VASSILI PRIVALIHHIN

RECENT ADVANCES IN QUANTUM MACHINE LEARNING: A SURVEY WITH A COMPARATIVE ANALYSIS

UNIVERSIDADE DO ALGARVE
Faculdade de Ciências e Tecnologia
2024

VASSILI PRIVALIHHIN

RECENT ADVANCES IN QUANTUM MACHINE LEARNING: A SURVEY WITH A COMPARATIVE ANALYSIS

Master of Science in Informatics Engineering

Work done under the supervision of: José Valente de Oliveira
Joana Coutinho Sousa

UNIVERSIDADE DO ALGARVE
Faculdade de Ciências e Tecnologia
2024

RECENT ADVANCES IN QUANTUM MACHINE LEARNING: A SURVEY WITH A COMPARATIVE ANALYSIS

Declaração de autoria de trabalho

Declaro ser o autor deste trabalho, que é original e inédito. Autores e trabalhos consultados estão devidamente citados no texto e constam da listagem de referências incluída.

Declaration of authorship of the work

I hereby declare to be the author of this work, which is original and unpublished. Authors and works consulted are properly cited in the text and included in the reference list.

(Vassili Privalihhin)

Abstract

Quantum mechanics, a fundamental theory in physics that describes the behavior of nature at atomic and subatomic scales, offers significant advantages over classical physics in various contexts. These advances have laid the foundation for the field of quantum computing, which leverages the unique properties of quantum mechanics to solve complex computational problems more efficiently than classical computers. At the core of quantum computing are qubits, the quantum analogs of classical bits. Unlike classical bits, which are restricted to binary states (0 or 1), qubits utilize superposition and entanglement, enabling them to exist in multiple states simultaneously and interact in ways that amplify computational capabilities.

This study presents an overview of quantum algorithms and tools for quantum machine learning, with a focus on recent advancements. Through a comparative analysis and empirical evidence, the research highlights the potential advantages of quantum algorithms in various applications, such as data processing, pattern recognition, and algorithmic complexity. The findings suggest that quantum methods may surpass their classical counterparts in certain domains. However, a key challenge remains: the current limitations of quantum hardware. Despite the theoretical benefits, practical implementation is constrained by the noisy and error-prone nature of quantum devices. Consequently, the experiments conducted in this study were performed in simulation environments, which demonstrated potential improvements when applying quantum paradigms.

Keywords: Quantum Machine Learning, Quantum Neural Network, Quantum Computing, Quantum Algorithms, Quantum Circuits, Quantum Kernel Methods, Quantum Simulations, Hybrid Quantum-Classical Models

Resumo

A mecânica quântica, uma teoria fundamental da física que descreve o comportamento da natureza em escalas atómicas e subatómicas, oferece vantagens significativas em relação à física clássica em vários contextos. Estes avanços estabeleceram as bases para o campo da computação quântica, que aproveita as propriedades únicas da mecânica quântica para resolver problemas computacionais complexos de forma mais eficiente do que os computadores clássicos. No cerne da computação quântica estão os qubits, os análogos quânticos dos bits clássicos. Ao contrário dos bits, que estão restritos aos estados binários (0 ou 1), os qubits utilizam os princípios da superposição e do entrelaçamento, permitindo que existam em vários estados simultaneamente e interajam de maneiras que amplificam as capacidades computacionais.

Este estudo apresenta uma visão geral dos algoritmos quânticos e ferramentas para aprendizagem automática quântica, com foco nos avanços recentes. Através de uma análise comparativa e evidências empíricas, a pesquisa destaca as potenciais vantagens dos algoritmos quânticos em várias aplicações, como processamento de dados, reconhecimento de padrões e complexidade algorítmica. Os resultados sugerem que os métodos quânticos podem superar os seus equivalentes clássicos em certos domínios. No entanto, um desafio significativo persiste: as limitações atuais do hardware quântico. Apesar dos benefícios teóricos, a implementação prática é limitada pela natureza ruidosa e propensa a erros dos dispositivos quânticos. Consequentemente, as experiências realizadas neste estudo foram executadas em ambientes de simulação, que demonstraram potenciais melhorias com a aplicação de paradigmas quânticos.

Palavras Chave: Aprendizagem Máquina Quântica, Rede Neural Quântica, Computação Quântica, Algoritmos Quânticos, Circuitos Quânticos, Métodos de Kernel Quântico, Simulações Quânticas, Modelos Híbridos Quântico-Clássicos

Contents

List of Figures

List of Tables

List of Acronyms

QML	Quantum Machine Learning.
NN	Neural Network.
CNN	Convolutional Neural Network.
QNN	Quantum Neural Network.
PQC	Parameterized Quantum Circuit.
QSVM	Quantum Support Vector Machines.
NISQ	Noisy Intermediate-Scale Quantum.
QPCA	Quantum Principal Component Analysis.
QEC	Quantum Error Correction

— *It always seems impossible until it's done.*

Nelson Mandela

1

Introduction

With the prominent field of study such as Quantum Computing (QC) gaining a lot of popularity in the last decade, some questions started to arise, especially in relation to its applicability to another prominent field, such as Artificial Intelligence (AI), since an intersection of these two may revolutionize the development of AI, specifically the whole paradigm of machine learning, bringing new computational capabilities (2). These new computational capabilities may improve the already existing traditional machine learning algorithms, making them much more efficient especially when applied on huge amounts of data, or data with high complexity in terms of patterns and dimensions, highlighting the importance of the study (3). This report examines recent advances in quantum machine learning, presenting practical examples and empirical evidence to explore their potential for solving complex problems more efficiently than classical methods.

1.1 Methodology and materials for the literature review

To ensure a structured and thorough literature search and review, an organized search strategy was adopted as well. The literature search and review process mainly involved defining relevant databases and applying inclusion and exclusion criteria in order to filter out unsuitable studies. A search string was created to retrieve literature on key topics within quantum computing and machine learning. The IEEE Xplore database was mainly used for its broad coverage of general-purpose journals and its prominence in the field of computer science. It also offered extensive search capabilities and is accessible through the University of the Algarve's licensing agreements. The initial search yielded a total of more than 7,900 studies published between 2013 and 2024. The following query string was used to gather relevant studies:

′("Quantum" AND ("Clustering" OR "Support Vector Machines" OR "Decision Trees" OR "Classification" OR "Regression" OR "Reinforcement Learning" OR "Supervised Learning" OR "Unsupervised Learning" OR "Random Forest" OR "Deep Learning" OR "Neural Network" OR "Data Analysis" OR "Natural Language Processing" OR "Computer Vision" OR "Convolutional Neural Networks" OR "Gradient Boosting" OR "Principal Component Analysis" OR "Feature Engineering" OR "Model Evaluation" OR "Cross-Validation" OR "Hyperparameter Tuning" OR "Transfer Learning" OR "Ensemble Learning"))′.

After the search, a set of relevant studies was selected for further review based on predefined inclusion and exclusion criteria, which will be outlined in a subsequent section. However, another decent set of studies and respective journals was acquired through the search engine as well, especially when a particular study could not be acquired through the selected database.

1.2 Inclusion and exclusion criteria

The inclusion and exclusion criteria had to be defined as well in order to identify the relevant papers for our literature review to be able to answer the research questions.

The inclusion criteria is defined as follows:

1) The papers are written in English.

2) The papers have been published in the last 11 years, unless they present foundational or highly-cited theoretical contributions to quantum paradigms in machine learning.

3) The papers present a clear view of quantum paradigms employed for machine learning tasks.

4) The papers provide either practical applications (e.g., simulators, quantum devices) or significant theoretical advancements relevant to quantum paradigms in machine learning.

Accordingly, the exclusion criteria is defined as follows:

1) The papers are not written in English.

2) The papers are older than 11 years, except for foundational or highly-cited theoretical works that remain relevant.

3) The papers do not provide a clear view of quantum paradigms employed for machine learning tasks.

4) The papers do not contribute either practical applications or meaningful theoretical insights into quantum paradigms for machine learning.

1.3 Objectives and Scope of the Study

The main objective of the study being conducted is first of all to identify the current state of the art and its latest tendencies and developments in the field of quantum machine learning and its potential.

1.3.1 Research questions

The main research questions, which will help to navigate the study in the right direction are the following:

1) What are the primary quantum computing paradigms applied in machine learning?

2) What are the main applicable fields for quantum-enhanced algorithms?

1.4 Context of the work

To better understand the context of the work, one should understand how quantum computing and machine learning came to intersect with each other. First of all, machine learning as a known concept was coined by Samuel A. L. (4), where a study was conducted in order to confirm whether a computer could be programmed in a way that it can play a better game of checkers than a person who wrote the program. In conclusion, the computer was able to outperform the person after 8-10 hours of machine-playing time, which would be considered quite too much of time nowadays for a task like this, as since then machine learning has improved to a much greater extent.

As for the history of quantum computing development, its concepts started to emerge in 1980s, with the introduction of quantum Turing machine by an American physicist (5), but practical implementation of an actual quantum computer took some time. L C. Isaac et al. (6) introduced the two-qubit quantum computer to implement the Grover's search algorithm and the results were positive, as the computer required a fewer steps to come to the final state than the classical computer. Since then, there were huge advancements in the field, especially with future experiments by increasing the amount of qubits and accordingly decreasing the error rates (7).

At some point in time, starting in early 2000s, both fields of machine learning and quantum computing evolved to intersect (8), leading to emergence of new field called

Quantum Machine Learning (QML). One of the first studies in the field demonstrated how quantum computation could be used in pair with neural networks to implement quantum associative memory by employing simple spin of two-state quantum systems and representing patterns as quantum operators (9). As of late 2010s until now, there are some notable advancements in the field, where prominent companies such as Google and IBM make it possible to create, run and benchmark your custom quantum circuits in their Cloud environments (10), which can further be integrated with machine learning algorithms. Additionally, there are specific libraries, written in various programming languages, available for everyone to utilize which include quantum-enhanced classical algorithms, allowing for convenient experimentation (11). Hence, the tools and the quantum-enhanced algorithms are to be reviewed within the field to further compare their performance with classical algorithms. Therefore, the main challenge of the work here is to identify these quantum-enhanced algorithms and the tools which supposedly overcome their classical alternatives. Overcoming such a challenge may help the community to further analyse and integrate such algorithms into the future AI solutions offering potentially unmatched precision.

To summarize the section, Quantum Machine Learning has seen significant advancements in recent years, with potential applications in fields ranging from drug discovery to financial modeling. A recent systematic review by García et al. (12) provides a comprehensive overview of QML techniques and their practical implementations.

1.5 Organization of the study

The report is structured to first provide the reader with a clear understanding of the motivation behind the work, highlighting its significance and relevance in the field. The main objective of the research is then outlined, followed by the formulation of specific research questions that guide the investigation.

In the Developed Work chapter, the experiments are presented in detail, incorporating some concepts from the literature reviewed as well. This chapter explains the methods used and describes the algorithms applied, emphasizing the quantum computing paradigms involved. Each experiment is designed to extract meaningful insights, and the process of applying quantum methods to machine learning is explained.

Next, in the Experimental Results chapter, the outcomes of the experiments are displayed through comprehensive data visualizations, statistical analyses, and their accompanying interpretations. This section also includes a Discussion to evaluate the significance of the results and their implications for both theory and practical applications.

Finally, the study is concluded in the Conclusion chapter, where the strengths and limitations of the work are analyzed. The findings are referenced to assess whether the initial objectives were met, and recommendations are made for potential improvements and future research directions.

2
State of the Art

In this chapter, some of the latest studies conducted by others in the realm of quantum algorithms and tools tailored for machine learning applications are thoroughly reviewed to get the first insights of where we are now and where we are heading. This will help us to also identify the gaps in the field which need to be addressed to advance further and accordingly answer the research questions.

2.1 Key Quantum Concepts

In order to answer the first research question, the key quantum concepts and their paradigms employed in machine learning are to be identified within this section.

One of the main quantum paradigms described in the works reviewed is the use of qubits to encode the classical data into quantum states, an idea especially prominent in the classification tasks.

Starting with the most fundamental notion within the field, a qubit. A qubit is the unit of quantum information in every quantum computer, serving as the quantum counterpart to the classical bit. Unlike a classical bit, which is deterministic and can exist only in one of two states (0 or 1), a qubit exploits the principles of quantum mechanics to exist in a superposition of states. This means that a qubit can represent 0, 1, or any quantum superposition of these states, for example by being able to represent 0 and 1 at the same, but with various probabilities. These probabilities are defined by the qubit's wave function, which describes its quantum state and determines the probability distribution of its possible values. However, when a qubit is measured, this superposition collapses to a single classical state (either 0 or 1), consistent with the probabilities defined by its wave function prior to measurement (13). This unique property of qubits allows quantum computers to process and store information in ways that are fundamentally different from classical computers. Therefore, encoding the classical data into quantum states would allow the further application of quantum computing for machine learning tasks. One of the following reports will address such a use-case in order to solve an image classification problem.

Additionally, many papers on that topic introduce the notion of quantum circuits. Essentially, a quantum circuit is a sequence of operations (known as quantum gates) that manipulate the state of qubits. One common quantum gate, which will be introduced later on in the developed work chapter, is the Pauli-X Gate, also known as the quantum NOT gate, which is one of the fundamental quantum gates in quantum computing (14). Its primary function is to flip the state of a qubit, analogous to the classical NOT gate, which inverts a binary bit. The action of the Pauli-X Gate is as follows: If a qubit is in the state $|0\rangle$, applying the Pauli-X gate will flip it to the state $|1\rangle$. Conversely, if a qubit is in the state $|1\rangle$, the Pauli-X gate will flip it to the state $|0\rangle$. Mathematically, the Pauli-X gate is represented by the following matrix:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

When this matrix acts on a qubit, it performs the bit-flip operation. For example:

$$X|0\rangle = |1\rangle \quad \text{and} \quad X|1\rangle = |0\rangle$$

In addition to flipping the state of a qubit, the Pauli-X gate is also important because it serves as a building block for more complex quantum operations and is essential in the creation of entangled states, particularly when used in combination with other gates like the Controlled-NOT (CNOT) gate (15).

2.2 The Bloch Sphere

Another crucial concept used in the field of quantum computing is the Bloch Sphere, which is a geometrical representation of the quantum state of a two-level quantum system, specifically a qubit. It provides an intuitive way to visualize the state of a qubit as a point on the surface of a unit sphere in three-dimensional space (16). In the Bloch Sphere, any pure quantum state of a qubit can be represented as a point on the sphere's surface. The general quantum state of a qubit, denoted as $|\psi\rangle$, is typically expressed as a linear combination of the basis states $|0\rangle$ and $|1\rangle$:

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$$

Here, θ and ψ are spherical coordinates that describe the position of the qubit on the Bloch Sphere:

- θ $(0 \leq \theta \leq \pi)$ represents the angle between the qubit's state vector and the positive z-axis.
- ϕ $(0 \leq \phi < 2\pi)$ represents the azimuthal angle in the x-y plane from the positive x-axis.

The north and south poles of the Bloch Sphere correspond to the basis states $|0\rangle$ and $|1\rangle$, respectively. Superpositions of these states, which are crucial in quantum computing, are represented by points on the sphere between these poles.

The notion of the Bloch Sphere is particularly important because it provides insights into the effects of quantum gates on qubits. Quantum operations, such as the Pauli-X, Y, and Z gates, correspond to rotations of the qubit's state vector around specific axes on the Bloch Sphere (17). For instance:

- The Pauli-X gate induces a 180° rotation about the x-axis, flipping the qubit between $|0\rangle$ and $|1\rangle$.
- The Pauli-Y and Z gates similarly rotate the qubit state around the y-axis and z-axis.

Understanding these rotations is crucial for analyzing and designing quantum circuits, which will be often used in the succeeding sections, as the operations performed by quantum gates directly manipulate the qubit's state in this spherical representation (See Figure 2.1) (18).

2.3 Quantum Circuits

In quantum machine learning, the circuits are often parameterized, meaning they include adjustable parameters that are optimized during training, much like the weights in classical machine learning models, something that has been presented by C. Marco et al. in (19), where the Variational Quantum Algorithms (VQAs) have shown promise. VQAs essentially use hybrid quantum-classical approaches where quantum circuits are optimized via classical optimizers to solve problems in quantum chemistry and machine learning as well.

Another crucial paradigm observed in some reviewed papers, which efficiently improves the accuracy of models, is the use of quantum entanglement within quantum circuits. Entanglement becomes relevant when a circuit involves two or more qubits.

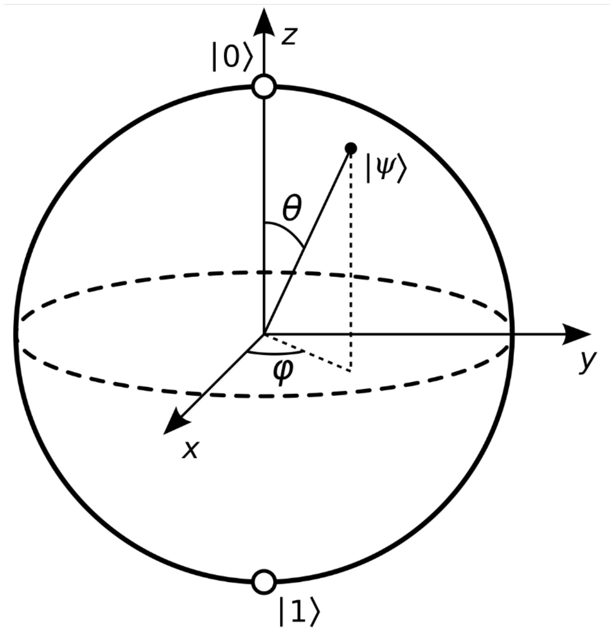

Figure 2.1: The Bloch Sphere

In quantum computing, entanglement is a phenomenon where the quantum states of two or more qubits become interconnected, such that the state of one qubit cannot be described independently of the state of the other(s) (20). This entangled state is typically established using specific quantum gates within the circuit. A common example is the Controlled-NOT (CNOT) gate, which is widely used to create entanglement by entangling the state of one qubit (the control qubit) with another qubit (the target qubit). In the context of a classification problem, this entanglement can lead to correlated measurement outcomes, meaning that the measurement of one qubit could provide information about the expected outcome of another, depending on the nature of the entangled state.

Another important definition to be aware of in the context of quantum circuits, is Observable. An observable is a Hermitian operator that represents a measurable physical quantity in a quantum system. When a quantum state is measured, the observable produces an eigenvalue corresponding to one of the operator's eigenstates, providing a meaningful result such as a qubit's state. Observables are crucial for extracting information from quantum circuits. For example, the Pauli-Z operator is commonly used to measure whether a qubit is in the $|0\rangle$ or $|1\rangle$ state. In quantum machine learning, observables are applied at the end of a quantum circuit to obtain output values for tasks like classification. The combination of different observables, such as Pauli-Z and the identity operator, can help project quantum states and derive binary results essential for classification. Therefore, by selecting appropriate observables, quantum models can effectively interpret and output relevant information from qubits (21).

Additionally, some of the papers reviewed, especially in the field of clustering problems, apply circuits such as swap-test as an alternative to the classical Euclidean distance in order to estimate the distance between two vectors with data, essentially suggesting an improvement in distance-based clustering techniques like K-means algorithm. Speaking of clustering, Particle Swarm Optimization with the employment of quantum paradigms into it for solving clustering problems suggests a considerable improvement in performance (22).

2.4 Quantum Support Vector Machines

Quantum Support Vector Machines (QSVM) are quantum-enhanced versions of classical Support Vector Machines (SVM), which are widely used for classification tasks. The key innovation in QSVM lies in applying quantum computing's ability to handle high-dimensional data spaces more efficiently through the use of **quantum kernels** and **feature maps**. Classical SVMs map data into higher-dimensional feature spaces using kernels, allowing linear separation in complex datasets. QSVMs achieve a similar goal by using quantum circuits to generate quantum feature maps, which can exploit the quantum principles of superposition and entanglement to represent data more compactly and allegedly process it faster than classical methods. So, one of the significant advances brought by QSVMs, is the use of quantum kernels (23). A quantum kernel represents the inner product of quantum states corresponding to data points. For instance, the inner product between two quantum states $|\psi(x)\rangle$ and $|\psi(x')\rangle$, representing the quantum kernel, will be given by:

$$K(x, x') = \langle \psi(x) | \psi(x') \rangle$$

Quantum computers can compute these kernels exponentially faster than classical systems in certain cases, especially for data embedded into high-dimensional Hilbert spaces (24). A Hilbert space is an abstract vector space that generalizes the concept of Euclidean space to an infinite number of dimensions (25). In quantum computing, states are represented as vectors in these spaces, which allows for more complex computations. This quantum feature mapping allows QSVMs to tackle complex classification problems more efficiently than classical SVMs, where computing inner products in a high-dimensional space becomes computationally expensive. Quantum kernel methods have shown potential advantages in fields like drug discovery, finance, and high-energy physics analyses (26), due to their ability to manage large feature spaces.

Another major advance is the development of quantum feature maps, where classical data is encoded into quantum states via a parameterized quantum circuit. These feature maps enable the classifier to capture intricate patterns and correlations in the data that classical methods might miss. Different types of quantum feature maps, such as Pauli feature maps and ZZ feature maps, have been used in QSVMs to map classical data into a higher quantum state space (23).

So with all this, QSVMs bring potential speedups in processing large datasets with high-dimensional features, as classical SVMs rely on approximations when dealing with complex data structures, but QSVMs can exploit quantum parallelism to compute inner products more efficiently (3). This is particularly important for tasks like image classification, natural language processing, and other domains involving high-dimensional feature spaces.

QSVMs are also designed to work on current Noisy Intermediate-Scale Quantum (NISQ) devices. Although quantum hardware is still limited in terms of qubit coherence and gate reliability, QSVMs have been adapted to work with these limitations, showing promising results in early experiments on quantum hardware. For example, hybrid QSVMs that combine classical preprocessing with quantum kernels have demonstrated competitive results on NISQ devices (27).

2.5 Quantum Principal Component Analysis

Quantum Principal Component Analysis (QPCA) is the quantum analog of classical Principal Component Analysis (PCA), which is widely used in data analysis and machine learning to reduce the dimensionality of data while preserving its most significant features. In classical PCA, the algorithm identifies the principal components, or directions in the data space where the variance is highest and projects the data onto these components. QPCA performs a similar task, but it applies quantum computing's ability to process large datasets and handle high-dimensional spaces more efficiently by encoding data into quantum states. One of the most significant advances QPCA brings is the potential for exponential speedup in data processing. Classical PCA involves diagonalizing covariance matrices to find principal components, which can be computationally expensive for large datasets. QPCA can diagonalize quantum density matrices more efficiently, thanks to quantum parallelism, reducing the time complexity from polynomial to logarithmic in some cases (28). This makes QPCA particularly attractive for high-dimensional datasets, where classical PCA becomes computationally unusable.

QPCA operates directly on quantum states, making it suitable for quantum-native data that comes from quantum simulations or experiments, such as those in quantum chemistry or high-energy physics (29). Classical PCA struggles with quantum data, requiring inefficient conversions. In contrast, QPCA can identify the principal components of quantum states without converting them to classical data, maintaining the quantum correlations that classical methods might lose.

QPCA has also advanced through quantum noise tolerance. As a fact, quantum systems are prone to noise, which can distort data. Recent algorithms for QPCA can estimate eigenvalues and eigenvectors of noisy quantum states, offering resilience that is crucial for real-world applications on Noisy Intermediate-Scale Quantum (NISQ)

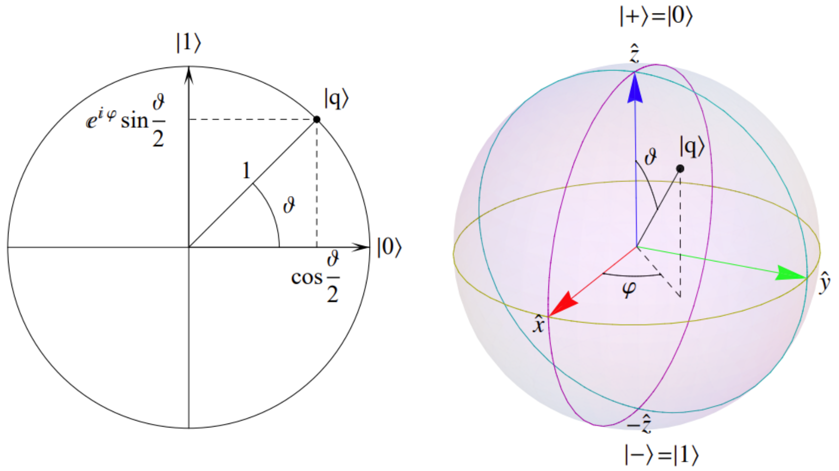

Figure 2.2: Representation of a qubit in Hilbert space (left) and on the Bloch sphere (right)

devices. This ability to estimate the most relevant components of noisy quantum data brings QPCA closer to practical use in fields such as quantum chemistry and material science (30).

There are some recently discovered claims regarding the improved dimensionality reduction brought by QPCA. While classical PCA reduces data dimensionality by projecting onto the most significant components, QPCA offers better efficiency in dimensionality reduction for quantum-enhanced machine learning tasks. Namely, the quantum advantage arises from the fact that QPCA can process data in **Hilbert spaces** of exponentially larger dimensions compared to classical spaces (24). This allows QPCA to preserve more complex relationships in the data while reducing its size for machine learning applications. See Figure 2.2 showing the representation of a qubit in the Hilbert space.

The left side of the figure shows a qubit's state represented in a 2D plane, where the qubit's state vector $|q\rangle$ is described using two angles, θ and ϕ. These angles control how much the qubit state is in $|0\rangle$ or $|1\rangle$, known as a superposition. The right side represents the same qubit on the Bloch sphere, a 3D visualization that helps to see how quantum states evolve. On the Bloch sphere, the angles θ and ϕ describe the position of the qubit's state on the surface of the sphere.

Recent studies, such as (30) and (29), have applied QPCA to areas like high-energy physics and quantum chemistry. In quantum chemistry, QPCA helps in analyzing molecular data by identifying the most relevant quantum states, which can be used for predicting molecular properties more accurately as claimed in (31).

2.6 Hybrid Quantum CNNs

In one study by K. Bishrelt (1), the author explored the application of a Hybrid Quantum Convolutional Neural Network (HQCNN) to classify emotions, specifically distinguishing between happy and sad expressions extracted from images. A notable concept introduced in their approach is data re-uploading, a method that iteratively feeds input data into a quantum circuit to enhance the circuit's processing capabilities, especially when constrained to a single qubit (32). This technique is somewhat analogous to the Universal Approximation Theorem in classical neural networks, where sufficient iterations and appropriate configurations can enable even a simple model to approximate complex functions.

The experiment was conducted using TensorFlow Quantum (TFQ), a quantum circuit simulator, chosen over cloud-based Noisy Intermediate-Scale Quantum (NISQ) devices due to accuracy limitations in the latter. The quantum circuit design includes layer gates, each consisting of a unitary operation that encodes input data by rotating qubits on the Bloch sphere. This is followed by a parameterized unitary gate optimized to minimize a cost function, which decreases when the measured quantum state closely matches the target class (32).

Here the Bloch sphere plays a crucial role in visualizing quantum states within the classification process. Distinct regions on the Bloch sphere represent different classes, and for multi-class problems, these regions should be well-separated. Measurements of the qubit's state are expected to fall near one of these regions, and repeated measurements with varied input data allow for the assessment of fidelity — a measure of how tightly clustered these quantum states are within their respective regions.

Despite the promise of single-qubit classifiers, that study acknowledges that they

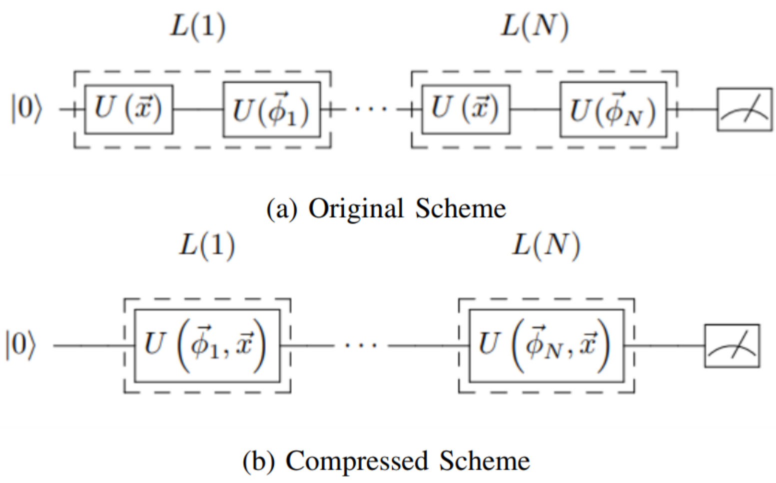

Figure 2.3: Quantum circuit scheme of single-qubit classifier with data re-uploading (1)

offer limited advantages over classical models. The introduction of additional qubits and the use of entanglement between them could enhance performance, but this would require more complex quantum circuits. The main distinction between classical and quantum CNNs highlighted in the study is the quantum CNN's ability to encode data into quantum states via a trainable quantum circuit (see its scheme in the Figure 2.3), with each iteration refining the circuit's parameters. The output qubit is measured using a Pauli-Z gate to yield a binary result.

The Hybrid Quantum Convolutional Neural Network model implemented in the study (1) consists of two quantum convolution layers, a max-pooling layer, and a classical fully connected layer, culminating in a softmax activation function. The binary classification task, focused on distinguishing happy emotions from sad emotions, showed improved accuracy when entanglement operations were incorporated, achieving a greater validation accuracy score. Taking into consideration this was accomplished using a dataset reduced to 1000 samples per class due to technical constraints, in contrast to other studies utilizing the full dataset (33).

Overall, that study suggests that ensemble methods, which average predictions from multiple models, could potentially reduce prediction variance, further enhancing model performance.

An obvious weak point in their study is the use of a single qubit only, since the performance could potentially be further improved by applying multiple qubits within the circuit, and therefore it would be possible to employ entanglement between these multiple qubits as well.

2.7 Hardware Advances for Quantum Machine Learning

Moving onto the physical state of the art, there are recent advancements in quantum hardware that have had an essential impact on making QML more practical and scalable. First of all, the number of qubits in quantum processors has grown significantly, with devices now containing over 100 qubits. For instance, IBM unveiled its 127-qubit quantum processor, named Eagle, in 2021, which was installed in the University of Tokyo in 2023 for further experimentation. Companies such as IBM and Google have pushed the limits of qubit scalability while also improving qubit coherence times (34), which measure how long qubits can maintain their quantum state. The improved coherence is crucial for performing longer computations, making complex quantum machine learning algorithms feasible. Y. Xinyuan et al. (35) discusses how various noise control techniques have been applied to superconducting qubits, boosting their coherence times and enhancing their reliability for quantum algorithms.

Another largest barrier to effective quantum computing has been qubit errors due to decoherence and noise within NISQ devices. **Quantum error correction (QEC)** codes have seen significant improvements, allowing quantum systems to maintain accurate computations for a longer time. Techniques such as surface codes and stabilizer codes are actively being implemented, reducing noise interference and improving error detection during quantum operations. Surface codes are topological error-correcting codes that use a grid of qubits to detect and correct local errors, such as bit-flip and phase-flip. They are highly scalable and are a leading candidate for fault-tolerant quantum computing (36). Stabilizer codes are a broader class of error-correcting codes that include surface codes as well. They use stabilizer operators to maintain quantum states and detect errors, so both techniques are crucial for reducing noise and improving accuracy during quantum computations (37). Moreover, the recent hardware designs have further focused on noise mitigation, improving the accuracy of quantum gate operations. Strategies like **dynamic decoupling** and **post-processing methods** have allowed QML models to run with fewer errors. For instance, a recent study on qudit dynamical decoupling (38) demonstrated how control pulses can reduce environmental noise and improve qubit fidelity in superconducting processors. Another study by M. Filip et al. (39) concerns post-processing methods which reduce noise-induced biases by applying post-processing on the outputs from quantum circuits. This method contrasts with quantum error correction, as it does not directly reduce noise in each individual circuit run, but instead adjusts the final results by mitigating the overall noise effect after the computations are completed. Noise reduction is especially important for hybrid quantum-classical algorithms, which are central to quantum machine learning applications on (NISQ) devices. Quantum hardware has also improved in terms of **qubit interconnectivity**, making operations between qubits more efficient. In this context, qubit interconnectivity refers to the ability of qubits within a quantum processor to interact with one another. In quantum computing, interconnectivity is crucial because operations like entanglement and multi-qubit gates (e.g. CNOT gate) rely on qubits exchanging quantum information. One study by Holmes et al. (40) specifically highlights how qubit interconnectivity affects the efficiency of quantum algorithms. This is all important for QML algorithms that often require entanglement across multiple qubits.

So, a key focus here has been the **scalability**, with quantum processors being designed to handle more qubits while maintaining connectivity. This has directly improved the capacity of quantum systems to handle QML tasks that require the manipulation of high-dimensional data. Hybrid quantum-classical systems, where quantum processors are integrated with classical systems, have also become a standard model, suggesting more efficient processing for machine learning tasks (41). Another study by Liu C. et al. (42) introduces the Quantum-Train (QT) framework, which integrates quantum computing with classical machine learning to address issues such as data encoding, model compression, and hardware requirements for inference. The QT framework employs a quantum neural network (QNN) combined with a classical mapping model to reduce the number of parameters required during training, while maintaining competitive performance.

2.8 Quantum Software Libraries

Quantum software libraries play a crucial role in the development of QML models by providing accessible frameworks for designing, simulating, and executing quantum circuits. The essential libraries, such as **Qiskit**, **Cirq**, and **TensorFlow Quantum** allow researchers to implement quantum algorithms without the need to directly access the quantum hardware. They enable simulations that model the behavior of quantum systems, for instance by integrating quantum operations with classical machine learning methods. By offering interfaces to quantum processors, these libraries also close the gap between theoretical quantum algorithms and practical implementation for those who do not have the capability to use real quantum computers.

One of the most essential libraries is **Qiskit**, which is an open-source framework developed by IBM for designing, simulating, and executing quantum circuits. It enables seamless integration with real quantum hardware, making it a powerful tool for implementing quantum machine learning models and exploring quantum algorithms on IBM's quantum processors (43).

Another important framework in the field is **Cirq**, which was developed by Google in 2018. It focuses on the precise design of quantum circuits and is highly adaptable for Noisy Intermediate-Scale Quantum (NISQ) hardware. It provides tools for optimizing quantum circuits to handle noise and imperfections in current quantum devices, making it a good choice for practical experimentation on real quantum systems.

Another commonly used framework is **TensorFlow Quantum**, which integrates quantum computing with the TensorFlow framework, allowing its users to build and train hybrid quantum-classical models as well. It is designed to support quantum machine learning, making it possible to combine quantum circuit simulations with classical machine learning workflows. Within our experimentation in the succeeding chapters, these frameworks will be often used to build various quantum circuits used for machine learning tasks.

These and many other related frameworks were thoroughly studied by Fürntratt et al. in (44) .

3

Developed work

The main focus of this chapter will be the field of image classification, being one common problem applied with quantum machine learning. The problem at hand chosen to benchmark the models is the classification of images into two clothing categories: Sandal and Ankle boot (see the Figure 3.1). The Fashion-MNIST dataset is chosen for training, which contains the respective images presented.

Just as like in the related study in the preceding chapter, the implementation is done within a simulation run with the TensorFlow and TensorFlow Quantum (TFQ) libraries, which were introduced within the work done by Broughton et al. (11). Additionally, the `"cirq"` framework is utilized in order to build the quantum circuits, and with its successful simulations, this circuit could be further evaluated on a real quantum device. However, the challenge here is the current availability of quantum devices

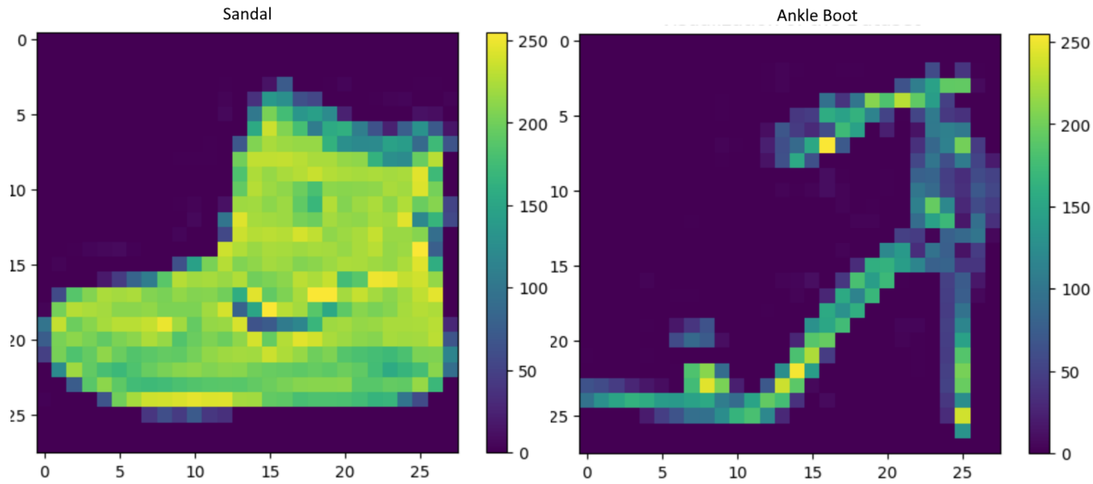

Figure 3.1: Data samples for the classification task

and their current state (them being NISQ devices, meaning they have a limited number of qubits and existing noise and error rates which may generate inaccurate results). Also, the proposed model is built with different values for parameters to see whether it could be improved further. To summarize, this chapter aims to address a classification problem, namely the classification of two different clothing categories, so then in the end we would have a model, which would be able to accurately classify an image of the said clothing piece, as opposed to happy and sad facial emotion presented in the related study in the preceding chapter.

For the experimentation purposes, the qBraid platform has been chosen, since it offered seamless work of frameworks such as TFQ and many others. It offers integrated IDE, where we could setup the desired environment based on the desired packages. In this case, the TensorFlow package with TFQ has been installed and activated with its Kernel. This platform is cloud-based, meaning that all the computations are done remotely on a server with no load on our local machine, considering that quantum simulations are quite heavy in terms of computations and related processes.

3.1 Data preprocessing

Starting with the data preparation part, which is the Fashion-MNIST dataset, a dataset of 60 000 28x28 grayscale images of 10 fashion categories, with a test set of 10,000 images, is loaded from the TensorFlow package and correspondingly split into the training and testing sets. The values within the original Fashion-MNIST dataset correspond to the presence of colour, or its intensity, 0 being Black, 1 - 254 being shades of gray, and 255 being white. The data within these sets had to be normalized to be in the range from 0 to 1, by dividing each pixel value by 255, so then the model could be trained effectively without issues with its gradients. This will also help us to converge our model faster due to an easier training process, which works best with smaller and similar to each other values. Because of this, the weights during the training will not get overly saturated, as an example. After cutting out the rest of the categories from the dataset and splitting it into training and testing set, we ended up with 10200 samples for the training set and 2000 samples for the testing set. As mentioned previously, we leave two fashion categories (classes) in the dataset only, turning the problem into a binary classification problem.

Furthermore, the images' data is downscaled to 2×2 pixels, but later some experiments were conducted with images downscaled to 4×4 pixels, due to the current limitations of quantum hardware, particularly the restricted number of qubits available as well as computational capabilities while done in a simulation. In this case, each downscaled image consists of 4, or 16 pixels, and these pixels can be encoded into the quantum states of 4 (or 16) qubits. The downscaling makes it feasible to encode and process the images within the constraints of quantum computing devices by allowing each pixel to be represented by a single qubit. See Figure 3.2 for an example of a downscaled image.

From the training set we further obtain the validation set being 15% of it, which will be used to evaluate the model's performance during training as well as evaluate its parameters.

Then we further simplify, or flatten, the images' data, where each image will be represented by one vector with four elements (N, 1, 4, 1), making it easier to process and encode it for quantum circuits. Each element represents the intensity of one pixel. This format is straightforward to encode into qubits, as each pixel (element) can directly

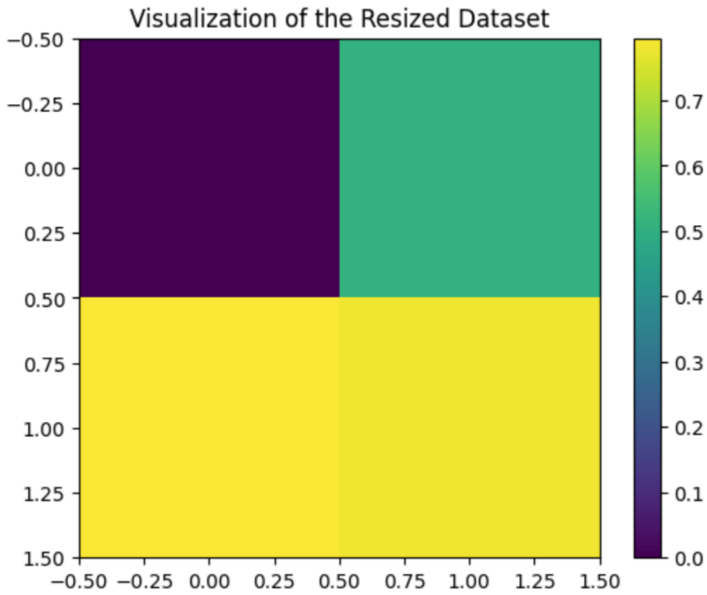

Figure 3.2: Downscaled sample of an image

correspond to a qubit state. Here is the structure of such a vector:

A = {N, 1, 4, 1}

where A_0 is the number of samples (hence images),

A_1 is the batch dimension, or just a placeholder for the batch size parameter,

A_2 is the number of pixels in one flattened image. Hence this dimension holds the intensity of each pixel,

A_3 is the channel dimension, or colour channel, which is '1' in this case as we are dealing with greyscale images.

Additional step in the data preprocessing part, is the binary encoding of the images' data, as currently the data is in the range from 0 to 1 in the float decimal format after the normalization step. During this step, values which are greater than 0.5, are converted to 1, and the values less than 0.5 are converted to 0. This step is necessary for the generation of quantum circuits, since 1 and 0 will dictate whether a quantum gate has to be created or not, respectively. Namely, if a value in the dataset is 1, then X-Pauli gate (flipping the state of the qubit from $|0\rangle$ to $|1\rangle$) is applied to the corresponding qubit. Generally, each image is converted to a quantum circuit, and each circuit will have as many qubits as there are pixels in one image.

For now, let us stick with all images being downscaled to 2×2 pixels for the sake of the following examples. To give an example of quantum circuit's generation, let us take the first image from the training set: [0., 0.41568628, 0.7137255, 0.73921573]. After binary encoding, it turns to: [0, 0, 1, 1]. And the generated quantum circuit will be as follows:

(1, 0): X-Gate

(1, 1): X-Gate

Recall that the values for the first image were [0,0,1,1]. This implies that we should apply X gate to the last two qubits since all the qubits are initially in the 0 states. Applying a X gate will change this state from zero to one. Therefore, we shall apply X Gate on the last two qubits. We have initialized the four qubits in a rectangular grid. Therefore, the initialized qubits are (0,0), (0,1), (1,0) and (1,1). In this circuit, note that we have a X gate on the qubits (1,0) and (1,1) which are the last two qubits. Hence, we have successfully created circuit for our image, and the same is done for the rest of images contained in the testing and validation sets.

Moreover, in order to evaluate the model's performance, Hinge Loss was employed as the loss function. Hinge Loss is particularly suitable for binary classification tasks, as it focuses on maximizing the margin between the decision boundary and the correctly classified points. In this context, the loss function penalizes not only incorrect predictions but also correct predictions that are too close to the decision boundary, ensuring that the model outputs are well-separated from the classification threshold. This characteristic makes Hinge Loss an appropriate choice for quantum machine learning models, where the objective is to clearly distinguish between the two possible classes based on the quantum state outputs, which may often be prone to errors and noise. The Hinge Loss can be integrated in other kinds of models as well, such as quantum support vector machines (QSVM) (45). To use the proposed Hinge Loss, the labels within all datasets were converted to 1 and -1. The Hinge Loss function is defined as:

$$L(y, f(x)) = \max(0, 1 - y \cdot f(x))$$

where y is the true label, and *f(x)* is the predicted value.

3.2 Quantum Neural Network Design

After the data preprocessing part, a model is ready to be defined, which will be the Quantum Neural Network. Building the Quantum Neural Network involves two major steps, building a class that adds gates layer by layer, and defining the QNN using the class from the first step. In essence, the QNN within this context, is another quantum circuit, which has four data qubits, and one read-out qubit, so 5 qubits in total. A few operations are also defined within that class, namely for the creation of a single-qubit and two-qubits gates. This circuit is also parameterized, so its weights are optimized during the training.

Starting with the readout qubit (-1, -1), we apply X-Pauli gate to it first to change its state to $|1\rangle$, and then apply the Hadamard gate, which would put the qubit into superposition of states: $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, which is necessary for subsequent Icing Coupling interactions, or in other words, the entanglement operations around X and Z axis' respectively (46). The circuit is finalized by applying another Hadamard gate by the end of the read-out qubit, in order to extract the output from the circuit in a binary format for the final classification. See the circuit's diagram in the Figure 3.3.

The quantum machine learning model is structured with two primary layers: an input layer and a Parameterized Quantum Circuit (PQC) layer. The input layer accepts quantum circuits derived from images, where each pixel is encoded as a qubit based on its value. These quantum circuits are then processed by the PQC layer, which includes both the data qubits and additional qubits such as the readout qubit. The PQC layer consists of trainable parameters that adjust the quantum operations applied to the qubits, such as entanglement through XX and ZZ couplings. The input circuits and the PQC are combined into a single composite quantum circuit, which is executed to produce an output. The construction of PQCs is a crucial theme in the area of quantum machine learning (47). Then the output, derived from the readout qubit, will have its superposition of states collapsed by the final Hadamard gate as explained earlier.

For model's training, the Adam optimizer has been employed, being an advanced gradient descent optimization algorithm that is widely used in training deep learning

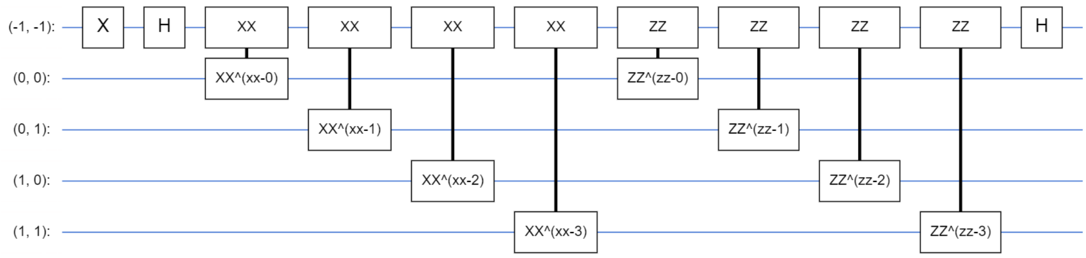

Figure 3.3: Diagram of Quantum Circuit

models, including Quantum Neural Networks (QNNs). It combines the benefits of two other popular optimizers: AdaGrad, which adapts the learning rate based on the frequency of updates, and RMSProp, which adjusts the learning rate based on the moving average of squared gradients. The Adam optimizer, frequently used in both classical and quantum machine learning, adapts the learning rate for each parameter by considering both the first moment (mean) and the second moment (uncentered variance) of the gradients. This makes Adam particularly effective in dealing with the noisy gradients often encountered in quantum models. Its ability to handle sparse gradients and maintain stable convergence through adaptive learning rates is critical for particularly optimizing quantum circuits. Another study highlights Adam's effectiveness in handling noisy gradients, which is crucial for quantum models (48).

For our experimentation purposes, the learning rate of 0.001 has been picked as a parameter in the Adam optimizer. Other values, less and greater than this, were tested as well, but they led to worse results, namely either too slow convergence, or too fast convergence, eventually leading to overfitting. Other parameters were set for training the model, such as batch size being 64 for the case when images were downscaled to 2x2 pixels (or 4 qubits), and the batch size of 128 for the case of 4x4 pixels (or 16 qubits), the decision driven by the hardware capabilities and data complexity, and 10 epochs at maximum were set, as being the point at which the model converges to its optimum and no fluctuations were observed in the loss and accuracy values further on.

The results observed after training the model are presented in the chapter of experimental results.

3.3 Quantum Classifier Implementation

In this section, we describe the construction of another model, a quantum classifier using a data re-uploading Parameterized Quantum Circuit (PQC) with two qubits. The observable in this classifier measures the state of the last qubit, employing a Pauli-Z operator. The Pauli-Z gate flips the phase of a qubit in the $|1\rangle$ state, making it well-suited for classification tasks. An identity observable is also constructed, and both observables are combined into $-0.5Z + 0.5I$, which projects the state onto $|1\rangle\langle 1|$, yielding 1 if the qubit is in the $|1\rangle$ state, and 0 otherwise. This results in a binary classification output.

The PQC with data re-uploading is initialized with two qubits and two layers to match the moderate data complexity of the problem. Entangling layers are applied, including terminal entanglement. The circuit applies rotation gates $R_x(\theta)$, $R_y(\theta)$, and $R_z(\theta)$ on each qubit, parameterized by θ, which are optimized during training. Controlled-Z gates are used between the qubits to introduce entanglement, with one after the first rotation layer and another as terminal entanglement. The overall architecture consists of the input layer, feeding data directly into the PQC, which functions as both the processing and output layer (see the Figure 3.4 of the built PQC). The model is compiled with the Adam optimizer, using a learning rate of 0.01. Mean Squared Error (MSE) is used as the loss function, and accuracy and AUC metrics evaluate performance. The training process begins with a batch size of 32 and 20 epochs, aligning with the complexity and size of the dataset.

3.4 Quantum CNN with Quantum Convolution Layers

Within this section, a Fully Quantum Convolutional Neural Network (QCNN) is presented, which builds upon both the Quantum Classifier described in a previous section and Quantum Convolution layers. The Quantum Convolution Layers are the quantum analog of classical convolutional layers, designed to process data using quantum circuits. These layers apply a series of quantum gates, such as Pauli-X, Pauli-Y, and

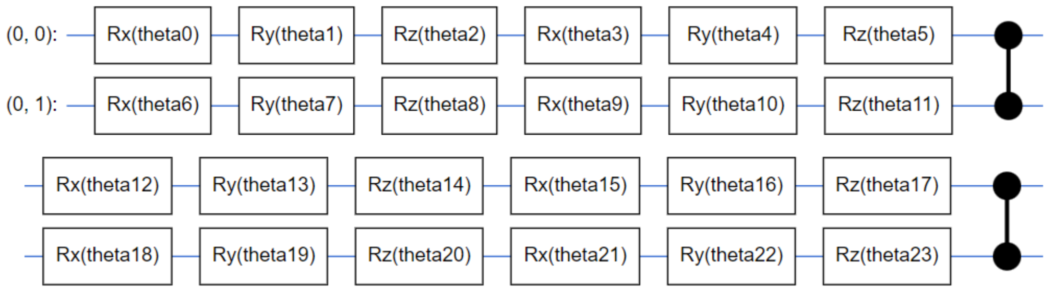

Figure 3.4: Parameterized Quantum Circuit

CNOT gates, to qubits that encode the input data, similar to how classical convolution layers apply filters. In this implementation, data is encoded into a set of n qubits, and the quantum convolution layer extracts features from the input data by performing operations across entangled qubits.

The architecture begins by creating observables for the quantum classifier. Each observable, including the Pauli-Z gate and identity ("I") gate, is applied to pairs of qubits. This design allows the model to be probabilistic in nature, outputting values such as the likelihood of a qubit being in state $|0\rangle$ or $|1\rangle$. Measurements are taken after the application of these gates, and the results are used to update the parameters of the Parameterized Quantum Circuit (PQC).

The final layer of the network is the Quantum Classifier, which uses a re-uploading PQC. The PQC consists of k layers, each with rotational gates (Rx, Ry, Rz) and entanglement gates to maintain coherence between qubits. The parameters of these gates are optimized during training using a gradient-based optimizer Adam, and the loss function is based on MSE for classification tasks.

The complete source code is provided in Appendix .1 for further reference in case of need for replication. The following chapter will present the experimental results, such as performance metrics evaluated over time, during and after the training process of the model presented.

4

Experimental results

This chapter outlines the experimental outcomes, including the performance indicators assessed over time, both during and following the training phase of the models discussed in the previous chapter on Developed work.

4.1 Comparison of Quantum and Classical Models

Making predictions for the input data from the testing set, we generated confusion matrices for the models trained on 2x2 and 4x4 images. These confusion matrices illustrate the number of mislabeled and correctly labeled images relative to their actual labels (see Figure 4.1 and Figure 4.2).

A few conclusions can be drawn from these results. For example, the model trained on 2x2 images was better at identifying ankle boots but frequently mislabeled sandals

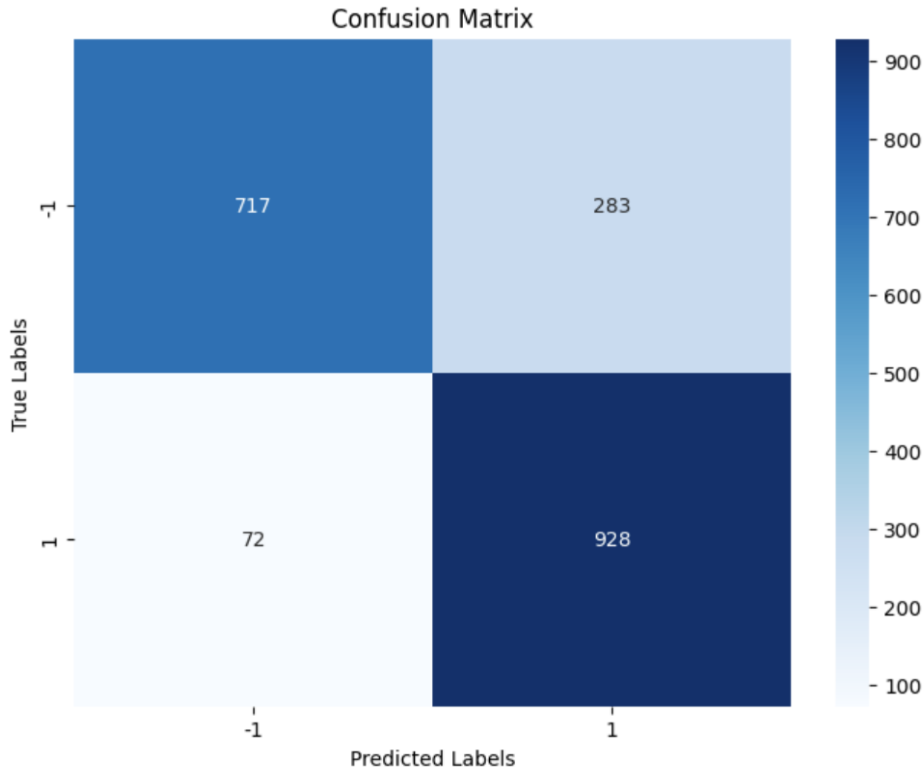

Figure 4.1: Confusion Matrix for model trained on 2x2 images

as ankle boots. This is likely due to slight imbalances in the dataset, which becomes less pronounced when the images are downscaled to 4x4 pixels, as seen in the improved confusion matrix (Figure 4.2).

In both quantum and classical models, specific parameters are being optimized during the training process. In the quantum model, the learned parameters are the rotation angles of the quantum gates in the parameterized quantum circuit (PQC) used in the classifier. These parameters are updated during training to minimize the loss function. In the classical convolutional neural network (CNN), the learned parameters include the weights and biases of the convolutional and fully connected layers. The training process is guided by gradient-based optimization algorithms. For this experiment, The Adam optimizer with a learning rate of 0.1 was used, optimizing the rotation angles in the PQC. As for the classical CNN, the Adam optimizer was used as well with a learning rate of 0.001 for adjusting weights and biases in the model.

As observed during and after the models' training, with the increase of pixels in images, the bias issue gets mitigated, most probably due to the model being able to learn

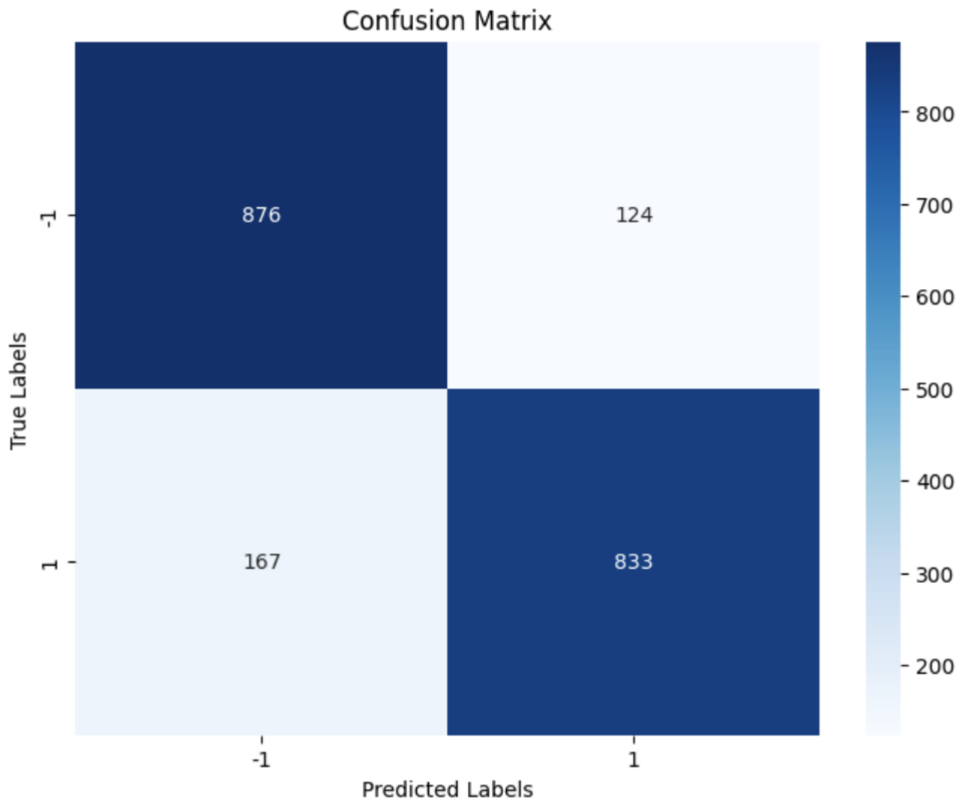

Figure 4.2: Confusion Matrix for model trained on 4x4 images

better with more features extracted, and eventually better at distinguishing sandals from ankle boots.

The following plots were generated to observe the change in accuracy over 10 epochs, and as can be seen, the accuracy did not noticeably change after the second epoch reaching the optimum of nearly 0.8 in case of images being downscaled to 2x2 (See the Figure 4.3), on the other hand reaching noticeably better accuracy in case of 4x4 images (See the Figure 4.4).

The rapid convergence of accuracy in the first case after the second epoch, reaching approximately 0.8, can be attributed to several factors. First, the simplicity of the model or the dataset might allow it to find a near-optimal solution quickly, and that is the most possible cause, since the datasets were considerably downscaled to contain 4 pixels, which are represented by 4 qubits. Second, the learning rate might be set relatively high, enabling faster convergence but limiting further improvements in accuracy, but as stated earlier, the picked value led to better results anyway. Finally, the model's capacity or regularization could be preventing overfitting, resulting in mini-

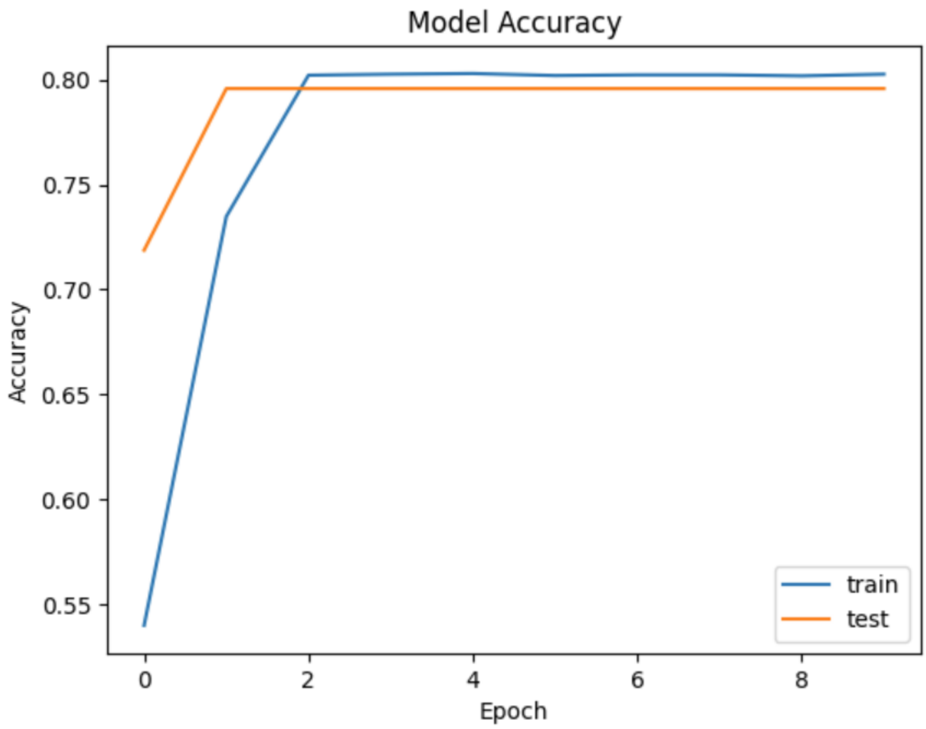

Figure 4.3: Plot of model's accuracy over each epoch with 2x2 images

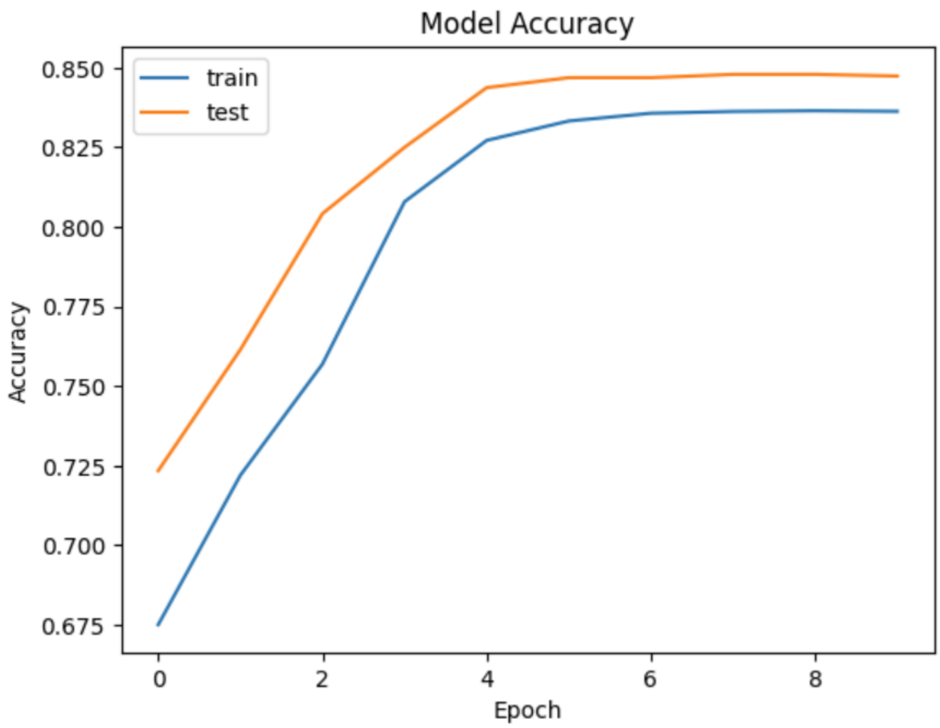

Figure 4.4: Plot of model's accuracy over each epoch with 4x4 images

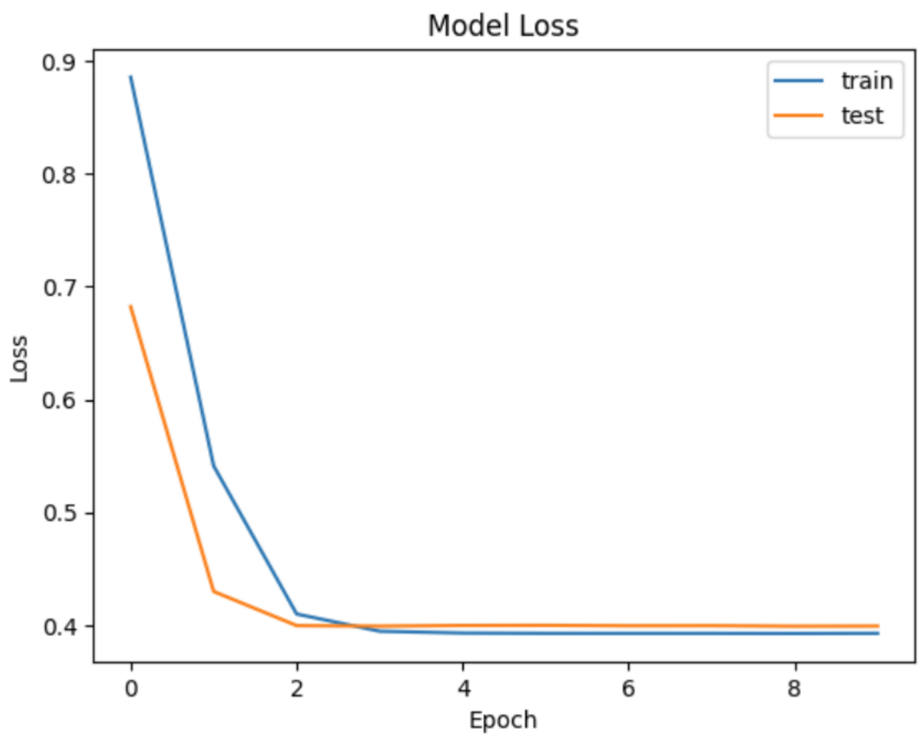

Figure 4.5: Plot of the model's loss over each epoch

mal changes after early epochs as the model stabilizes around its best performance. So, increasing the number of pixels in the data means more features for the model to learn from, leading to the convergence at later epoch, and apparently a better accuracy of nearly 0.85.

Similarly, a plot capturing the loss change over epochs for the first case of 2x2 images has been generated as well, and the picture is almost identical, where the loss does not change after the second epoch. See the Figure 4.5

Furthermore, the models were compared with their classical counterpart - the Convolutional Neural Network with fully connected layers. This classical model is trained on the full-resolution images, meaning they were not downscaled such as the quantum models, so all of the 28x28 images were used in the process. The training on the classical model was performed with a batch size of 64 and 50 epochs, and that led to the best results in terms of accuracy score reaching 0.99 (See the Figure 4.6) and the loss reaching 0.05.

Apparently, the Confusion Matrix yielded better results as well compared to quan-

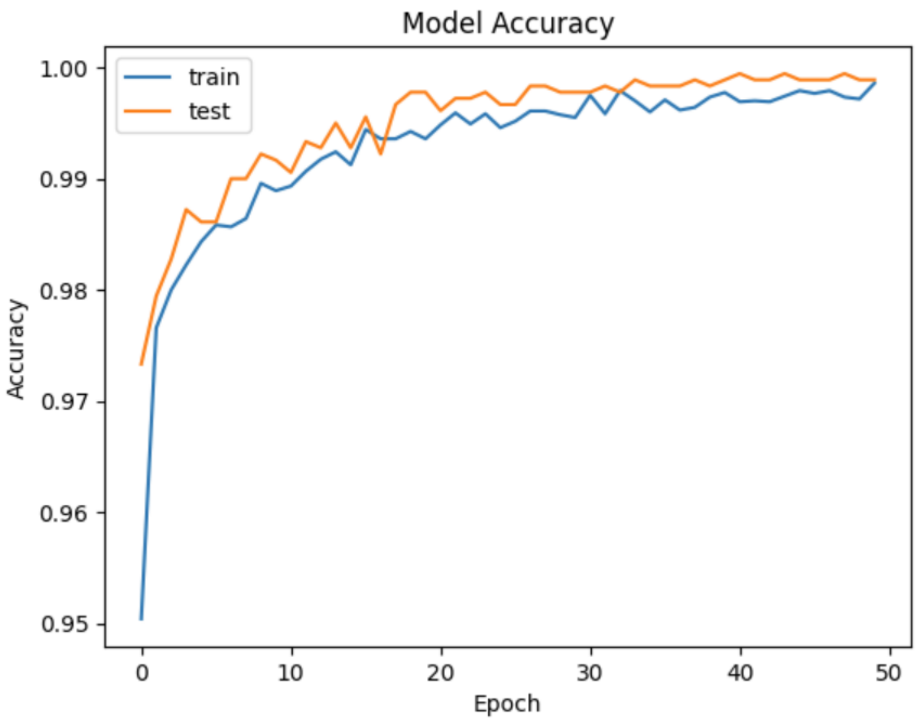

Figure 4.6: Classical model's accuracy over each epoch with full-resolution images

tum models. See the Figure 4.7.

However, it would be much more fair to train the classical model on downscaled images to 2x2. Obviously, the architecture on the CNN had to be adjusted, since with only 2x2 input, applying multiple convolutional layers with filters does not make much sense, as the spatial dimensions of the input are already very small. The parameters for the training were slightly adjusted to accommodate smaller input, with batch size set to 32, and epochs to 20. Needless to say, such a model learnt almost immediately converging to its optimum after a few epochs, reaching an accuracy score of 0.87 and and the loss of around 0.31.

The following Table 4.1 shows the accuracy scores along with the loss estimated on the testing dataset across the built models, achieved by the end of their training.

Model	**Accuracy**	**Loss**
QNN (4x4 images)	0.84	0.36
QCNN (28x28 images)	0.95	0.17
Classical CNN (4x4 images)	0.92	0.19
Classical CNN (28x28 images)	0.99	0.05

Table 4.1: Comparison of Accuracy and Loss between Quantum Model and Classical CNN

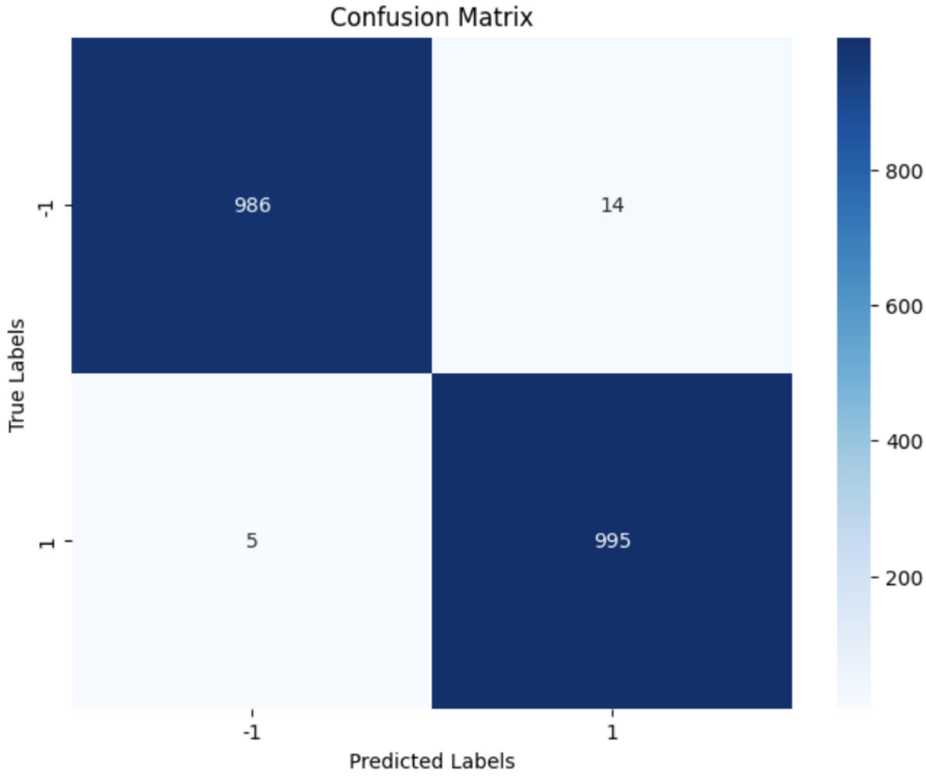

Figure 4.7: Confusion Matrix for the classical model

The experimental results indicate that classical models continue to outperform quantum models in the domain of image classification, though quantum models demonstrate competitive performance. In general, better results might be achieved if the data were downscaled to a lesser degree or not downscaled at all. However, scaling beyond 16 pixels led to system crashes due to hardware limitations, primarily related to current RAM capacity in the simulation environment. While these experiments were conducted in simulation, it is possible that quantum hardware could yield improved results, though hardware access and constraints limited this investigation.

4.2 Discussion

Looking back at the experimental results obtained, Quantum Neural Network (QNN) trained on the downscaled 4x4 images, presented the worst results out of the four models, but the use of different parameters' values, such as the number of epochs, should be taken into consideration due to limited hardware capability. As for the best model presented, which is the Classical CNN trained on full 28x28 images, presented the best results, both in terms of accuracy and loss. However, the quantum models usually took less epochs to converge to the optimal values.

It is acknowledged that more evaluation metrics could be used beside accuracy, such as precision, recall, and F1 scores. Additionally, a confusion matrix could be built in order to identify the weakest points of the classifier by observing the number of False Positives and False Negatives for example. Moreover, it would be useful to compare these built models with other existing benchmark models created by others.

Overall, the results obtained in this study provide supportive evidence that the application of quantum paradigms can potentially surpass classical approaches in terms of the time it could take to converge to optimal solution, or the number of epochs. This suggests that quantum methods, under certain conditions, might offer more efficient solutions for some problems. It is also important to take into consideration the scope of the quantum model presented, given their implementation being done within the simulation frameworks. Quantum hardware could potentially offer considerable speedups.

5
Conclusions

This chapter will summarize the work by providing a comprehensive analysis of the key achievements, identifying areas of strength and those requiring further development, and outlining directions for future research.

5.1 Evaluation of Research Findings

The study began by establishing the reason behind selecting quantum machine learning as a prominent field of research. The introductory chapter outlined the importance of the topic, explained the research theme to the reader, and clearly stated the research objectives and questions.

Following this, the necessary background was provided to position the work within the broader context of machine learning and quantum computing, highlighting the evolution of these fields and their convergence into quantum machine learning (QML). The historical development of QML was briefly introduced, providing the reader with essential insights into how these domains intersect. The chapter concluded by outlining the structure of the study, summarizing the content of each subsequent chapter.

The State of the Art chapter provided a comprehensive review of existing tools and methodologies, including quantum software libraries such as Qiskit, Cirq, and TensorFlow Quantum, which are critical for developing quantum models. It also discussed key advances in quantum hardware and algorithms, giving a detailed review of how these innovations have enabled the application of quantum methods to machine learning.

Building on the reviewed studies, code was developed to replicate and extend the concepts from the literature, applying quantum paradigms such as quantum convolution layers and quantum circuits to images' data. The results, as detailed in the experimental results chapter, demonstrated competitive performance from quantum methods, showing potential improvements over classical approaches in certain aspects.

5.2 Future Research Directions

Future research should address several key challenges and areas of improvement. First, developing quantum algorithms capable of handling noisy quantum data is essential for the continued progress of quantum machine learning. The current limitations of NISQ hardware require further advancements in error correction and noise mitigation techniques to improve the reliability of quantum computations.

Another important direction is the improvement of quantum hardware itself. Enhancing qubit coherence times and qubit connectivity will be crucial for scaling up quantum machine learning models. As quantum processors evolve, so will the ability to solve larger, more complex problems, which are currently beyond the capacity of existing devices presenting some noise and errors to their calculations.

Additionally, future work could explore the application of quantum machine learning to fields such as quantum chemistry, material science, and healthcare, where quantum algorithms have the potential to outperform classical methods. Expanding the scope of quantum algorithms and integrating them with classical models will help closing the gap between theory and practical applications.

Finally, theoretical advancements in areas such as quantum neural networks and quantum kernel methods could lead to more efficient models that are capable of handling high-dimensional datasets and complex machine learning tasks.

References

[1] Bishrelt Khurelsukh. Hybrid quantum convolutional neural networks in tensorflow quantum. In *2022 33rd Irish Signals and Systems Conference (ISSC)*, pages 1–7, 2022.

[2] Nathan Wiebe, Ashish Kapoor, and Krysta Svore. Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning. *arXiv preprint arXiv:1401.2142*, 2014.

[3] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical review letters*, 113(13):130503, 2014.

[4] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

[5] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of statistical physics*, 22:563–591, 1980.

[6] Isaac L Chuang, Neil Gershenfeld, and Mark Kubinec. Experimental implementation of fast quantum searching. *Physical review letters*, 80(15):3408, 1998.

[7] Elizabeth Gibney. Hello quantum world! google publishes landmark quantum supremacy claim. *Nature*, 574(7779):461–463, 2019.

[8] Carlo A Trugenberger. Probabilistic quantum memories. *Physical Review Letters*, 87(6):067901, 2001.

[9] Dan Ventura and Tony Martinez. Quantum associative memory. *Information sciences*, 124(1-4):273–296, 2000.

[10] Rajlaxmi Vilas Bhoite and S L Sreekuttan. Benchmarking quantum algorithms on xanadu, ibm, and google quantum computers. In *2022 International Conference on Trends in Quantum Computing and Emerging Business Technologies (TQCEBT)*, pages 1–5, 2022.

[11] Michael Broughton, Guillaume Verdon, Trevor McCourt, Antonio J Martinez, Jae Hyeon Yoo, Sergei V Isakov, Philip Massey, Ramin Halavati, Murphy Yuezhen Niu, Alexander Zlokapa, et al. Tensorflow quantum: A software framework for quantum machine learning. *arXiv preprint arXiv:2003.02989*, 2020.

[12] David Peral García, Juan Cruz-Benito, and Francisco José García-Peñalvo. Systematic literature review: Quantum machine learning and its applications, 2023.

[13] Eunmi Chae, Joonhee Choi, and Junki Kim. An elementary review on basic principles and developments of qubits for quantum computing. *Nano Convergence*, 11(1):11, 2024.

[14] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[15] Benjamin Schumacher and Michael Westmoreland. *Quantum processes systems, and information*. Cambridge University Press, 2010.

[16] Stanislav Filatov and Marcis Auzinsh. Towards two bloch sphere representation of pure two-qubit states and unitaries. *Entropy*, 26(4):280, 2024.

[17] Eleanor G Rieffel and Wolfgang H Polak. *Quantum computing: A gentle introduction*. MIT press, 2011.

[18] Peter Hu, Yangqiuting Li, Roger SK Mong, and Chandralekha Singh. Student understanding of the bloch sphere. *European Journal of Physics*, 45(2):025705, 2024.

[19] Marco Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, et al. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, 2021.

[20] Ryszard Horodecki, Paweł Horodecki, Michał Horodecki, and Karol Horodecki. Quantum entanglement. *Reviews of modern physics*, 81(2):865, 2009.

[21] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. An introduction to quantum machine learning. *Contemporary Physics*, 56(2):172–185, 2015.

[22] Lili Bai, Zerui Song, Haijie Bao, and Jingqing Jiang. K-means clustering based on improved quantum particle swarm optimization algorithm. In *2021 13th International Conference on Advanced Computational Intelligence (ICACI)*, pages 140–145, 2021.

[23] Vojtěch Havlíček, Antonio D Córcoles, Kristan Temme, Aram W Harrow, Abhinav Kandala, Jerry M Chow, and Jay M Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, 2019.

[24] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.*, 122:040504, Feb 2019.

[25] J. von Neumann. *Mathematical Foundations of Quantum Mechanics*. Princeton University Press, 1932.

[26] Xavier Vasques, Hanhee Paik, and Laura Cif. Application of quantum machine learning using quantum kernel algorithms on multiclass neuron m-type classification. *Scientific Reports*, 13(1):11541, 2023.

[27] Mateusz Slysz, Krzysztof Kurowski, Grzegorz Waligóra, and Jan Węglarz. Exploring the capabilities of quantum support vector machines for image classification on the mnist benchmark. In *International Conference on Computational Science*, pages 193–200. Springer, 2023.

[28] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature physics*, 10(9):631–633, 2014.

[29] Zeheng Wang, Timothy van der Laan, and Muhammad Usman. Quantum kernel principal components analysis for compact readout of chemiresistive sensor arrays, 2024.

[30] B Camino, J Buckeridge, PA Warburton, V Kendon, and SM Woodley. Quantum computing and materials science: A practical guide to applying quantum annealing to the configurational analysis of materials. *Journal of Applied Physics*, 133(22), 2023.

[31] Chengqiang Lu, Qi Liu, Chao Wang, Zhenya Huang, Peize Lin, and Lixin He. Molecular property prediction: A multilevel quantum interactions modeling perspective, 2019.

[32] Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I Latorre. Data re-uploading for a universal quantum classifier. *Quantum*, 4:226, 2020.

[33] Shervin Minaee, Mehdi Minaei, and Amirali Abdolrashidi. Deep-emotion: Facial expression recognition using attentional convolutional network. *Sensors*, 21(9):3046, 2021.

[34] Aaron Somoroff, Quentin Ficheux, Raymond A. Mencia, Haonan Xiong, Roman Kuzmin, and Vladimir E. Manucharyan. Millisecond coherence in a superconducting qubit. *Phys. Rev. Lett.*, 130:267001, Jun 2023.

[35] Xinyuan You, Ziwen Huang, Ugur Alyanak, Alexander Romanenko, Anna Grassellino, and Shaojiang Zhu. Stabilizing and improving qubit coherence by engineering the noise spectrum of two-level systems. *Physical Review Applied*, 18(4):044026, 2022.

[36] Austin G Fowler, Matteo Mariantoni, John M Martinis, and Andrew N Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A—Atomic, Molecular, and Optical Physics*, 86(3):032324, 2012.

[37] Daniel Gottesman. *Stabilizer codes and quantum error correction. Caltech Ph. D.* PhD thesis, Thesis, eprint: quant-ph/9705052, 1997.

[38] Vinay Tripathi, Noah Goss, Arian Vezvaee, Long B Nguyen, Irfan Siddiqi, and Daniel A Lidar. Qudit dynamical decoupling on a superconducting quantum processor. *arXiv preprint arXiv:2407.04893*, 2024.

[39] Filip B Maciejewski, Zoltán Zimborás, and Michał Oszmaniec. Mitigation of readout noise in near-term quantum devices by classical post-processing based on detector tomography. *Quantum*, 4:257, 2020.

[40] Adam Holmes, Sonika Johri, Gian Giacomo Guerreschi, James S Clarke, and A Y Matsuura. Impact of qubit connectivity on quantum algorithm performance. *Quantum Science and Technology*, 5(2):025009, March 2020.

[41] Avinash Chalumuri, Raghavendra Kune, and BS Manoj. A hybrid classical-quantum approach for multi-class classification. *Quantum Information Processing*, 20(3):119, 2021.

[42] Chen-Yu Liu, En-Jui Kuo, Chu-Hsuan Abraham Lin, Jason Gemsun Young, Yeong-Jar Chang, Min-Hsiu Hsieh, and Hsi-Sheng Goan. Quantum-train: Rethinking hybrid quantum-classical machine learning in the model compression perspective. *arXiv preprint arXiv:2405.11304*, 2024.

[43] Sebastian Raubitzek and Kevin Mallinger. On the applicability of quantum machine learning. *Entropy*, 25(7):992, 2023.

[44] Hermann Fürntratt, Paul Schnabl, Florian Krebs, Roland Unterberger, and Herwig Zeiner. Towards higher abstraction levels in quantum computing. In Flavia Monti, Pierluigi Plebani, Naouel Moha, Hye-young Paik, Johanna Barzen, Gowri Ramachandran, Devis Bianchini, Damian A. Tamburri, and Massimo Mecella, editors, *Service-Oriented Computing – ICSOC 2023 Workshops*, pages 162–173, Singapore, 2024. Springer Nature Singapore.

[45] Liu Hailing, Zhang Jie, Qin Sujuan, and Gao Fei. Quantum algorithm for soft margin support vector machine with hinge loss function. *Journal of China University of Posts and Telecommunications (English)*, 29(4):32, 2022.

[46] S-A Guo, Y-K Wu, J Ye, L Zhang, W-Q Lian, R Yao, Y Wang, R-Y Yan, Y-J Yi, Y-L Xu, et al. A site-resolved two-dimensional quantum simulator with hundreds of trapped ions. *Nature*, pages 1–6, 2024.

[47] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

[48] Marco Wiedmann, Marc Hölle, Maniraman Periyasamy, Nico Meyer, Christian Ufrecht, Daniel D. Scherer, Axel Plinge, and Christopher Mutschler. An empirical comparison of optimizers for quantum machine learning with spsa-based gradients. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, September 2023.

Appendices

.1 Code for Quantum Convolutional Neural Network Implementation

```python
import cirq
import tensorflow as tf
from qconv2d_drc import QConv2D_DRC
from reuploading_pqc import ReUploadingPQC

# Define number of qubits
n_qubits = 2

# Construct the observable for the quantum classifier
qubits = cirq.GridQubit.rect(1, n_qubits)
Z_0 = cirq.PauliString(cirq.Z(qubits[0]))
I_0 = cirq.PauliString(cirq.I(qubits[0]))
Z_1 = cirq.PauliString(cirq.Z(qubits[1]))
I_1 = cirq.PauliString(cirq.I(qubits[1]))
observables = [0.5*Z_0 + 0.5*I_0, -0.5*Z_1 + 0.5*I_1]

# Full size MNIST images as input with shape (28, 28, 1)
input_tensor = tf.keras.Input(shape=(28, 28, 1), dtype=tf.dtypes.float32,
    name='input')

#settings for the quantum convolutional layers
drc_settings = {
    "n_qubits": [1],
    "n_layers": [1],
    "use_ent": [True],
    "use_terminal_ent": [True]
}

# Define the quantum convolution layers
qconv_1 = QConv2D_DRC(1, (3,3), (2,2), drc_settings, 1, padding="valid").
    call(input_tensor)
qconv_2 = QConv2D_DRC(1, (3,3), (2,2), drc_settings, 2, padding="valid").
    call(qconv_1)

# Max pooling layer to reduce dimensionality
pool = tf.keras.layers.MaxPooling2D(pool_size=(2, 2), strides=(2,2),
    padding="valid")(qconv_2)

# Flatten the tensor
flat = tf.keras.layers.Flatten()(pool)

# Apply the Quantum Classifier layer
quantum_classifier = ReUploadingPQC(n_qubits, 1, flat.shape[-1],
    use_entanglement=True, use_terminal_entanglement=True, observables=
    observables)(flat)

# Final output layer with sigmoid activation
output = tf.keras.layers.Activation('sigmoid')(quantum_classifier)

# Create the full QCNN model
model = tf.keras.Model(inputs=[input_tensor], outputs=output)
```

Figure 1: The code in Python for implementing the proposed QCNN