXiv preprint arXiv:2001.10872* (2020).

[Ber+20b] V. Bergholm et al. "PennyLane: Automatic differentiation of hybrid quantum-classical computations." In: *arXiv preprint arXiv:1811.04968* (Feb. 2020). arXiv: `1811.04968 [physics]`.

[Bia+17] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd. "Quantum machine learning." In: *Nature* 549(7671) (2017), pp. 195–202.

[Bro+21] J. Brownlee, Q. Zhang, L. Ma, Y. Yang, J. Liang, H. Heidari, and A. Abbasi. "Machine Learning Applications: A Survey." In: *arXiv preprint arXiv:2109.07207* (2021).

[Car+22a] M. C. Caro, H.-Y. Huang, M. Cerezo, K. Sharma, A. Sornborger, L. Cincio, and P. J. Coles. "Generalization in quantum machine learning from few training data." In: *Nature communications* 13.1 (2022), p. 4919.

[Car+22b] M. C. Caro, H.-Y. Huang, M. Cerezo, K. Sharma, A. Sornborger, L. Cincio, and P. J. Coles. "Generalization in quantum machine learning from few training data." In: *Nature communications* 13.1 (2022), p. 4919.

[CCL19] I. L. Cong, S. Choi, and M. D. Lukin. "Quantum convolutional neural networks." In: *Nature Physics* 15.12 (2019), pp. 1273–1278.

[Chu+23] J. Chu, X. He, Y. Zhou, J. Yuan, L. Zhang, Q. Guo, Y. Hai, Z. Han, C.-K. Hu, W. Huang, et al. "Scalable algorithm simplification using quantum AND logic." In: *Nature Physics* 19.1 (2023), pp. 126–131.

[DR22] A. Dames and E. Richuso. *What Is Quantum-Safe Cryptography and Why Do We Need It?* IBM Cloud Blog. Accessed on March 10, 2022. Mar. 2022.

[GAO19] U. G. A. O. (GAO). *Status and Prospects for Quantum Computing*. Tech. rep. GAO-19-204SP. 2019.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[Har21] Harvard Office of Technology Development. "Quantum Computer Simulates Molecule's Behavior Accurately." In: *Harvard OTD News* (2021).

[Hav+19] V. Havlıček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta. "Supervised learning with quantum-enhanced feature spaces." In: *Nature* 567.7747 (2019), pp. 209–212.

[HKP22] T. Hur, L. Kim, and D. K. Park. "Quantum convolutional neural network for classical data classification." In: *Quantum Machine Intelligence* 4.1 (2022), p. 3.

[IS23] R. Ibrahim and M. O. Shafiq. "Explainable Convolutional Neural Networks: A Taxonomy, Review, and Future Directions." In: *ACM Computing Surveys* 55.10 (2023), pp. 1–37.

[Jaz+19] F. Jazaeri, A. Beckers, A. Tajalli, and J.-M. Sallese. "A review on quantum computing: From qubits to front-end electronics and cryogenic MOSFET physics." In: *2019 MIXDES-26th International Conference" Mixed Design of Integrated Circuits and Systems"*. IEEE. 2019, pp. 15–25.

[JZH21] C. Janiesch, P. Zschech, and K. Heinrich. "Machine learning and deep learning." In: *Electronic Markets* 31.3 (2021), pp. 685–695.

[Kly+14] M. Klysik, S. Garg, S. Pokharel, J. Meier, N. Patel, and K. Garg. "Challenges of imaging for cancer in patients with diabetes and obesity." In: *Diabetes technology & therapeutics* 16.4 (2014), pp. 266–274.

[KSH12] A. Krizhevsky, I. Sutskever, and G. Hinton. "ImageNet classification with deep convolutional neural networks." In: *Advances in Neural Information Processing Systems*. Vol. 25(2). 2012, pp. 1097–1105.

[Lar+22] M. Larocca, F. Sauvage, F. M. Sbahi, G. Verdon, P. J. Coles, and M. Cerezo. "Group-invariant quantum machine learning." In: *PRX Quantum* 3.3 (2022), p. 030341.

[LBH15] Y. LeCun, Y. Bengio, and G. Hinton. "Deep learning." In: *Nature* 521(7553) (2015), pp. 436–444.

[Li+18] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. "Visualizing the Loss Landscape of Neural Nets." In: *Advances in Neural Information Processing Systems (NeurIPS)* 31 (2018), pp. 6389–6399.

[Liu+21] J. Liu, K. H. Lim, K. L. Wood, W. Huang, C. Guo, and H.-L. Huang. "Hybrid quantum-classical convolutional neural networks." In: *Science China Physics, Mechanics & Astronomy* 64.9 (2021), p. 290311.

[Mac+22] I. MacCormack, C. Delaney, A. Galda, N. Aggarwal, and P. Narang. "Branching quantum convolutional neural networks." In: *Physical Review Research* 4.1 (2022), p. 013117.

[Mat+22] A. Matic, M. Monnet, J. M. Lorenz, B. Schachtner, and 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.

[McC+18] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. "Barren plateaus in quantum neural network training landscapes." In: *Nature communications* 9.1 (2018), p. 4812.

[Mit97] T. Mitchell. *Machine Learning*. McGraw Hill Education, 1997.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[NC02] M. A. Nielsen and I. Chuang. *Quantum computation and quantum information*. 2002.

[Pa19] A. Paszke and et al. "PyTorch: An imperative style, high-performance deep learning library." In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019, pp. 8026–8037.

[Pes+21] A. Pesah, M. Cerezo, S. Wang, T. Volkoff, A. T. Sornborger, and P. J. Coles. "Absence of barren plateaus in quantum convolutional neural networks." In: *Physical Review X* 11.4 (2021), p. 041011.

[Pre+19] L. M. Prevedello, S. S. Halabi, G. Shih, C. C. Wu, M. D. Kohli, F. H. Chokshi, B. J. Erickson, J. Kalpathy-Cramer, K. P. Andriole, and A. E. Flanders. "Challenges related to artificial intelligence research in medical imaging and the importance of image analysis competitions." In: *Radiology: Artificial Intelligence* 1.1 (2019), e180031.

[Sch+20] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe. "Circuit-centric quantum classifiers." In: *Physical Review A* 101.3 (2020), p. 032308.

[Sen+22] P. Sen, A. S. Bhatia, K. S. Bhangu, and A. Elbeltagi. "Variational quantum classifiers through the lens of the Hessian." In: *Plos one* 17.1 (2022), e0262346.

[SII21] SIIT. *How Logic Gates Work: OR, AND, XOR, NOR, NAND, XNOR, and NOT*. `https://siit.co/blog/how-logic-gates-work-or-and-xor-nor-nand-xnor-and-not/2918`. 2021.

[SJA19] S. Sim, P. D. Johnson, and A. Aspuru-Guzik. "Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms." In: *Advanced Quantum Technologies* 2.10 (Oct. 2019). DOI: `10.1002/qute.201900079`.

[SK22a] D. Sarvamangala and R. V. Kulkarni. "Convolutional neural networks in medical image understanding: a survey." In: *Evolutionary intelligence* 15.1 (2022), pp. 1–22.

[SK22b] M. Schuld and N. Killoran. "Is quantum advantage the right goal for quantum machine learning?" In: *Prx Quantum* 3.3 (2022), p. 030101.

[SSM21] M. Schuld, R. Sweke, and J. J. Meyer. "Effect of data encoding on the expressive power of variational quantum-machine-learning models." In: *Physical Review A* 103.3 (2021), p. 032430.

[Sta23] Stanford University. *CS230 Deep Learning*. `https://cs230.stanford.edu/`. 2023.

[Tho] G. Thomas. *Visualizing the loss landscape of neural nets*. `https://www.cs.umd.edu/~tomg/projects/landscapes/`.

[Tsi20] S. Tsimenidis. "Limitations of deep neural networks: A discussion of G." In: *Marcus' critical appraisal of deep learning. ArXiv* (2020).

[Uni] D. L. University. *Qiskit Hadamard Gate Tutorial*.

[Wik21] Wikipedia. *Controlled NOT gate — Wikipedia, The Free Encyclopedia*. [Online; accessed 24-March-2023]. 2021.

[Wu17] J. Wu. "Introduction to convolutional neural networks." In: *National Key Lab for Novel Software Technology. Nanjing University. China* 5.23 (2017), p. 495.

[Ya21] J. Yang and et al. "Medmnist v2: A large-scale lightweight benchmark for 2D and 3D biomedical image classification." In: *CoRR* (2021). arXiv: `2110.14795 [cs.CV]`.

[ZLZ21] Y. Zhang, J. Liang, and T. Zhang. "Embedding Principle of Loss Landscape of Deep Neural Networks." In: *arXiv preprint arXiv:2105.14573* (2021).

[ZS19] Z. Zhang and E. Sejdić. "Radiological images and machine learning: trends, perspectives, and prospects." In: *Computers in biology and medicine* 108 (2019), pp. 354–370.