Quantum Machine Learning for Big Data Analytics

A thesis submitted

in partial fulfillment for the award of the degree of

Doctor of Philosophy

by

Avinash Chalumuri

Department of Avionics
Indian Institute of Space Science and Technology
Thiruvananthapuram, India

October 2022

Certificate

This is to certify that the thesis titled ***Quantum Machine Learning for Big Data Analytics*** submitted by **Avinash Chalumuri**, to the Indian Institute of Space Science and Technology, Thiruvananthapuram, in partial fulfillment for the award of the degree of **Doctor of Philosophy** is a bona fide record of the original work carried out by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. K. Raghavendra
Thesis Advisor–ISRO
Scientist/Engineer 'SG'
Section Head, HPCDS
ADRIN, ISRO.

Dr. B. S. Manoj
Thesis Advisor–IIST
Professor
Department of Avionics, IIST.

Dr. Deepak Mishra
Professor and Head
Department of Avionics, IIST.

Place: IIST, Thiruvananthapuram
Date: October 2022

Declaration

I declare that this thesis titled ***Quantum Machine Learning for Big Data Analytics*** submitted in partial fulfillment for the award of the degree of **Doctor of Philosophy** is a record of the original work carried out by me under the supervision of **Dr. K. Raghavendra** and **Dr. B. S. Manoj**, and has not formed the basis for the award of any degree, diploma, associateship, fellowship, or other titles in this or any other Institution or University of higher learning. In keeping with the ethical practice in reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited.

Place: Thiruvananthapuram **Avinash Chalumuri**

Date: October 2022 (SC18D045)

This thesis is dedicated to

My Parents, Teachers, and Professors

For being the role models and the source of enlightenment throughout my life

My Family

For their patience, unconditional love, and support

"Fall in love with some activity, and do it! Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough. Work as hard and as much as you want to on the things you like to do the best. Don't think about what you want to be, but what you want to do. Keep up some kind of a minimum with other things so that society doesn't stop you from doing anything at all."

RICHARD P. FEYNMAN

(May 11, 1918 – February 15, 1988)

Theoretical Physicist and Noble Prize Winner

Acknowledgements

"Great teachers, in any field , do far more than convey information;
They pass along something of themselves."
– PETER BUFFETT

Our philosophies are mostly a product of what we practice and whom we follow as our role models and teachers. I am so fortunate to have learned from some truly *Great teachers* who taught me the joy of life-long learning that helped me during my PhD. This thesis work owes its existence to many persons without whose help it would not have been possible to complete.

First, I express my sincere gratitude to my PhD advisor *Dr. B. S. Manoj*, Professor, Department of Avionics, Indian Institute of Space Science and Technology (IIST), for suggesting the new and exciting interdisciplinary research area. He walked me through all the challenges I faced throughout my research. His tremendous research experience, indispensable comments, and critical insights helped me throughout the thesis. He made me realize that PhD is the start of a journey as a responsible researcher. He has also been a philosopher guide and helped me improve my overall attitude towards life. More importantly, his *patience in listening and answering* all my questions and concerns throughout these years is impressive. I consider him as one of the *Great teachers* with a kind heart I have ever met in my life. He is a true inspiration and his support throughout my PhD is fantastic.

I am very grateful to my dear supervisor from the Indian Space Research Organization (ISRO), *Dr. K. Raghavendra*, Scientist Engineer 'SG', Head, High-Performance Computing and Drones (HPCDS), ADRIN, ISRO. He always prioritized our research discussion and was readily available for discussions, *even late nights*. He has been a scientist guide with a good grasp of the subject and an excellent researcher. I will always admire him for his time and support throughout the thesis. His vast knowledge of *spatial data analytics* greatly helped in our research work. I am immensely thankful for his time and efforts in improving my thinking perspective as a researcher.

I am so grateful to the members of my Doctoral Committee, *Dr. Deepak Mishra*, Professor and Head, Department of Avionics, IIST, *Dr. Rama Rao Nidamanuri*, Professor and Head, Department of Earth and Space Sciences, IIST, *Dr. S. Sumitra*, Associate Professor,

Department of Mathematics, and *Dr. B. S. Vineeth*, Department of Avionics, IIST, for reviewing my work and providing timely feedback and suggestions throughout the PhD work. I extend my gratitude to *Dr. M. V. Panduranga Rao*, Associate Professor, Department of Computer Science and Engineering, Indian Institute of Technology (IIT), Hyderabad, *Dr. Kaushik Mukherjee*, Associate Professor, Department of Mathematics, IIST, *Dr. Solomon Ivan*, Associate Professor, Department of Physics, IIST, and *Dr. Ashok Kumar*, Assistant Professor, Department of Physics, IIST, for the valuable discussions during the initial days of my research.

It was an eye-opening experience to discuss with *Dr. Maria Schuld*, Senior Researcher, Xanadu Quantum Technologies (XQT), Canada, *Mr. Mark Fingerhuth*, Co-Founder, Head R&D, ProteinQure, Canada, and *Mr. Mehdi Bozzo-Rey*, Director, Business Development at Quantum Algorithms Institute, Canada, during the initial days of my PhD that helped me to choose to work on Quantum Machine Learning (QML). I also thank *Mr. Nathan Killoran*, Head of Software, Algorithms & QML, XQT, *Mr. Josh Izaac*, Computational Quantum Physicist & Lead Developer, *PennyLane*, XQT, *Ms. Olivia Di Matteo*, Former Quantum Computing Educator & Researcher, XQT, and *Ms. Sai Lakshmi*, Machine Learning Engineer, Tata Consultancy Services, Hyderabad, and *Mr. Kannan Suresh,* Scientist Engineer, ISRO Inertial Systems Unit, Trivandrum, for their helpful suggestions and discussions that immensely helped me to perform machine learning experiments using *TensorFlow*, quantum simulations, and experiments using *PennyLane*.

My sincere thanks to *Prof. Kapila Rohan Attele*, Department Chair, Mathematics & Computer Science, Chicago State University, USA, and *Dr. Pascal Paschos*, Sr. Computational Scientist, University of Chicago, USA, for their invaluable timely support and discussions. I am also thankful to my professors at Jawaharlal Nehru Technological University, Kakinada (JNTUK), India, especially, *Prof. J. V. R. Murthy*, Director, Incubation & IPR, and *Prof. M. H. M. Krishna Prasad*, Principal, University College of Engineering, for their kindness and never-ending moral support.

I express my sincere thanks to ISRO and IIST for giving me this tremendous opportunity and resources to do my PhD. I sincerely thank the All India Council for Technical Education and the Indian National Academy of Engineers, and Technology IHub Foundation, Indian Institute of Technology Palakkad for funding my research in parts. I am so grateful to the management and administration of Gayatri Vidya Parishad College of Engineering (GVPCE), Visakhapatnam, India, for their financial support and their confidence in me. Special thanks to the Department of Computer Science and Engineering, GVPCE.

I also thank, IIT Palakkad Technology IHub Foundation (IPTIF) for supporting me through the Doctoral Fellowship Grant (Grant Number: IPTIF/HRD/DF/032).

Association with intellectual people helps us improve, shape habits, and form behavior. I feel lucky to be surrounded by such intellectual people in our Systems and Networks Lab (SysNet Lab), IIST. It was an immense pleasure to be associated with my senior research scholar *Dr. Sarath Babu*, currently, Research Scientist at the Department of Electrical and Computer Engineering (ECpE), Iowa State University (ISU), USA and I thank him for the technical and non-technical discussions. He provided me tips to *think and code like a computer scientist* and inspired me to develop the *habit of book reading*. I also thank *Dr. Abhishek Chakraborty* my senior research scholar and currently Postdoctoral Scholar, Qualcomm Institute, University of California, San Diego, USA for helping me learn *how to keep cool during PhD*. I extend my respect and gratitude to *Dr. Prescilla Koshy*, former Postdoctoral Fellow of SysNet Lab, for providing me insights on her research on the Internet of Things and BlockChain Technology. I cannot express in words the companionship provided by *Mr. Debabrata Dalai*, my colleague at SysNet Lab and a coolest person, which made my life at the lab more happy and enjoyable. I also thank, *Mrs. Divya R. S.*, lab staff of SysNet Lab, for her sisterly support and providing me a homely atmosphere at SysNet Lab.

It was a great experience to work with *Mr. Sathwik Reddy* and *Mr. Saikiran Chalumuru* on *Quantum Big Data Analytics*, and *Mr. Mayur Pawar Sunil* and *Mr. Sanghpriy Gautam* on *Quantum Internet*, during their summer internship and final projects at SysNet Lab. I also appreciate my continuing association with *Mr. Indranil Ghosh*, a former summer intern at SysNet Lab, currently a PhD student at Massey University, New Zealand.

It was a joyful experience to share knowledge with undergraduate and postgraduate students at SysNet Lab, as part of my teaching assistantship and learn many new things. I also thank the IEEE Student Branch of IIST for giving me the opportunity to serve as an executive member and volunteer for the several activities organized inside and outside the campus.

Happiness is having real friends at your side and being surrounded by love. Life at IIST would have been nothing for me without *Mr. Pramod Martha*, my PhD colleague and best buddy. Beyond words are the happy experiences that I got from my friends at IIST, especially *Mr. Debabrata Dalai, Mrs. Sreekala K., Mrs. Anjitha R. G., Mr. Krishnan Unni R. A., Ms. Bhasha Sathyan, Mr. Elangovan K., Mr. Akshay R. S. R, Mr. Jyothirmoy Dey, Ms. Ann Mary, Mr. Govind Kumar Sharma, Mrs. Tina B. S., Mr. Guru Krishna Tej, Mr. Narendra Singh, Mr. Gourahari Nayak, Mr. Danish Handa, Mrs. Ashwathy P. T., Mr. Dayal G., Mr. Vibin Jose, Mr. Ajin,*

Mr. Renjith Thomas, Mr. Sarath K.P., Mr. Sajith V. S., Mr. Sreehari B Nair, Ms. Jayati Vijaywargiya, Mrs. Gopika R., Mrs. Elizabeth George, Mr. Jeeva B., and Mr. Nitesh Kumar Agarwal.

Family is the most important thing in the world. I thank my parents *Dr. Ch. Satya Rao*, PhD in Psychology, and *Smt. Jayalakshmi*, BA in Economics, for their years of daily care, love, encouragement, endless worrying about my well-being, and being the role models of my life. I also thank my sister *Dr. Alekhya Naidu*, PhD in English Literature, for her care and being a lovely sibling. I also thank my in-laws and other family members for their support through the years. Life would be incomplete for me without the unconditional love from my wife *Mrs. Chinoja Avinash*. Her role as a working woman and taking responsibility through the years is priceless. Finally, I thank my adorable daughter *Miss Srinika Chalumuri*, for being my stress buster and driving force to finish my PhD.

Avinash Chalumuri

Abstract

Enormous amount of data is generated globally at an exponential rate in various sectors such as government, agriculture, finance, defense, engineering, and medicine. Due to the recent advancements in technology, a rapid growth rate in data generation and collection of different varieties of data is observed. The exponentially growing data that can be analyzed computationally to reveal patterns, trends, and associations are considered as *big data*. Numerous benefits can be obtained from analyzing such *big data* that helps in decision-making and finding solutions for various real-time problems. Over the years, Artificial Intelligence (AI)/Machine learning (ML) algorithms revolutionized the way of analyzing vast amounts of data for valuable insights and applications. Machine learning emphasizes developing computer programs that can access and learn from data to build a generalized model. Supervised learning is an efficient learning technique to build ML models using "labeled" training data to predict the output. The model parameters such as weights are optimized to produce the desired result during the training process. Also, hyperparameters are tuned during the training phase to improve the accuracy of the model.

Due to the rapid speeds at which the data grows, big data processing using ML algorithms is an area of concern. ML algorithms face many challenges in dealing with big data, including computational resources, model selection, optimizing the parameters, and increasing the algorithm accuracy. Deep learning algorithms such as convolutional neural networks (CNNs) require huge computational facilities to process very-large datasets for supervised learning. Also, difficulty in training such deep learning models increases with the large and complex datasets. In this context, Quantum Machine Learning (QML) is emerging as a field of interest in computer science with the intersection of quantum computing and machine learning. Quantum computers are fundamentally different from classical computers as principles of quantum mechanics are used for information processing. Hence, quantum computing techniques can solve specific computational problems difficult for a classical computer. Also, quantum computing can enhance classical machine learning techniques as powerful quantum tools exist for linear algebra. As linear algebra is the basis for machine learning, quantum computing offers practical performance advantages over classical approaches. Hence, there is a necessity to explore the area of quantum machine learning, to advance the existing machine learning techniques.

In this thesis, we propose and study quantum machine learning techniques to enhance supervised learning of classical data. The ability of quantum computational approaches to improve classical machine learning algorithms is explored. In particular, the proposal and study of hybrid quantum-classical machine learning methods to solve supervised learning problems are addressed in the thesis. As a first step towards quantum machine learning, an artificial neural network (ANN) using quantum bits (qubits) as artificial neurons is proposed for binary classification. The quantum computing approach for ANN (QC ANN) aims to develop a clear understanding of the impact of qubits in training an artificial neural network for binary classification of numerical data. Further, the work is extended to design a quantum multi-class classifier (QMCC) for multi-class classification. QMCC is intended to be a quantum circuit with parameterized quantum layers for machine learning. For QMCC, an encoding method for state preparation to input the data into qubits is also proposed. QMCC, in total, is a parameterized quantum circuit with multiple trainable layers for the multi-class classification of numerical data. The experimental results show that the proposed techniques perform binary and multi-class classification with good accuracy.

In the second phase of our research, we proposed QML frameworks for processing spatial big data analytics tasks. At first, a three-layered hybrid quantum-classical (HybridQC) architecture is proposed for satellite image scene classification. The proposed model consists of the following steps: (i) a classical preprocessing step, (ii) a quantum processing step to extract image representations, and (iii) a deep neural network built with the extracted image representations. Our experimental results show that the total parameters for training a deep neural network reduced with the proposed approach. Next, a data augmentation technique is proposed using a quantum circuit that can be used to enhance datasets during training deep neural networks. Also, a hybrid model with a combination of vanilla convolutional neural network (CNN) and quantum processing is proposed for image scene classification. Finally, we propose quantum processing techniques to process synthetic aperture radar (SAR) images for deep learning.

We also discuss the performance advantages of the hybrid quantum-classical approach over classical computation on both numerical data and spatial data. Our work show that quantum computational techniques can enhance classical machine learning by reducing the trainable parameters of the models and, as a result, improvement in the classification accuracy can also be observed. Finally, the thesis is concluded with the future scope of QML algorithms in solving complex machine learning problems.

Contents

List of Figures **xvii**

List of Tables **xxi**

Abbreviations **xxiii**

Nomenclature **xxv**

1 Introduction **1**

1.1 Quantum Computing: Preliminaries 3

1.2 Motivation for the Research Work 11

1.3 The Focus and Contributions of the Thesis 13

1.4 Organization of the Thesis 14

1.5 Summary 16

2 Quantum Machine Learning for Big Data Processing **17**

2.1 Scope of Data Processing 18

2.2 Quantum Machine Learning Framework 19

2.3 Summary 24

3 Related Work **25**

3.1 Quantum Machine Learning Methodologies 25

3.2 Hybrid QML Approach 27

3.3 Summary 31

4 Hybrid QML Models for Supervised Learning **32**

4.1 Quantum Computing for Artificial Neural Network 32

4.2 Quantum Multi-Class Classifier 36

4.3 Experiments on Numerical Data for Classification 42

4.4 Performance Evaluation of QC ANN 43
4.5 Performance Evaluation of QMCC 47
4.6 Major Observations 56
4.7 Summary 57

5 Hybrid QML Architectures for Spatial Data Analytics 58
5.1 Quantum-Enhanced Deep Neural Networks 59
5.2 Hybrid Quantum-Classical Convolutional Neural Network 67
5.3 Experiments on Spatial Data for Classification 71
5.4 Performance Analysis of HybridQC 72
5.5 Performance Analysis of HQCNN 81
5.6 General Observations 85
5.7 Summary 86

6 Quantum Processing of SAR images for Deep Learning 87
6.1 Quantum Processing Techniques for SAR Images 88
6.2 Performance Analysis of DL models using QSAR Images 92
6.3 Summary 93

7 Conclusions and Future Directions 95
7.1 Future Directions 96

Bibliography 97

List of Publications 113

List of Figures

1.1 Representation of a classical bit and a quantum bit. 4
1.2 Superposition state of a qubit after Hadamard operation. 6
1.3 Representation of R_y gate operation. 6
1.4 Change in state of qubits after *CNOT* operation. 8
1.5 Illustration of a quantum circuit. 9
1.6 Measurement notation in a quantum circuit. 10
1.7 Quantum circuit for entanglement. 10

2.1 Data processing and computational techniques. 18
2.2 Block diagram of QML approach. 19
2.3 Three embedding techniques. 20
2.4 A hybrid classical-quantum circuit model. 23

3.1 Present state and future goal of research on hybrid and QML algorithms. . . 26

4.1 Classical artificial neural network. 33
4.2 Quantum circuit with qubits as nodes of ANN. 34
4.3 Quantum circuit for QMCC. 37
4.4 Generalized architecture of QMCC. 38
4.5 State preparation for QMCC. 39
4.6 Cost comparison of QC ANN using NM. 43
4.7 Cost comparison of QC ANN using Adam. 44
4.8 Training accuracy comparison of QC ANN using NM. 44
4.9 Training accuracy comparison of QC ANN using Adam. 45
4.10 Validation accuracy comparison of QC ANN using NM. 45
4.11 Validation accuracy comparison of QC ANN using Adam. 45
4.12 Experimental result of qubit encoding on IBMQX. 48

4.13 Cost plot for QMCC using angle embedding. 49
4.14 Training accuracy of QMCC using angle embedding. 50
4.15 Test accuracy of QMCC using angle embedding. 50
4.16 Illustration of amplitude embedding model. 51
4.17 Generalized architecture of AEM with n-layers. 51
4.18 Cost comparison of QMCC and AEM on Iris dataset. 53
4.19 Training accuracy of QMCC and AEM on Iris dataset. 53
4.20 Test accuracy of QMCC and AEM on Iris dataset. 53
4.21 Cost comparison of QMCC and AEM on BNA dataset. 54
4.22 Training accuracy of QMCC and AEM on BNA dataset. 54
4.23 Test accuracy of QMCC and AEM on BNA dataset. 54
4.24 Cost comparison of QMCC and AEM on WIL dataset. 55
4.25 Training accuracy of QMCC and AEM on WIL dataset. 55
4.26 Test accuracy of QMCC and AEM on WIL dataset. 55

5.1 Three-level hybrid quantum-classical architecture. 60
5.2 Illustration of quantum state mapping. 61
5.3 Quantum circuit for an input of 2^n values. 62
5.4 Quantum circuit for an input of 2^2 values. 63
5.5 Illustration of amplitude encoding. 63
5.6 Illustration for the architecture of a classical deep learning model. 66
5.7 Quantum circuit for data augmentation. 68
5.8 Illustration of computational layers in HQCNN. 70
5.9 Examples of satellite images. 71
5.10 Quantum measurement-based feature extraction on UCM. 73
5.11 Quantum measurement-based feature extraction on AID. 73
5.12 Quantum measurement-based feature extraction on NWPU-RESISC45. . . . 74
5.13 Grayscale values of the image used to experiment on *ibm_santiago*. 75
5.14 Quantum circuit with basic gates on *ibm_santiago*. 77
5.15 Illustration of parameters of a neural network. 78
5.16 Accuracy of HybridQC on UCM. 79
5.17 Accuracy of HybridQC on AID. 80
5.18 Accuracy of HybridQC on NWPU-RESISC45 dataset. 80
5.19 Quantum circuit for data augmentation on *ibm_santiago*. 83
5.20 Test accuracy comparison of the models. 84

5.21 Identified advantages of the proposed hybrid approaches. 85

6.1 Comparison of SAR and OPT images from Google Earth Engine. 88
6.2 Illustration of QPT-1. 89
6.3 Illustration of quantum circuit for QPT-1. 89
6.4 QSAR images using QPT-1. 89
6.5 Illustration of QPT-2. 90
6.6 Illustration of quantum circuit for QPT-2. 90
6.7 QSAR images using QPT-2. 90
6.8 SAR images from OpenSARUrban dataset. 92
6.9 Quantum circuit for QPT-1 on ibm_santiago. 93
6.10 Quantum circuit for QPT-2 on ibm_santiago. 94

List of Tables

1.1 Different types of data and characteristics 2

3.1 Different types of data and characteristics 31

4.1 Hyperparamters for training QC ANN 43
4.2 Accuracy of Classical ANN and QC ANN 46
4.3 Comparison of QMCC and AEM 52

5.1 Details of fully connected deep neural networks 66
5.2 Details of quantum measurement-based features extracted 67
5.3 Configuration details of *ibm_santiago* quantum computer 76
5.4 Comparison of accuracy and trainable parameters for HybridQC 79
5.5 Traning and validation accuracy of HQCNN 82
5.6 Experimental details for classification of 3 class labels 83

6.1 Quality metrics comparison of QPTs with original SAR images 92
6.2 Accuracy (%) of deep learning models on QPTs 93

Abbreviations

Adam	Adaptive momentum
AEM	Amplitude Embedding Method
AI	Artificial Intelligence
BAM	Binary Alignment Map
BNA	Bank Note Authentication
CDN	Content Delivery Network
CMOS	Complimentary Metal Oxide Semiconductor
CNOT	Controlled NOT
CSV	Comma Seperated Values
CTN	Cadterns Sloper Format
DL	Deep Learning
ECW	Enhanced Compression Wavelet
EPR	Einstein–Podolsky–Rosen
FASTA	Fast Analysis of Sequences Toolbox - All
GB	Giga Bytes
GIS	Geographic Information System
HDFS	Hadoop Distributed File System
HHL	Harrow–Hassidim–Lloyd
HQCNN	Hybrid Quantum-classical Convolutional Neural Network
HybridQC	Hybrid Quantum-Classical
JPEG	Joint Photographic Experts Group
JSON	Java Script Object Notation
KB	Kilo Bytes
MP4	MPEG-4_Part_14
MPEG	Moving Picture Experts Group
NISQ	Noisy Intermediate-Scale Quantum
NLP	Natural Language Processing

NM	Nesterov Momentum
NOSQL	Not Only SQL
OPT	Optical
QC ANN	Quantum Computing apporach for Artificial Neural Network
QDA	Quantum Circuit for Data Augmentation
QMCC	Quantum Multi-Class Classifier
QML	Quantum Machine Learning
QPT	Quantum Processing Technique
QPU	Quantum Processing Unit
QSAR	Quantum processed SAR
QUBIT	Quantum Bit
SAR	Synthetic Aperture Radar
SGD	Stochastic Gradient Descent
SHP	Shape File
SHX	Shape File Index
SQL	Structured Query Language
WAV	Waveform Audio
WBCD	Wisconsin Breast Cancer Dataset
WIL	Wireless Indoor Localization
ZB	Zettabytes

Nomenclature

$\langle \, \vert \, \rangle$	Dirac Notation of bra-ket
$\langle \, \vert$	Bra symbol
$\vert \, \rangle$	Ket symbol
$\vert 0 \rangle$	Read as ket-zero
$\vert 1 \rangle$	Read as ket-one
ψ	Symbol psi
$\vert \psi \rangle$	Read as ket-psi
$\langle \psi \vert$	Read as bra-psi
$\otimes$	Tensor product
O	Hermitian operator
U	Unitary operation
$\mathbb{C}$	Set of complex numbers
$\vert \, \vert$	Modulus operator
θ	Rotational angle
U_θ	Parameterized quantum operation
σ_z	Pauli-Z operation
R_x	Rotational gate around X-axis
R_y	Rotational gate around Y-axis
R_z	Rotational gate around Z-axis
A^T	Transpose operation on matrix A
v	Classical data value
$\mathcal{D}$	N-dimensional numerical dataset
$\mathbf{x}^d$	d^{th} input data item from $\mathcal{D}$
$\mathbf{y}^d$	d^{th} output data item from $\mathcal{D}$
S	Softmax function
exp	expectation value
$\sum$	Summation

Chapter 1

Introduction

"Data is the new science. Big Data holds the answers."

– PAT GELSINGER

"Quantum physics is no longer an abstract theory for specialists.
We must now absolutely include it in our education and also in our culture."

– CLAUDE COHEN-TANNOUDJI

Big data analytics refers to processing datasets to extract valuable insights by discovering valuable patterns in the data. Due to technological advancement, the exponential growth rate in data generation can be seen in every data. International Data Corporation expects the global data of 33 ZB in 2018 to reach 175 Zettabytes by 2025 [1]. Analyzing such big data is crucial for decision-making to gain numerous benefits from the data produced. The ability of existing technologies to process the fast-growing big data scenarios of the future and obtain insights is questionable. Consider an example of Spatial data [2, 3] where an increasing number of affordable satellite services generate big data with an accuracy of the order of centimeters. Such big data can be analyzed and used for the following purposes: to identify land usage, to monitor factors that influence crop yield, to find areas prone to flooding, to see the impact of development and property pricing, to monitor the foot traffic around shopping centers, and to estimate how customers behave.

The term *Big Data* [4] coined by *Roger Mougalas* from *O' Reilly Media* refers to a large set of data (which can be a collection of many data sets) which is almost impossible to manage and process using traditional business intelligence tools. Big data characteristics can be described using the $5V$*'s — Volume, Velocity, Value, Variety, and Veracity*. Huge *volume* of data is generated every day with a unimaginable *velocity*. The *value* of the data can be realized if insights are obtained from it by analyzing the *variety* of data. The quality and trustworthiness of the data is considered as *veracity*. The invention of new technology and

devices led the rapid growth of sources for the data. Statistics [5] show that the estimated rate at which data is created every day in 2022 on average is 2.5 quintillion data bytes.

The rapid generation of data results in a variety of data with different properties useful for analysis. Analyzing the data uncovers various facts and insights in such huge volumes than the amount that can be inferred for making decisions. Table 1.1 lists most commonly generated data used for analytics. The information is collected from internet sources such as Wikipedia. The table also provides information about data types and different properties of data, along with the most widely used analytical tools. In the table, 1 KB refers to 1 Kilo Byte = 1024 Bytes, and 1 GB refers to 1 Giga Byte = 1,048,576 Kilo Bytes.

Table 1.1: Different types of data and characteristics

S.No	Type of Data	Size of Individual Data Item	Storage Locations	File Format (widely used)	Analytics Tools (widely used)
1	Micro Blogging (e.g., Twitter)	140 Characters	Hadoop Clusters and HDFS	CSV	Twitter Analytics
2	Image	KB-GB	Farms of Servers, Amazon CDN, and Amazon S3	JPEG, ECW	Machine Learning/DL Tools
3	Video	KB-GB	Farms of Servers, Amazon CDN, and Amazon S3	MP4	Machine Learning/DL Tools
4	Genomics	GB	AWS Data Centers	BAM, FASTA	Google Genomics
5	Network	GB	Data Centers and Cloud Storage	CSV	Snaplytics, Google Analytics
6	Spatial	KB-GB	Data Centers and Cloud Storage	SHP, SHX, and JSON	GIS Software, R and Hadoop
7	Biomedical	KB-GB	Data Centers and Cloud Storage	CTN, CSV, and JSON	Hadoop and R
8	Literature	KB-GB	Data Centers and Cloud Storage	NOSQL	Machine Learning/NLP Tools
9	Voice	KB-GB	Data Centers and Cloud Storage	WAV	Machine Learning/NLP Tools

Over the years, Artificial Intelligence (AI)/Machine learning (ML) techniques gained prominence in dealing with big data. Machine learning algorithms [6] removed a lot of obstacles in big data processing and turning the data into information for successful decision making. Apart from these wide applications, the challenge is handling huge amounts of data and producing useful insights. Existing data analytics processes find difficulty in han-

dling large amounts of data. Quantum computing has a better ability to deal with the data in exponential scaling in terms of data representation and speed. Quantum computers are the quantum-mechanical systems that represent data in quantum bits or qubits. Qubits are different from classical bits in terms of data representation and operations. Hence, quantum computing is becoming prominent in solving difficult problems for a classical computer to solve. In the following sections, we explain the preliminaries of quantum computing, related to our research work. We primarily discuss the basics of quantum computing related to information processing using qubits.

1.1 Quantum Computing: Preliminaries

In 1965, Gordon E. Moore, co-founder of Intel Inc., stated: "The number of transistors incorporated in a chip will approximately double every 24 months" [7]. However, in recent years, Moore's law is slowing down due to the difficulty in cost-effectively producing miniature-sized transistors [7]. Also, as the size of the transistors become small, there exist the possibility of electrical leakage, and further, the chips generate heat. Due to the heat, there is an increase in the cost of cooling the computational systems. The size of the transistors becomes so small that the materials start to behave in a quantum mechanical way. Hence quantum computing became prominent, and many companies such as IBM [8] and Google [9] started manufacturing quantum processors with materials that behave in a quantum mechanical way. Quantum computers operate fundamentally differently and are suitable to solve complex optimization problems that cannot be solved on a classical computer [10]. Quantum computers can also deal with big data as they provide exponential scaling [11]. Hence, quantum computational techniques can be used to enhance machine learning algorithms.

A Quantum computer is a complex computational device that works on the principles of quantum mechanics, quantum information theory, and computer science [12]. Hence, quantum computers are fundamentally different from classical computers. As quantum mechanics is considered the basis of the physical universe, a classical computer can also be described by quantum mechanics. However, quantum mechanical properties are not used by a classical computer for information processing. In 1981, Richard P. Feynman, in a physics lecture, stated that: "*Nature is not classical, and for the simulation of nature, quantum mechanical computation systems are necessary.*" [13] Quantum computers use specific quantum mechanical properties such as superposition and entanglement to perform computations.

Figure 1.1: Representation of a classical bit and a quantum bit.

1.1.1 Quantum Computer vs Classical Computer

The following are the fundamental differences between quantum computing and classical computing:

1. Quantum computer works on qubits, and a qubit is a two-level quantum-mechanical system represented using Dirac notation (Bra-ket). The state of a qubit can be represented as $|0\rangle$, $|1\rangle$ or a linear combination of both states, also called superposition. In superposition, the state of a qubit is indeterminate and can emerge into one of the two definite states only after measurement. In classical computers, classical bits are used to represent binary digits 0 and 1. Classical bits represent the voltage levels ON or OFF, and hence, there are no intermediate states in classical computation. Whereas in quantum computation, a qubit can take any intermediate state between $|0\rangle$ and $|1\rangle$ as a linear combination of both the states as shown in Fig. 1.1.

2. Calculation power of a quantum computer increases exponentially. As the number of qubits increases, more linear combinations can be represented using multiple qubits. In a classical computer, computational power increases linearly as the number of transistors on a chip increases.

3. Quantum computers require specialized hardware, where information is processed using qubits. Qubits are manufactured using subatomic particles (or quantum particles) that can exist in two discrete states when measured. In comparison, a classical computer uses hardware made of CMOS circuits.

4. Quantum computers consist of one or more quantum processing units (QPUs) that are inherently parallel due to superposition. Hence, QPUs are used to solve specific

problems and operations that can run in parallel. However, general-purpose tasks such as writing a word document playing multimedia files cannot be performed on a quantum computer. Hence, quantum computers are not a replacement for classical computers as they are not universally faster.

5. Qubits in a quantum computer exhibit entanglement, a quantum mechanical property. Entanglement is a quantum mechanical property where two qubits can be entangled with each other. Any operation on one of the entangled qubits affects the state of the other qubit. Classical computers do not exhibit the property of entanglement.

6. Information processing on a classical computer is performed using logic gates, whereas quantum logic gates are used on a quantum computer. The state of a qubit can be modified using quantum gates that are represented by unitary operators (U).

1.1.2 Quantum Gates and Representations

The basic information-carrying unit on a quantum computer is the qubit. A qubit is a two-level quantum mechanical system represented by two-dimensional complex Hilbert Space, $\mathbb{C}^2$. There can be quantum computers with qutrits, three-level systems, and a term for more general n-level systems is qudits. The state of a qubit at any given time can be represented by a vector in a complex Hilbert space, a vector space with an inner product. The computational basis states of a qubit, $|0\rangle$ and $|1\rangle$ can be represented with vectors as:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

In gate-based quantum computers, the state of a qubit can be modified using quantum gates. Quantum gates are unitary quantum operators (U) in the form of matrices. The operators are applied to vectors to change the state of the qubit to the desired state. If a qubit is in an initial state of $|0\rangle$, a sequence of quantum gates can be applied to change the state as

$$U|0\rangle = |\psi\rangle, \tag{1.1}$$

where $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, and α and β are complex numbers. According to the Born rule [14], the square of the modulus of the amplitudes of a quantum state is the probability of the

quantum state. Hence, $|\alpha|^2 + |\beta|^2 = 1$, such that the probabilities must sum to one, as qubit emerges to one of the states after measurement.

Single-Qubit Gates

A quantum gate that acts on a single qubit is called a unary operator. There are several single-qubit gates in quantum computing, such as the Hadamard gate. The mathematical computation involved to apply a *Hadamard gate* to a qubit is detailed below.

Figure 1.2: Superposition state of a qubit after Hadamard operation.

The state vector of the initial state $|0\rangle$ of a qubit is

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{1.2}$$

As shown in Fig. 1.2, the qubit is passed through *Hadamard gate* where it transforms the initial state to a superposition of $|0\rangle$ and $|1\rangle$ as

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1+0 \\ 1+0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}. \tag{1.3}$$

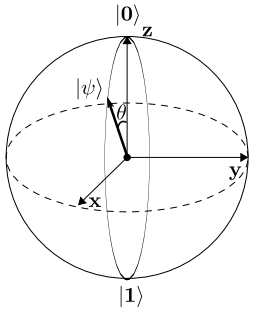


Figure 1.3: Representation of R_y gate operation.

A qubit can also be rotated through the intermediate states between $|0\rangle$ and $|1\rangle$ using single qubit rotational gates such as R_x, R_y, and R_z gates. Fig. 1.3 represents the state of the qubit after rotational gate on the Bloch sphere. For example, when R_y gate is applied

to a qubit in state $|0\rangle$, where

$$R_y(\theta) := \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}, \tag{1.4}$$

then the state of qubit is represented using the following state vector as:

$$\begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) + 0 \\ \sin\left(\frac{\theta}{2}\right) + 0 \end{pmatrix} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) \end{pmatrix}. \tag{1.5}$$

Hence, R_y operation of the qubit in state $|0\rangle$ changes the state of the qubit to state $|\psi\rangle$ where,

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)|1\rangle. \tag{1.6}$$

Two-Qubit Gates

Quantum operators also exist for operation on two qubits at a time. Such operators are called binary operators. When two qubits are operated together, the computational basis states can be represented as $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. The state vectors can be computed as

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{1.7}$$

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{1.8}$$

$$|10\rangle = |1\rangle \otimes |0\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{1.9}$$

$$|11\rangle = |1\rangle \otimes |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \tag{1.10}$$

Correspondingly, computational basis states for n qubits can also be represented using vectors. In a quantum computer, two qubits can be modified together using the gates such as Controlled-NOT gate (*CNOT* gate). *CNOT* operator is represented in the matrix form as

$$\mathit{CNOT} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{1.11}$$

When a *CNOT* gate is applied on two qubits, the first qubit acts a *control qubit*, and the second as *target qubit*. If the *control qubit* is in state $|1\rangle$, we flip the *target qubit*. However, no operation is performed on the *target qubit* if the *control qubit* is in state $|0\rangle$. As operation on one qubit affects the state of the other qubit, *CNOT* gate is used to entangle two qubits in a quantum computer. For example, as shown in Fig. 1.4 when *CNOT* is performed on the state $|11\rangle$, the state is converted to $|10\rangle$ as follows.

$$\mathit{CNOT} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0+0+0+0 \\ 0+0+0+0 \\ 0+0+0+1 \\ 0+0+0+0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle \tag{1.12}$$

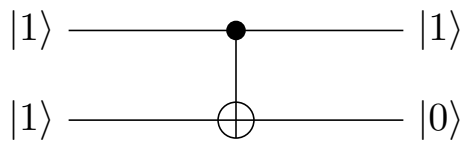


Figure 1.4: Change in state of qubits after *CNOT* operation.

Thus, to perform operations on qubits, quantum gates are used. An algorithm with quantum operations can be designed as a quantum circuit to solve a problem.

Qubits in the quantum circuit are processed using combinations of unary and binary gates to obtain the desired result. As shown in Fig. 1.5, the entire unitary operations are considered as a quantum circuit where each line is considered as wire of a qubit.

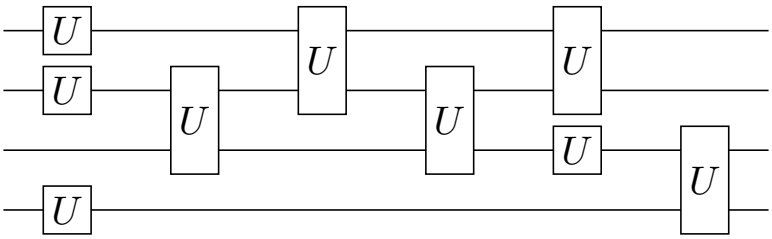


Figure 1.5: Illustration of a quantum circuit.

Measurement of Qubits

After the unitary operations, qubits are measured to know the state of the qubits. Measurement in classical computing does not affect the state of the classical bits. However, in quantum computing, measurement is also an operator that acts on a qubit. Every measurement is represented by a Hermitian operator (O). Hermitian operators are used for measurement as eigenvalues of Hermitian operators are real numbers. In quantum computations, expectation values are calculated using a measurement operator to obtain the probabilistic expected value of the result. For example, consider the *Pauli-Z* operator for measurement represented as

$$\sigma_z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.13}$$

The *Pauli-Z* expectation value $\langle \sigma_z \rangle$ is then calculated as $\langle \psi | \, \sigma_z \, | \psi \rangle$ for a qubit in state $|\psi\rangle$ where,

$$\langle \psi | = |\psi\rangle^T. \tag{1.14}$$

If the qubit is in state $|0\rangle$, *Pauli-Z* expectation value can be calculated as

$$\langle \psi | \, \sigma_z \, | \psi \rangle = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1. \tag{1.15}$$

If the qubit is in state $|1\rangle$, *Pauli-Z* expectation value can be calculated as

$$\langle\psi|\,\sigma_z\,|\psi\rangle = \begin{pmatrix}0 & 1\end{pmatrix}\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\begin{pmatrix}0\\ 1\end{pmatrix} = \begin{pmatrix}0 & 1\end{pmatrix}\begin{pmatrix}0\\ -1\end{pmatrix} = -1. \tag{1.16}$$

Further, if the qubit is in a superposition of $|0\rangle$ and $|1\rangle$, *Pauli-Z* expectation value results a value between $[-1, 1]$. The measurement operation in a quantum circuit is represented as shown in Fig. 1.6 as qubit measurement results is a classical value, indicated using double

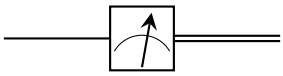


Figure 1.6: Measurement notation in a quantum circuit.

line after the measurement operation. Thus, using measurement operation, we can obtain a classical result as a final output of quantum processing. Further, the classical values can also be post-processed using another classical algorithm or a quantum algorithm for further processing based on the application.

Superposition and Entanglement

In the following example, we show the use of the Hadamard gate, *CNOT* gate, and measurement to depict the quantum mechanical properties superposition and entanglement. As shown in Fig. 1.7, a quantum circuit with three operations is considered to entangle two qubits q_0 and q_1.

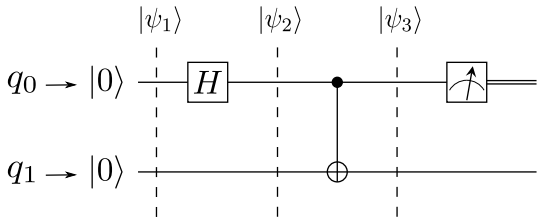


Figure 1.7: Quantum circuit for entanglement.

Initially, the two qubits are in state $|0\rangle$ and the state of the entire quantum system with

the two qubits can be represented as

$$|\psi_1\rangle = |00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{1.17}$$

Then, Hadamard gate is applied on q_0 (Eqn.1.3) obtain a superposition of basis states. The state of the quantum system is represented as

$$|\psi_2\rangle = H|0\rangle \otimes |0\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} \tag{1.18}$$

Finally, *CNOT* gate is applied on the two qubits and the state of the quantum system is represented as

$$|\psi_3\rangle = CNOT|\psi_2\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} + 0 + 0 + 0 \\ 0 + 0 + 0 + 0 \\ 0 + 0 + 0 + 0 \\ 0 + 0 + \frac{1}{\sqrt{2}} + 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} \tag{1.19}$$

$$|\psi_3\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle . \tag{1.20}$$

Thus, from Eqn. 1.20, if q_0 is in the state $|0\rangle$, then q_1 is also in $|0\rangle$. If q_0 is in the state $|1\rangle$, then q_1 is also in $|1\rangle$. Hence, there is a 50/50 chance to obtain the classical value 0 or 1 when the two-qubit quantum system is measured. Thus, the two qubits are in a completely entangled state, also known as the Bell state, or an EPR pair.

1.2 Motivation for the Research Work

Machine learning revolutionized big data analytics with efficient algorithms and techniques for supervised learning [6]. However, big data processing using ML algorithms faces many challenges in dealing with big data due to the rapid growth of data. The areas of concern are computational resources, model selection, optimizing the parameters, and

increasing the algorithm accuracy for tasks such as classification. In this context, Quantum Machine Learning (QML) is emerging as a field of interest in computer science with the intersection of quantum computing and machine learning.

A quantum computer is a computational device that is governed by the laws of quantum mechanics. All the operations on a quantum computer can be described only with the laws of quantum theory [15]. The two main quantum computing paradigms are quantum annealing and gate-modeled quantum computing [16]. Quantum annealing uses the effects called quantum fluctuation, a temporary random change in the amount of energy. Quantum annealing is used to solve quadratic unconstrained binary optimization (QUBO) problems. Thus, computational problems are solved as energy minimization problems using quantum annealing. In gate-modeled quantum computers, information is encoded on qubits, and quantum gates are used to transform the state of the qubits. Due to the new computing paradigms, quantum computing is expected to handle intractable problems that classical computers alone cannot handle. For example, a discrete Fourier transform on 2^n amplitudes can be implemented using a quantum circuit with Hadamard gates and controlled phase-shift gates with exponential speedup [17]. Also, qubits can encode classical data into the superposition of quantum states. Quantum gates are used to process the information encoded into qubits. Quantum computers can provide exponential scaling using qubits for information processing [18].

Quantum computing can enhance classical machine learning (ML) techniques as powerful quantum tools exist for linear algebra [19]. As linear algebra is the basis for machine learning, quantum computing techniques can be incorporated into ML methods for practical performance advantages over classical approaches [20]. The process of manipulating the states of qubits by arbitrarily changing the gate parameters for the desired result is closely related to the training process of machine learning algorithms. The QML algorithm can be designed as a quantum circuit with a sequence of different quantum gate operations to solve ML problems. Quantum machine learning techniques can be used to enhance big data processing for real-world, high-impact applications. However, there exist a limitation on the number of qubits on quantum computers as of today. Also, besides many advantages, the existing quantum computers are restricted by ambient noise. Qubits interact with the environment, thereby leading to the loss of the information encoded in the qubits. Therefore, using present-day noisy intermediate-scale quantum devices (NISQs) [21] to solve ML problems is challenging. Designing QML algorithms that can fit the existing quantum computers and handle the limitations is essential.

Our research work focuses on *Quantum Machine Learning for Big Data Analytics*, the intersection of quantum computing and classical machine learning to enhance supervised learning of classical big data. We investigate the ability of quantum computational techniques in enhancing classical machine learning algorithms for big data analytics. In particular, we propose and study hybrid machine learning methods using quantum and classical computers together to solve supervised learning problems.

1.3 The Focus and Contributions of the Thesis

The major objective of the thesis is *to design methods using quantum computational techniques to enhance classical machine learning for big data analytics*. The efficacy of the proposed methods is verified on two types of data, numerical data, and spatial data for the classification task. While achieving the objective, we contribute the following to the field of big data analytics using quantum machine learning.

1. Proposed an artificial neural network (ANN) using a *quantum computing approach with qubits as artificial neurons* (QC ANN). The model, designed as a parameterized quantum circuit, performs the following tasks:

   (a) Classical data is encoded into qubits, and information on qubits is modified using a series of parameterized quantum gates.

   (b) The parameters of the quantum gates are optimized using classical optimization techniques, and the entire model is trained using supervised learning. Hence, the design methodology uses a hybrid approach.

2. Explored the scope of the hybrid approach in multi-class classification and *proposed a quantum multi-class classifier* (QMCC), consists of the following:

   (a) A new state preparation method to encode the classical data into qubits and testing the compatibility of the method on *IBM Quantum* [8] hardware.

   (b) A variational quantum circuit designed as a classifier with multiple layers of parameterized quantum gates and optimized the gate parameters using classical optimization.

   (c) A new procedure to post-process quantum measurement values suitable for multi-class classification.

3. Analyzed the performance of QC ANN and QMCC on benchmark numerical datasets for the classification task.

4. Proposed *a three-level hybrid quantum-classical architecture* (HybridQC) to process spatial data for the multi-class scene classification of satellite images, consists of the following:

   (a) Quantum measurement-based features extracted from satellite images to obtain quantum representations of images by measuring qubits in a quantum circuit.

   (b) A quantum circuit to extract measurement-based features, suitable for implementation on present-day NISQs or a quantum simulator.

   (c) Deep neural network models built using quantum representations of the images and such models are trained with minimum computational resources.

5. Proposed quantum-classical data processing techniques to process spatial data for satellite image scene classification. The following are the key contributions:

   (a) A data augmentation technique, proposed using a quantum circuit (QDA) to create blended images combining classical and quantum image representations.

   (b) A hybrid quantum-classical convolutional neural network (HQCNN), designed for the image classification task.

6. Analyzed the performance of HybridQC, QDA, and HQCNN on spatial data to prove the efficacy of the techniques for the classification task.

7. Proposed quantum processing techniques for processing of synthetic aperture radar (SAR) images for deep learning. The following are the key contributions:

   (a) Two quantum processing techniques to process SAR images.

   (b) A detailed study of quality metrics and accuracy of the deep learning models on the processed images.

1.4 Organization of the Thesis

The thesis is organized into the following seven chapters:

Chapter 1 starts with a discussion on the importance of big data analytics, followed by a brief discussion on the different types of big data and characteristics. The chapter introduces quantum computing concepts and the basics of quantum operations required for this thesis. Further, the chapter discusses the challenges in dealing with big data using classical machine learning methods. The motivation behind our research on using quantum machine learning for big data analytics is also explained. The chapter then defines the thesis focus and enumerates the major contributions. Finally, the organization of the thesis is outlined, followed by a chapter summary.

Chapter 2 introduces quantum machine learning (QML) and the importance of QML in enhancing classical computational techniques. The chapter also deals with the scope of data processing using quantum computing. QML framework used in our research is explained in this chapter, along with an emphasis on qubit encoding techniques. A brief comparison of the existing standard qubit encoding techniques is also given in this chapter. Further, the chapter introduces the concepts of parameterized quantum circuits, also called the hybrid QML approach, which is the basis for our research.

Chapter 3 discusses the existing work related to quantum machine learning. The related work in quantum machine learning is categorized into two methodologies: (*i*) QML algorithms and (*ii*) hybrid QML approach. Along with the challenges involved in using machine learning algorithms on big data, the chapter also covers the challenges and limitations of using quantum computers.

Chapter 4 focuses on designing hybrid QML models for supervised learning of numerical data. The chapter explains the importance of using the quantum computing approach for artificial neural networks (QC ANN). Further, the design details of the quantum multi-class classifier (QMCC) are given in this chapter. The chapter also presents the performance evaluation of QC ANN and QMCC in classifying numerical data for binary classification and multi-class classification, respectively. Major observations of the work are also presented in this chapter.

Chapter 5 deals with designing and developing the QML framework for spatial data analytics. The chapter explains the hybrid quantum-classical (HybridQC) architecture proposed to enhance the deep learning models. Further, the two quantum-classical image processing approaches: (*i*) quantum circuit for data augmentation (QDA) and (*ii*) hybrid quantum-classical convolutional neural network (HQCNN) are explained

in detail. The chapter analyzes the performance of HybridQC, QDA, and HQCNN on spatial data for image scene classification.

Chapter 6 presents the details of two quantum processing techniques developed to process SAR images for deep learning using a data-centric AI approach. The chapter explains the importance of a data-centric approach where input data is systematically processed/enhanced to build machine learning models. A detailed study of the quality metrics of the processed images is also presented in the chapter. Also, the accuracy results of the deep learning models using the processed images are given in the chapter. The immediate future work for SAR image processing is also discussed briefly in the summary of the chapter.

Chapter 7 concludes the thesis by enumerating the major conclusions and the directions for future research.

1.5 Summary

In this chapter, we discussed the characteristics and types of data generated around the world and the importance of analyzing such data for valuable insights. The need for exploring efficient computational techniques to handle such huge amounts of data is also discussed. A brief introduction to quantum computing is provided, along with the limitations of quantum computers at present. The motivation for our work is given in detail by describing the challenges that machine learning face due to the rapid growth of data. Further, the primary focus of the thesis is defined and the major contributions are enumerated. The organization of the thesis is outlined with the focus of each chapter. In Chapter 2, we discuss the scope of data processing and quantum machine learning with an emphasis on quantum encoding techniques and the hybrid QML approach used in our research.

Chapter 2

Quantum Machine Learning for Big Data Processing

"Quantum computing has rapidly advanced in both theory and practice in recent years, and with it the hope for the potential impact in real applications. One key area of interest is how quantum computers might affect machine learning."

– GOOGLE QUANTUM AI

Large-scale data processing for feature extraction is essential for machine learning algorithms to perform well in solving real-world problems. Although machine learning algorithms perform exceptionally well using classical computers at present, as the size of the data grows, the computational ability of classical computers comes under test. As the data size grows globally at an exponential rate, classical machine learning techniques soon become inundated. Hence, computationally efficient hardware and algorithms are required to analyze huge amounts of data for decision-making. Quantum machine learning (QML) recently gained prominence due to the computational ability of quantum computers in solving machine learning problems that are intractable on a classical computer. This chapter discusses the ability of quantum computing techniques in processing big data with emphasis on standard qubit encoding techniques and the quantum machine learning framework used in the thesis.

Data processing and feature extraction using quantum computing techniques are fundamentally different from classical computational techniques. QML can be used to perform different tasks on a quantum computer to solve specific problems. Qubits in a quantum computer represent data in a superposition of quantum states, and hence, hidden patterns that are difficult for a classical computer to analyze can be found using a quantum computer. Further, entanglement provides a way for the qubits to influence the state of one another during computation. Hence, quantum computers have a natural advantage

to enhance machine learning algorithms. QML algorithms use superposition and entanglement properties to process the information on qubits. Noise resilient data processing techniques can be designed using quantum machine learning as quantum computers deal with inherent noise [22]. In the following section, we discussed the scope of data processing using different computational techniques and highlighted our approach.

2.1 Scope of Data Processing

As shown in Fig. 2.1, there are four different methods to handle data using the existing computational techniques. Each of the data processing techniques, along with the computational techniques, are used to handle different types of data in various applications. Classical data here refers to the different types of data existing in the world that also includes big data. In our thesis, we mainly focus on the *Classical Data and Quantum Processing (CDQP)* category to work on numerical and spatial big data processing. The details are given in the further chapters of the thesis.

Data \ *Processing*	*Classical*	*Quantum*
Classical	Classical Data (CD) Classical Processing (CP)	**Classical Data (CD)** **Quantum Processing (QP)**
Quantum	Quantum Data (QD) Classical Processing (CP)	Quantum Data (QD) Quantum Processing (QP)

Figure 2.1: Data processing and computational techniques.

The details of each method are given in the following.

1. *Classical Data and Classical Processing (CDCP)*: Classical data refers to the data generated from sources such as remote sensing, social media, and transaction data. CDCP is the method where classical data is processed to obtain insights using classical computational techniques. CDCP is widely used in real-time applications such as image classification, object detection, target marketing on social media, and recommender systems.

2. *Classical Data and Quantum Processing (CDQP)*: In CDQP, classical data is processed

using quantum computational techniques. Quantum mechanical properties such as superposition and entanglement are used in quantum computational algorithms to process classical data. Complex machine learning problems for real-time applications can be solved using quantum computational techniques. Quantum computing also consists of techniques that can also be used to enhance the existing machine learning techniques. Our thesis focuses on CDQP to design QML methods and frameworks to solve machine learning problems.

3. *Quantum Data and Classical Processing (QDCP)*: Quantum technology is becoming popular, and many quantum devices are under design. While experimenting with the design of quantum devices, a lot of quantum data is generated. Also, devices similar to quantum sensors generate quantum data. QDCP is a method where quantum data generated from quantum devices (as quantum states) can be processed using classical computational techniques.

4. *Quantum Data and Quantum Processing (QDQP)*: Quantum computational techniques can also be used to process quantum data. Quantum devices in the future can generate quantum data such as state information of the qubits. As the number of qubits increases, QDQP becomes popular in handling quantum data. However, the QDQP methods are expected to be used by quantum physicists to analyze quantum data generated from quantum devices.

2.2 Quantum Machine Learning Framework

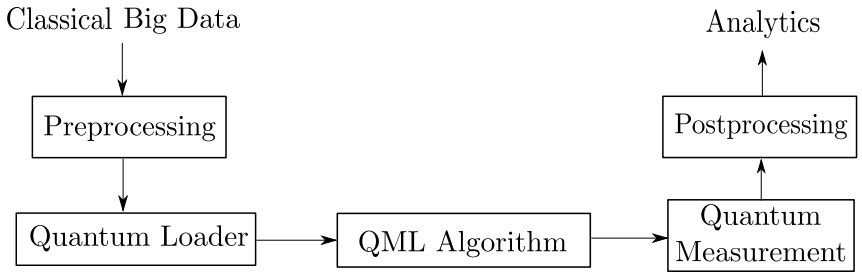


Figure 2.2: Block diagram of QML approach.

The Quantum machine learning approach for big data analytics can be performed using the following steps as shown in Fig. 2.2. Our work considers that the big data is collected and organized using the widely prominent techniques and made available online.

Further, big data processing is performed using the quantum machine learning framework.

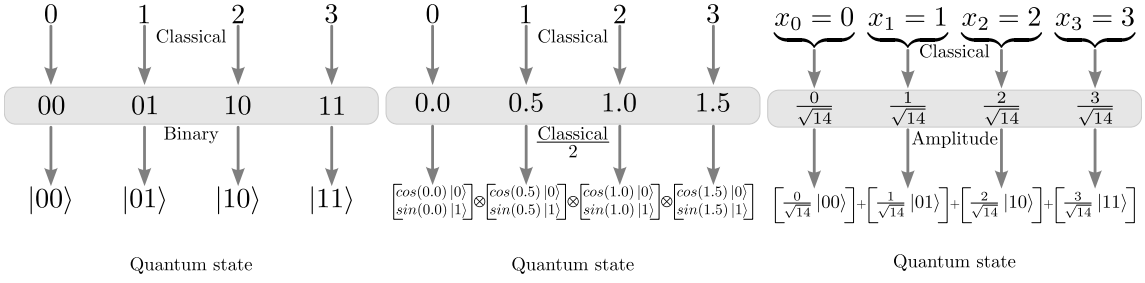
2.2.1 Preprocessing

A huge amount of classical data is generated from different big data sources. The data is cleaned to remove any missing values or discrepancies from the data. Further, data preprocessing techniques are used to transform the data suitable for encoding into qubits. Hence, preprocessing techniques such as data transformation and data reduction are performed on the classical data.

2.2.2 Quantum Loader

Quantum computational techniques require data to be encoded onto qubits for information processing. The qubit encoding process is also called a state preparation where classical data are encoded into qubits to form a quantum state. A quantum loader can be used for the purpose of qubit encoding. Quantum loader consists of a quantum circuit with different unitary operations that encode data into a quantum state using qubits.

The data encoding techniques for a quantum computer are classified into two categories: standard and application-specific. Standard encoding techniques consist of basis embedding, angle embedding, and amplitude embedding. Application-specific embedding techniques are designed based on the input data and target application, using quantum circuits with different unitary operations.



(a) Basis **(b)** Angle **(c)** Amplitude

Figure 2.3: Three embedding techniques.

Basis Embedding

In basis embedding, binary values of classical data are translated into quantum basis states as shown in Fig. 2.3(a). The classical data is converted to binary form, then the string of

binary inputs is translated as a quantum basis state. The binary value of classical data is encoded as a basis state with amplitude as one for that particular basis state. For example, to encode a classical value 2, using basis embedding, the quantum state is represented as

$$|\psi\rangle = 0\,|00\rangle + 0\,|01\rangle + 1\,|10\rangle + 0\,|10\rangle\,. \tag{2.1}$$

In basis embedding, a classical data value with n binary bits is encoded using a basis state of n qubits. Thus, basis embedding requires a huge number of qubits to encode high-dimensional data as binary representations of the classical data are encoded as basis states.

Angle Embedding

In angle embedding as shown in Fig. 2.3(b), classical data features are encoded as the rotational angles of qubits using unitary operations. The qubit rotation can be achieved around x-axis $R_x(v^i)$, y-axis $R_y(v^i)$, or z-axis $R_z(v^i)$ in a Bloch sphere [23] where v^i is the classical data value. The angle embedding encodes n classical features into a minimum of n qubits [15] where each feature is encoded as a rotational angle of a quantum rotational gate.

Amplitude Embedding

In amplitude embedding technique [24, 25], the classical features are mapped into amplitudes of a quantum state. The initial step in amplitude embedding is the conversion of classical data into angular representations, as shown in Fig. 2.3(c). The data is encoded into amplitudes of quantum states using uniformly controlled rotations as per Equation 2.2.

$$|\psi_{amp}\rangle = R\,(x_i, \beta)\,|q_1 q_2 .....q_{s-2} q_{s-1}\rangle\,|q_s\rangle \tag{2.2}$$

R is a function of x_i and β, where x_i is the i^{th} classical feature vector and β is a parameter depending on the dimensions of the classical features [15]. State $|\psi_{amp}\rangle$ is the result of n rotations with respect to the y-axis in a cascade where n is the power for embedding a classical feature vector x_i. In general, to associate each amplitude with a component of the input vector, the dimension of the vector must be equal to a power of two because the vector space of an n qubit register has dimension 2^n. Suppose the dimension of the vector is not an integer power of two. In that case, the input vector needs to be padded with additional zeros to increase the dimension. This padding is depicted in Fig. 2.3(c), where the number of classical features is three, the nearest possible value of 2^n is four. Hence one

additional 0 needs to be padded. After the rotations, the input vector $X = \{x_1, x_2, \ldots, x_n\}$ is encoded in the amplitudes of the quantum state as

$$|\psi\rangle = \sum_{i=0}^{n-1} x_i |i\rangle \tag{2.3}$$

where, x_i is the normalized value of the i^{th} classical feature. Thus, all the values of the input vector X are encoded as amplitudes of the quantum states in superposition. Hence, amplitude embedding is space-efficient as exponential scaling is provided using encoding of values.

Comparison of Embedding Techniques

Basis embedding is a primary way of encoding classical data using basis states. However, basis embedding encodes binary features into basis states, and hence the dimensionality and qubits requirement increases drastically. In angle embedding, a minimum of n qubits to encode n classical features. At present, noisy intermediate-scale quantum devices (NISQs) contain limited qubits to work with. Also, maintaining the coherence of many qubits is a difficult task [26]. Hence, basis and angle embedding schemes are not the best choices for high-resolution satellite images where encoding the classical data values requires a huge number of qubits. In amplitude embedding, 2^n classical data features can be encoded using only n qubits. Thus exponentially fewer qubits are required in comparison to other standard encoding techniques for data encoding. The selection of the data encoding technique depends on the type of data used for analysis and the application.

2.2.3 QML Algorithm

A QML algorithm is used to process and transform the state of the qubits using unitary gate operations. There are two classes of quantum machine learning algorithms. The first class of algorithms is a complete quantum circuit with non-parameterized quantum gates for specific tasks such as classification clustering of input data. The next class of algorithms is quantum circuits with parameterized quantum gates where gate parameters are optimized for a specific application. The optimization of the gate parameters of the quantum circuit is performed using classical optimization techniques as shown in Fig. 2.4. Thus, the second class of QML algorithms is considered a hybrid approach with both classical and quantum computations. Our thesis focuses on the hybrid approach where both quantum and classical computational techniques are used to solve machine learning algorithms. We

further propose novel approaches to enhance the existing machine learning algorithms.

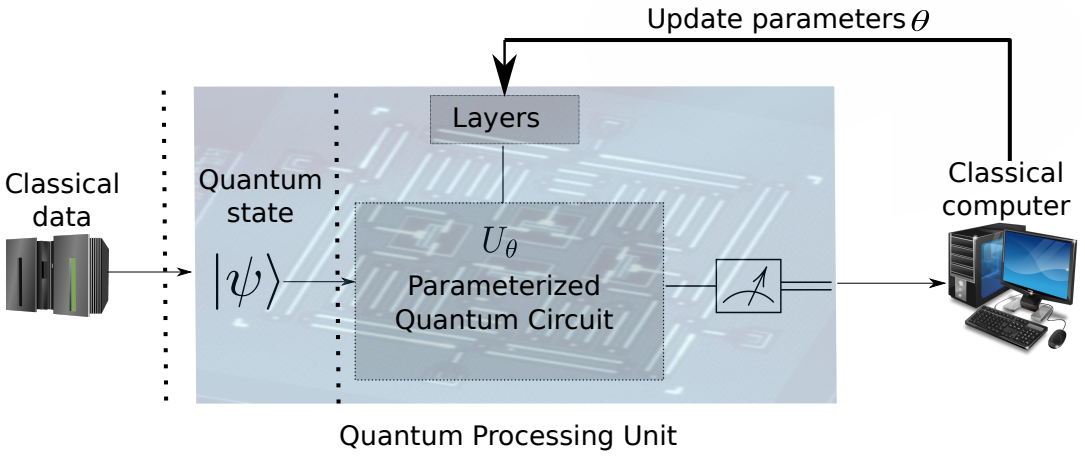


Figure 2.4: A hybrid classical-quantum circuit model.

Parameterized Quantum Circuits

Hybrid classical-quantum computation models [27] are built to provide the quantum advantage to classical computers. Fig. 2.4 depicts the outline of a hybrid computational model. The computation models are designed as a typical quantum circuit, called a variational circuit as they are suitable to implement on an existing quantum processing unit. A variational circuit is a quantum circuit that has a fixed initial state, parameterized quantum circuit, and measurement [28, 29, 30]. The significant difference from a traditional quantum circuit is that the gates used to build the variational circuit are parameterized as $U(x;\theta)$ where x (input) and θ are parameters. Further, parameters of the gates (θ) used in the variational circuit are optimized by a classical computer.

The model starts with a classical data item prepared as a quantum state. The quantum state $|\psi\rangle$ is then passed through a sequence of quantum operations which is designed as a variational circuit with parameterized gates (U_θ) and has a layered architecture with the repetition of the layers. The initial quantum state $|\psi\rangle$ with encoded classical data is modified through the layers to state $|\psi'\rangle$. The layers are repeated to engage the parameterized gates in a better learning process. Finally, the desired qubits are measured for the output value. Gate parameters in all the layers are updated based on the difference between the predicted output and the ground truth. A classical computer is used for the optimization of the gate parameters. In our work, we used parameterized quantum circuits to design

the hybrid models QC ANN and QMCC for numerical data classification. A detailed description of the designed variational circuits for hybrid models is given in Chapter 4.

2.2.4 Quantum Measurement

Quantum measurement is performed on one qubit or a set of the qubit to obtain the result. For example, if we design a QML algorithm for the binary classification task, the expectation value of projection-valued *Pauli-Z* measurement on a qubit in the state $|0\rangle$ gives the result as 1, and on $|1\rangle$ it results in -1. Hence input can be classified based on the measurement of the qubits. Further, in a hybrid approach, the measurement values are used to optimize the parameters of quantum gates.

2.2.5 Postprocessing

The output values of a quantum measurement are numerical values that can be represented in classical domain. The obtained values based on the measurement operator can be post-processed using either classical or quantum processing based on the application. In our work, we used classical postprocessing to process the quantum measurement values obtained during the process of computation in HybridQC, QDA, and HQCNN methods. Further details of the methods are given in Chapter 5.

2.3 Summary

In this chapter, we briefly described different data processing approaches using different types of computations. We also described in detail the different steps involved in the quantum machine learning framework for big data analytics with an emphasis on the quantum loader. Qubit encoding schemes are discussed with a comparison between the schemes. In Chapter 3, we discuss the existing work related to methodologies of quantum machine learning, emphasizing the hybrid QML approach.

Chapter 3

Related Work

"If I have seen further than others, it is by standing upon the shoulders of giants."
– ISAAC NEWTON

"The reading of all good books is like a conversation with the finest minds of past centuries."
– RENÉ DESCARTES

Quantum computers are fundamentally different from classical computers as quantum mechanical properties are used for information processing. Hence, it is believed that quantum systems produce information in form of quantum states that cannot be produced by classical systems [31]. The critical aspect is to understand and discover the power of quantum machine learning and what quantum computing can do for machine learning applications [32]. This chapter presents the details of existing quantum machine learning methodologies to solve machine learning problems using quantum computers.

3.1 Quantum Machine Learning Methodologies

There are two different methodologies that are used for quantum machine learning. The first one is to design quantum algorithms as a complete quantum circuit to solve machine learning problems. The designed complete quantum circuits require more qubits, and therefore, the algorithms can probably solve machine learning problems such as classification, clustering, and dimensionality reduction faster than classical algorithms [33, 34]. The second prominent methodology used for quantum machine learning is a hybrid approach using NISQ heuristics. As mentioned in the previous chapter, hybrid classical-quantum computational models are built to provide a quantum advantage to classical computers. In the next section, we discuss the related work on the hybrid approach in detail. The hybrid approach at present can enhance machine learning algorithms by giving support to

classical computation. Fig. 3.1 show the state of quantum machine learning research based on the methodologies used for solving the problems.

The idea of using a complete quantum circuit to solve machine learning problems gained momentum during the early 2000s. Powerful tools for linear algebra are designed using quantum computing such as Fourier transforms, finding eigenvectors and eigenvalues and solving linear equations. In 2009, the quantum algorithm for linear systems of equations, also called the HHL algorithm, is designed by Aram Harrow et al. for solving linear systems [19]. As linear algebra is the core computational component for machine learning, the quantum computing community believes that machine learning problems can be solved efficiently using quantum computation techniques.

In comparison, there exists an exponential quantum speedup over their best known classical counterparts [19, 35, 36]. The quantum algorithms can be used for a variety of data analysis and machine learning algorithms, including linear algebra, least-squares fitting, gradient descent, Newton's method, principal component analysis, linear, semi-definite, and quadratic programming, topological analysis, and support vector machines [33]. However, practical implementation of such techniques requires large-scale quantum computers.

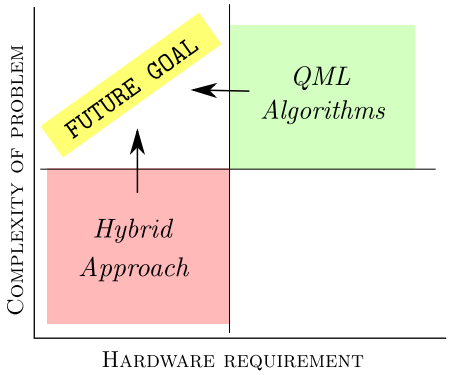


Figure 3.1: Present state and future goal of research on hybrid and QML algorithms.

Hybrid quantum-classical models can be used to overcome the limitations of quantum computers at present and utilize existing quantum computers to perform different tasks. Hybrid models consist of parameterized quantum circuits that use different quantum gates to be designed as machine learning models. The models can be designed with a few qubits available at present and can also be scaled based on the availability of qubits. The developed models can be used for a variety of data-driven tasks, such as supervised learning.

The following Section 3.2 provides details of the existing work related to the work presented in Chapters 4 and 5. Emphasis is given on the challenge of training machine learning models on huge amounts of data. Section 3.2 also discusses the techniques and methods used to overcome the limitations of a quantum computer.

3.2 Hybrid QML Approach

Machine learning algorithms revolutionized the way of analyzing huge amounts of data for useful insights [37, 38, 39] and applications. Artificial neural networks [40] removed a lot of obstacles in big data processing, thereby, turning the data into information for successful decision making. Data growth and size create new problems to be addressed and challenge the existing computing power of artificial neural networks. An ANN can learn complex relationships from the input data using the parameters between the network layers. These parameters are optimized using back-propagation during the training phase. As the size and dimension of the input data grow, training a neural network with fewer parameters is a difficult task.

In recent years, deep learning (DL) [41] based methods such as convolution neural networks (CNNs) proved to be efficient for various image processing and scene classification tasks. Such methods [42, 43, 44] extract features from an image using trainable multilayer networks and are proven to be successful for remote sensing image scene classification [45, 46]. CNN-based deep learning models process images in the form of multiple arrays using filters of variable sizes. Features from the images are extracted through convolution layers and pooling layers. Finally, the fully connected layers are used for decision-making. Fine tuning [47, 48] of CNN models proved to be efficient for the scene classification task. However, the implementation of such methods requires a high-performance computational facility as many parameters are used in training such models.

Extracting interpretable information and knowledge from the spatial big data generated by remote sensing imaging systems is also a challenging task. Reichstein et al. [49] explained the need and importance of integrating physical modeling with machine learning to address the challenge. Lei Ma et al. [50] discussed the prominence and efficiency of DL models for image scene classification, object detection, semantic segmentation, and land-use classification. The authors also highlighted the challenges in training the supervised DL models such as CNN.

The training of deep neural networks (DNNs) with more layers is a complex task in terms of optimization and tuning of parameters [51]. As the number of parameters in-

creases, the convergence of the optimization process is difficult during training, of the model [52]. The deeper neural network architectures with a large number of parameters often lead to multiple local minima [53]. Computationally expensive hardware and memory-intensive methods are required to manage a large number of parameters in a deep model with more layers [54]. Also, remote sensing image data is high-dimensional, and the availability of training samples is limited. Hence, the training process of DNNs is a complex task. Thus, stacking of the layers with more parameters is not an effective way to improve the efficiency of the models [55, 56].

Complex computational problems can be solved efficiently using a quantum computer [18]. Classical computing can store and process the data in binary digits. In contrast, quantum computers store and process the data using qubits [57] which use both 0 and 1 at the same time. The composite quantum state produced by the quantum-mechanical property is called superposition. Entanglement is another quantum-mechanical property that is used in many algorithms to create interactions between qubits. In this quantum era, there exists a necessity to explore the possibility of designing QML algorithms suitable for implementation on QPUs to harness the power of quantum computing. Quantum enhanced machine learning algorithms [58] can solve complex problems that are difficult for a classical computer. Biamonte et al. [31] described the advantages of using quantum computing in solving machine learning problems. In their work [59, 15], Schuld et al. designed a quantum circuit as a classifier for binary classification using a distance-based kernel function. Emphasizing the importance of QML, authors in [60, 61] proposed supervised machine learning models using a hybrid classical-quantum approach that uses a classical computer to optimize quantum parameters.

Tacchino et al. [62] implemented an artificial neuron on a quantum processor and discussed the possibility of implementing artificial neural networks on a real quantum computer. Open-source software [16] are developed to implement quantum algorithms on a real quantum processor or a quantum simulator. The work in [33, 11, 63] states that supervised, unsupervised, and reinforcement learning techniques can be performed using quantum computing. Schuld et al. [59, 15] implemented a distance-based classifier with a quantum interference circuit for binary classification. The authors devised a simple quantum circuit and used the quantum computational properties to create a quantum-circuit-based binary classifier. The main focus of the work is to leverage the computational power of a quantum computer instead of trying to execute a machine learning algorithm on a quantum computer. The work proved that quantum computing abilities are unconven-

tional and can be used to enhance machine learning algorithms. However, identifying the correct problem statement and constructing a quantum circuit as a machine learnig model are challenging. The basic idea is to use quantum interference to evaluate the distance measure of a kernel classifier in quantum parallel. Also, information encoding and avoiding complex quantum circuit design are critical in the scenarios. The authors demonstrated the circuit for the classification task on the Iris flower dataset on the 5-Qubit IBM quantum computer. Subsequently, different types of binary classifiers on quantum computers were designed [60, 61, 62].

The primary step to performing classification on a quantum computer is that the input data must be encoded in a quantum state. The data can be encoded as the amplitudes of individual qubits in an entirely separable state (qubit encoding) or the amplitudes of an entangled state (amplitude encoding). The work performed in [60] tests the quantum classifier using both methods. The circuits used are tree-like parameterized quantum circuits one known as a tree tensor network (TTN) and another known as the multi-scale entanglement renormalization ansatz (MERA). The circuits use single-qubit, two-qubit, and three-qubit rotations along with fixed CNOT gates. The authors tested the models on an IBM quantum computer.

Quantum computing can also be used to implement kernel methods [64, 65] for classification tasks. In [61], the authors proposed two classifiers one is a variational quantum classifier, and another is a quantum kernel estimator. The work shows that quantum computers can build a feature map that is difficult to estimate classically, thereby producing a quantum advantage. The observation made a crucial step towards building machine learning algorithms that can be implemented on noisy intermediate-scale (NISQ) devices. The kernel methods proposed are expected to work beyond binary classification.

Further advancement in the quantum research related to machine learning is proposed in [62]. The authors proposed the quantum equivalent of the classical perceptron. First, the equivalent of k-dimensional classical input and weight vectors is encoded on the quantum hardware using n qubits, where $k = 2^n$. The quantum perceptron model can sort out simple patterns, such as vertical or horizontal lines, among all possible inputs.Quantum versions of deep learning techniques such as convolutional neural networks [66], Boltzmann machines [67], and generative adversarial networks [68] were also developed. Superposition, entanglement, and interference are the quantum-mechanical properties that are used in many of such quantum machine learning algorithms [69]. QML algorithms can also be applied in the field of Chemistry [70].

Quantum image processing is an upcoming research area at the intersection of quantum computation and image processing. Quantum image representations, processing algorithms, and image measurement are the crucial areas in the research [71]. Pengao Xu et al. [72] developed a quantum circuit-based quantum image processing algorithm. The algorithm uses the Kirsch operator and an edge extraction method to perform real-time image processing with good accuracy. The developed algorithm is also better in image processing speed than the classical edge extraction algorithms. Mhafuzul et al. [73] presented a hybrid quantum-classical neural network for attack detection using image data.

Data augmentation is a very important technique used during the training of machine learning models. Augmentation refers to method that virtually increases the size of the different datasets. Chao Li et al. [74] presented a data augmentation technique on data such as data collected from inertial sensor data to improve deep learning performance. Data augmentation can also be performed on combinations of data from sensors such as inertial and image sensors [75]. Deep learning models for limited-size image datasets benefit from data augmentation techniques [56]. However, there is a necessity for new data augmentation techniques as different varieties of data is generated due to the advancement of sensor technology.

Motivated by the recent advancements in realizing quantum information processors, quantum versions of neural network algorithms were developed. Cong et al. [76] developed quantum circuit-based algorithm inspired by convolutional neural networks. Quantum neural networks are also developed as a variational quantum circuit built in the continuous-variable (CV) architecture [77]. Henderson et al. [78] created a quantum version of convolutional neural networks using a transformational layer called a quantum convolution, or quanvolutional layer. Random quantum circuits are used for operating on the input by locally transforming the data. Mari et al. [79] extend the concept of transfer learning to hybrid neural networks composed of classical and quantum elements. The hybrid model uses a pre-trained classical network augmented to a variational quantum circuit for classification.

Even though the existing quantum neural network models show an advantage over the classical methods, they are not completely scalable. As the number of quantum operations increases, the depth of the circuit increases and qubits lose coherence. Such quantum neural networks require training optimization [80] and state stabilization [81]. Hence, we propose a hybrid quantum-classical architecture to combine quantum computation techniques to handle data together with a classical deep neural network for image classification.

However, solving machine learning problems using the present state of a quantum computer has its own limitations. Quantum computers are complex systems due to their need for perfect isolation from the environment. Lack of isolation results in the qubits of quantum computers losing coherence. The property of decoherence of qubits is a challenging problem when designing a quantum algorithm as a quantum circuit on near-term quantum computers [21]. As mentioned in Section 1.2, QML algorithms at present face challenges such as availability of qubits and decoherence. Hybrid classical-quantum models are used for designing QML algorithms to overcome the existing limitations of a quantum computer. Table 3.1 summarizes the methods in the related work relevant to the thesis.

Table 3.1: Different types of data and characteristics

Method	Highlight of the work
Distance-based classifier	Quantum inference circuit for binary classification of input data [59]
Hierarchical quantum classifier	Multiple implementations of quantum tree tensor networks for classification [60]
Quantum kernel methods	Supervised learning using quantum kernel for enhanced feature spaces [61]
Quantum perceptron	Quantum version of classical perceptron implemented on quantum hardware [62]
Quanvolutional neural network	Quantum version of convolutional neural networks for extracting image features [78]
Circuit-centric classifier	A low-depth variational quantum algorithm for supervised learning [82]
Quantum transfer learning	Feature learning using quantum computation for image recognition and quantum state classification [79]

3.3 Summary

This chapter discussed the existing research on quantum machine learning using QML algorithms and a hybrid approach. The important observations from the literature are: *(i)* analyzing huge amounts of data using machine learning algorithms requires a huge number of trainable parameters, *(ii)* quantum computing enhances machine learning, and *(iii)* there exist limitations on qubits at present in the quantum processing units. Therefore, we consider designing hybrid quantum-classical models to overcome the limitations and enhance machine learning using quantum computational techniques. Chapter 4 describes the models developed for supervised learning on numerical data for binary classification. In Chapter 5 we proposed different hybrid techniques to handle big spatial data for scene classification.

Chapter 4

Hybrid QML Models for Supervised Learning

"The real voyage of discovery consists not in seeing new sights,
but in looking with new eyes"
– MARCEL PORUST

Supervised learning is a machine learning technique where the machine learning model is trained using labeled data to learn the features. We designed hybrid models using quantum and classical computations to perform supervised learning and solve classification problems in our work. To design the models, we carried out our research in two phases. The first phase presented in this chapter aims to understand the application of quantum computing on numerical data. We applied quantum computing in the area of artificial neural networks. Further, we design a quantum variational circuit for classification.

4.1 Quantum Computing for Artificial Neural Network

Artificial neural networks (ANN) are proven to be efficient in solving many problems for big data analytics using machine learning. The complex and non-linear features of the input data can be learned and generalized by ANN. In the big data era, enormous amounts of data arrive from multiple sources. A stage is expected to be reached where even supercomputers are likely to be inundated with big data. Training an ANN in such a situation is a challenging task due to the size and dimension of the big data. Also, a large number of parameters are to be used and optimized in the network to learn the patterns and analyze such data. Quantum computing is emerging as a field that provides a solution to this problem as a quantum computer can represent data differently using qubits. Qubits on quantum computers can be used to detect the hidden patterns in data that are difficult for a classical computer to find. Hence, there exists significant scope for application in the area of artificial neural networks. We primarily focus on training an artificial neural

network using qubits as artificial neurons in this work. The simulation results in the Section 4.4 show that our quantum computing approach for ANN (QC ANN) is efficient when compared to classical ANN on the benchmark dataset. The model with qubits as artificial neurons can learn the features of data using fewer parameters for a binary classification task on numerical data. We demonstrate our experiment using a quantum simulator and optimize the quantum parameters used in QC ANN using classical computation.

4.1.1 Methodology of QC ANN

In this section, we first briefly describe the classical ANN before discussing QC ANN. In a classical ANN (Fig. 4.1), the *input layer* takes N-dimensional data onto the nodes called artificial neurons. These nodes are connected to the nodes in the *hidden layer* and the connections contain certain weights which are parameters that can be optimized. The *hidden layer* nodes are connected to the *output layer* with weighted connections. Finally, an activation function is used to get an output value from the network. The loss is calculated between the actual and the target output in the next step. The parameters are optimized through backpropagation until the loss minimizes between the actual and target output. The training process always depends on the input dimension and the size of the dataset.

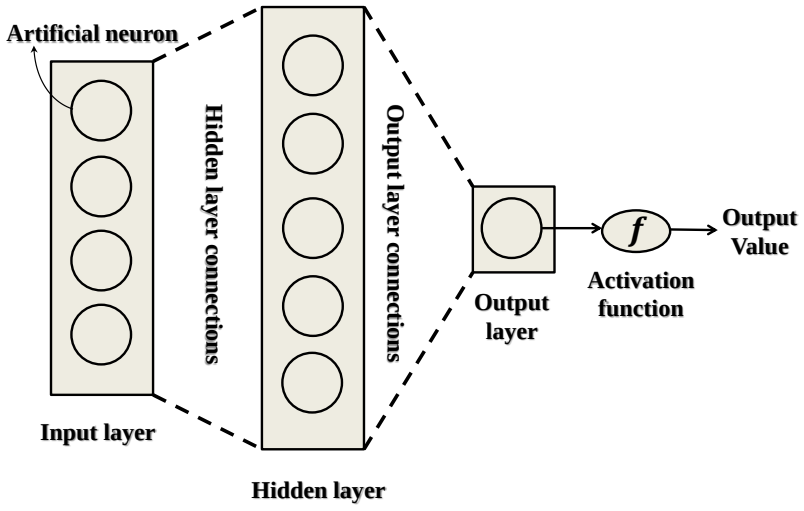


Figure 4.1: Classical Artificial Neural Network (ANN) with artificial neurons as the nodes of the network.

Training an ANN is a significant task where the parameters of the network are optimized by tuning the hyperparameters (parameters such as depth of the network and width of the network) to avoid overfitting or underfitting of the model. The top-level parameters of a machine learning model, that control the learning process are called hyperparameters. The training task becomes complex with the increase in dimension and size of the data because as the size of parameters to be optimized increases computation power requirement increases.

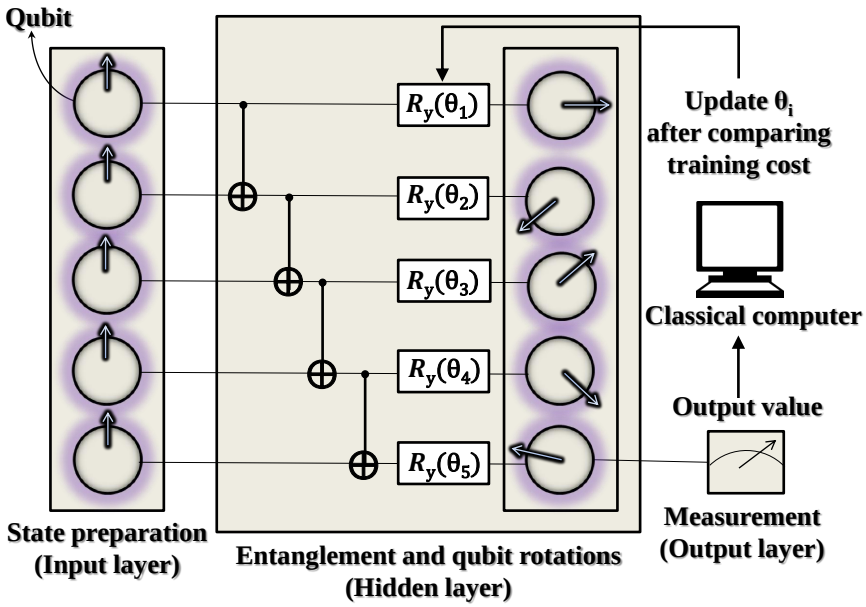


Figure 4.2: Quantum circuit with qubits as nodes of ANN.

Quantum computing has the potential to handle such complex tasks that are intractable on a classical computer. Fig. 4.2 represents the quantum computing approach used to build an ANN using qubits as artificial neurons. The approach is named quantum computing approach for ANN (QC ANN). The artificial neurons in the *input layer* of classical ANN are replaced with qubits. The N-dimensional input data is encoded as a quantum state with the superposition of 2^k states where k is the number of qubits. We use the amplitude embedding scheme [24] for the state preparation process. The *hidden layer* consists of connections between qubits for interaction as an *entanglement*. The states of the qubits are modified using gates with certain rotational parameters. Finally, we measure the state

of a qubit for an output value. We compare the actual output value with the target value and optimize the rotational parameters of gates until we get the desired output after measurement. All the quantum operations in Fig. 4.2 can be executed on a near-term quantum computer and the optimization of rotational parameters of single-qubit gates can be done using a classical computer. QC ANN is a hybrid approach that uses both classical and quantum computers together for the binary classification task.

The learning processes of QC ANN is different from classical ANN. The QC ANN is implemented as a circuit on a quantum computer. The state preparation process allows encoding of the input data onto qubits as a quantum state. That is, the input data is embedded as amplitudes of different states in superposition (Section 2.2.2). The qubits are now entangled in a way that modification of the state of one qubit influences the state of others. Then, the quantum state is modified with gates for single-qubit rotations through the hidden layers. In the *hidden layer* each qubit is rotated by a quantum gate and hence, the state of the system changes. Finally, the state of the desired qubit is measured to obtain a value and compared with the target value to calculate the loss. The rotational parameters are modified until the quantum system reaches a state at which the desired qubit, when measured, gives the target output value. The learning process here denotes that the quantum circuit learns the complex relationship between input and the output as rotational parameters of gates which transforms the state of the quantum system. Thus, the process of learning is entirely different from that in a classical ANN.

The advantages of QC ANN are:

1. Using k qubits, 2^k attributes of the input data are encoded as a superposition of states onto a quantum computer. Consider a normalized input data instance of 32 values given as $x_1, x_2, x_3, ..., x_{32}$. Now the input data can be encoded as a quantum state (using 5 qubits) with superpositon of different states as $x_1 \left|00000\right\rangle + x_2 \left|00001\right\rangle + x_3 \left|00010\right\rangle +, ..., + x_{32} \left|11111\right\rangle$ using amplitude embedding.

2. Using qubits as artificial neurons avoids the usage of nodes in the *hidden layers* as in classical ANN. Thus, the size of parameters to be optimized reduces, as there are no connections to be established between the layers.

3. Quantum measurement itself acts as an activation function that saves the computational effort.

However, QCANN can only perform binary classification. Encoding data into quantum computers is an important task and there is a necessity for different data encoding

techniques for encoding data into qubits. Hence, we addressed the considerations using a quantum multi-class classifier (QMCC). In the following section, we describe QMCC that performs multi-class classification using a different quantum circuit we designed.

4.2 Quantum Multi-Class Classifier

We propose quantum multi-class classifier (QMCC) as a variational circuit with a hybrid classical-quantum approach using quantum mechanical properties such as superposition and entanglement. We primarily focus on solving the machine learning problem of multi-class classification using a hybrid model based on both quantum and classical computers together for the classification task. A unitary operation on a single qubit for the state preparation is designed and also demonstrated using a real quantum computer on the IBMQX platform. The entire variational circuit for the classification task is implemented on a quantum simulator. We performed our quantum simulations on three benchmark datasets: *Iris dataset*, *Banknote Authentication (BNA) dataset*, and *Wireless Indoor Localization (WIL) dataset* for machine learning algorithms. Our simulation results show that the QMCC model classified *Iris dataset* with an accuracy of 92.10%, *BNA dataset* with an accuracy of 89.50%, and *WIL dataset* with an accuracy of 91.73%. The proposed model can also be extended to multiple class classifiers. The accuracy of a model is calculated from the confusion matrix [83] with true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The formula for accuracy is given as following:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}.$$

We designed a variational quantum circuit as a machine learning model for classification using the hybrid classical-quantum approach. The hybrid approach allows the machine learning model to overcome the existing limitations of a quantum computer and can perform better. Our model is designed for the classification of three classes and can be further extended to classify multiple classes. The major contributions of this work are as follows:

1. Multi-class classification using QMCC as a variational circuit that classifies three classes using the hybrid classical-quantum approach.

2. Provisioning of separate data qubits to encode classical data and ancilla qubits for measurement in the variational circuit design.

3. State preparation using a unitary operation to encode the classical data instance as a quantum state on a qubit and verification of the proposed state preparation on IBMQX [84] platform.

4. Proposal of a new approach for class label prediction after *Pauli-Z* measurement using *softmax* function for achieving better results.

5. Detailed analysis of the results using QMCC on a quantum simulator in comparison with the amplitude embedding scheme for state preparation.

4.2.1 QMCC as a Variational Circuit

The variational circuit is shown in Fig. 4.3 [85] is designed to classify dataset with four features.

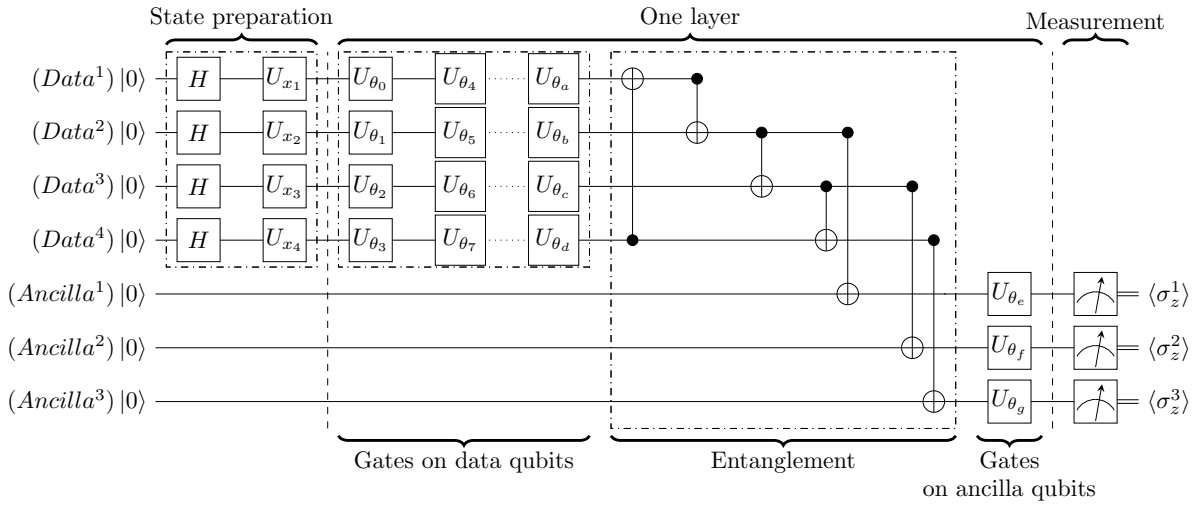


Figure 4.3: Illustration of the designed variational circuit model with one layer for classification of the three classes. The original implementation of the model consists a total of seven similar layers.

The operations performed on the variational circuit are as follows. Initially, all the four qubits in the variational circuit are prepared as $|0\rangle$. To classify any data on a variational circuit, each instance of the data is to be encoded into the four qubits. Firstly, for the purpose of encoding, all the four qubits are operated with *Hadamard* gate individually to place the qubits in an equal superposition of $|0\rangle$ and $|1\rangle$. Second, a classical data instance with four values is encoded into four individual qubits using unitary operation on each qubit with a square unitary matrix designed for the state preparation as given in the next

section. Hence, using the state preparation process, classical data is encoded on the qubits as a composite state.

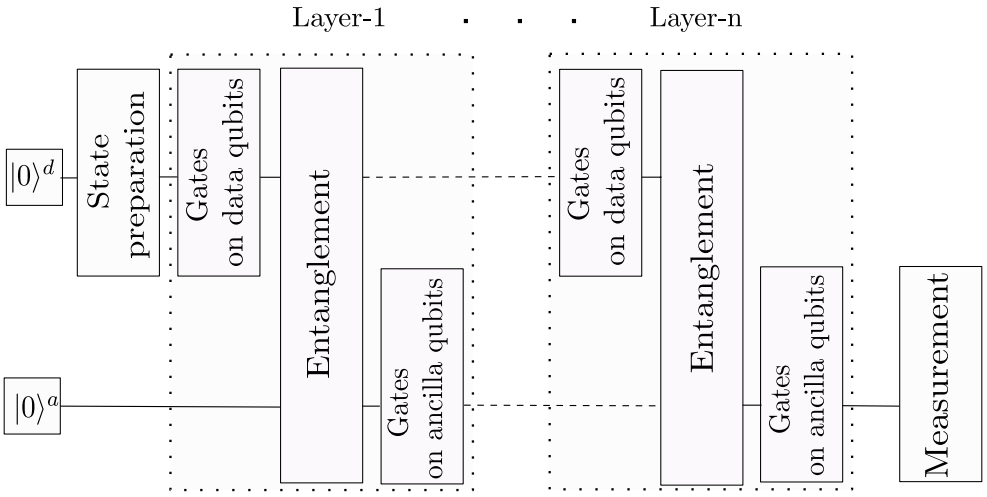


Figure 4.4: Generalized architecture of the designed variational circuit model with n-layers.

Algorithm 4.1 shows the working of the designed quantum multi-class classifier.

Algorithm 4.1: QUANTUM MULTI-CLASS CLASSIFICATION

Input: N-dimensional input features $\mathbf{x}^d \in \mathbb{R}^N$ re-scaled element-wise in order to lie in $[0, \frac{\pi}{2}]$

Output: Class label for the corresponding data vector y^d for the data set $\mathcal{D} = (\mathbf{x}^d, y^d)_{d=1}^D$ where $d = 1 \ldots D$ represents the row number in a dataset.

Initialize: $Data\_qubits \leftarrow |0\rangle$, $Ancilla\_qubits \leftarrow |0\rangle$
$k \leftarrow$ number of layers
$\boldsymbol{\theta} \leftarrow$ quantum gate parameters optimized using classical computer
Data_qubits $\leftarrow$ STATE PREPARATION $(x_i{}^d)$ $\forall i \in 1, 2, \ldots, N$
for $i \leftarrow 1$ ***to*** k **do**
    $R_Y(\boldsymbol{\theta})$ rotations on $Data\_qubits$
    ENTANGLEMENT ($Data\_qubits$, $Ancilla\_qubits$)
    $R_Z(\boldsymbol{\theta})R_Y(\boldsymbol{\theta})R_Z(\boldsymbol{\theta})$ rotations on $Ancilla\_qubits$
end
Values $\leftarrow$ MEASUREMENT ($Pauli\text{-}Z(Ancilla\_qubits)$)
Class label $\leftarrow$ SOFTMAX(Values)

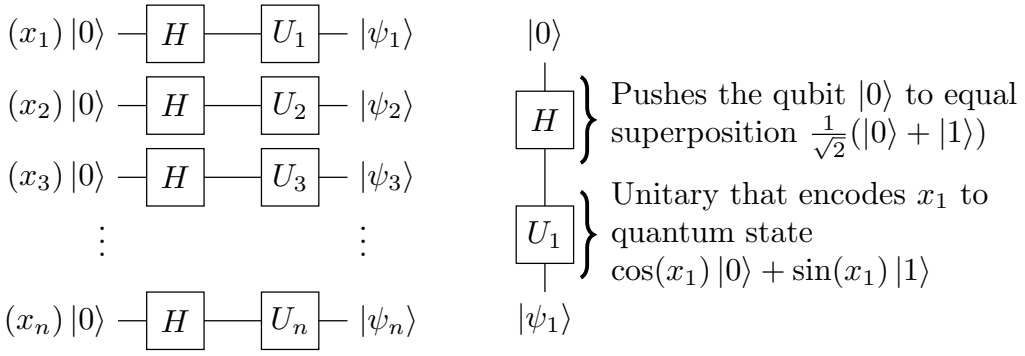
After the state preparation, the variational circuit is designed with multiple layers of entangled rotational gates on data qubits and ancilla qubits with adjustable parameters.

The experimental model of QMCC consists of seven trainable layers. Finally, the ancilla qubits are measured and output values of the measurement are processed to obtain class labels as per Eqn. 4.5. The parameters of the layers in the variational circuit are adjusted in multiple iterations using an optimization technique such that the final circuit with adjusted parameters of the gates predicts the class label of the given input with the desired accuracy.

State Preparation

Fig. 4.4 represents the generalized architecture of our variational circuit model. Quantum computers can handle data only as quantum states. Hence, a classical data item is encoded as a quantum state on the qubits for processing. The *State preparation* part of the circuit encodes the classical data onto the four data qubits. The encoding scheme used in this work consists of single qubit rotations [86].

Any N-dimensional classical dataset, $\mathcal{D} = (\mathbf{x}^d, y^d)_{d=1}^{D}$, where $\mathbf{x}^d \in \mathbb{R}^N$ is an N-dimensional input data, can be encoded into a quantum state with y^d as the class label for the corresponding data vector. The classical data vectors are re-scaled element-wise in order to lie in $[0, \frac{\pi}{2}]$ and each vector element is encoded on a qubit as $\psi_n^d = \cos(x_n^d)\,|0\rangle + \sin(x_n^d)\,|1\rangle$.



(a) Illustration of state preparation scheme.

(b) x_1 encoded as a quantum state $|\psi_1\rangle$.

Figure 4.5: State preparation scheme used to encode classical data instances into quantum states.

On quantum processing units, the qubits are considered in the state $|0\rangle$ initially. Hence, as in Fig. 4.5(a), we pass the qubits through the Hadamard gate to place the qubit in an equal superposition of $|0\rangle$ and $|1\rangle$. Then, the superposition state is passed through a square unitary matrix U that transforms the equal superposition to $\cos(x_1)\,|0\rangle + \sin(x_1)\,|1\rangle$ for a data element x_1 in a dataset as can be seen in Fig. 4.5(b). All the data elements in Fig. 4.5(a) are encoded in a similar way. Finally, the encoded data vectors are given to the next part of the circuit as a tensor product $\psi^d = \otimes_{n=1}^{N} \psi_n^d$. The mathematical computation involved to

encode a single instance is detailed below. The state vector of the initial state $|0\rangle$ of a qubit is

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}. \tag{4.1}$$

The qubit is passed through *Hadamard gate* where it transforms the initial state to a superposition of $|0\rangle \, and \, |1\rangle$ as

$$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{4.2}$$

Now, the superposition state is applied with the unitary gate operation U_{x_n} with the matrix designed to encode the classical data onto a qubit as

$$\begin{bmatrix} \cos\left(\frac{\pi}{4} - x_n\right) & -\sin\left(\frac{\pi}{4} - x_n\right) \\ \cos\left(\frac{\pi}{4} - x_n\right) & \sin\left(\frac{\pi}{4} - x_n\right) \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} \cos(x_n) \\ \sin(x_n) \end{bmatrix}. \tag{4.3}$$

Thus, the final state of a qubit is given with the following equation as

$$\psi_n = \cos(x_n) \, |0\rangle + \sin(x_n) \, |1\rangle. \tag{4.4}$$

Classification algorithm as parameterized quantum circuit

The classification algorithm is designed with layered architecture in the variational circuit. In each layer, there exist three parts: *Gates on data qubits*, *Entanglement*, and *Gates on ancilla qubits*.

The first part of the layer, *Gates on data qubits*, ensures single-qubit rotations where the qubits are rotated in a specified way for a data vector of a particular class label. The second part of the layer is *entanglement* of the circuit, designed to entangle data qubits with each other and also with the ancilla qubits. The entanglement between the qubits is designed in such a way that the rotation of each data qubit influences the rotation of all the other data qubits, including the ancilla qubits in a specific way. Finally, *Gates on ancilla qubits* rotate the ancilla qubits to enable a better learning process with a few more parameters.

All the gates used in the circuit are parameterized unitaries. Thus, the qubits are manipulated in a specified way through the unitaries. At the end, ancilla qubits are measured for the *Pauli-Z* expectation value. The output class encoding scheme, used for class labels, is based on *one-hot encoding* scheme. For a dataset with three classes, *Class*-1 is labeled as $[1, 0, 0]$, *Class*-2 as $[0, 1, 0]$, and *Class*-3 as $[0, 0, 1]$. Now, the *Pauli-Z* expectation values from

the measurement of ancilla qubits are passed through *softmax* function [87]. The input values to the *softmax* function (S) are transformed to a set of probabilities as

$$\mathrm{S}(\langle\psi_i|\,\sigma_z\,|\psi_i\rangle) = \frac{\exp(\langle\psi_i|\,\sigma_z\,|\psi_i\rangle)}{\sum_{i=1}^{3}\exp(\langle\psi_i|\,\sigma_z\,|\psi_i\rangle)}. \tag{4.5}$$

In Eqn. (4.5), $\langle\psi_i|\,\sigma_z\,|\psi_i\rangle$ gives the *Pauli-Z* measurement value on the qubit represented by $|\psi_i\rangle$. The predicted labels are obtained from the above *softmax* function using the output values from the qubits (where $i = 1, 2, \text{and } 3$ as three ancilla qubits are measured). We choose to update the circuit parameters by minimizing the *cross-entropy* loss between the predictions and the actual labels as defined by the following equation

$$J(\boldsymbol{\theta}) = -\sum_{n=1}^{3} y_n \log((p_n)_{\boldsymbol{\theta}}) \tag{4.6}$$

where $(p_n)_{\boldsymbol{\theta}}$ are the predicted probabilities obtained from the *softmax* function, over the gate parameters ($\boldsymbol{\theta}$), and y_n are the actual class labels ($n = 1, 2, \text{and } 3$ as each class label contains three elements). $\boldsymbol{\theta}$ represents all the gate parameters to be optimized. The gradient calculation used in our computational model is different from that used in classical machine learning. We used the analytic gradient [88] that is best suited for gate parameters update on NISQs to avoid the influence of noise. The analytic gradient is calculated as a derivative for the θ of the output value for that input. That is,

$$\partial_\theta f(\theta) = c[f(\theta + s) - f(\theta - s)]. \tag{4.7}$$

From Eqn. (4.7) it can be observed that the gradient value is the difference between the two output values of the circuit where f is the function to calculate the output value. The first value, $f(\theta + s)$ is the output of the circuit with the parameter θ increased by a value s, and the second value, $f(\theta - s)$ is the output of the circuit with the parameter θ decreased by a value s. Finally, the difference in the output values of the function is multiplied by a constant c, where c depends on s [88]. The given gradient calculation is not a finite differentiation. The essential idea of the gradient calculation is that the circuit is executed with the input and sampled to get the output value $f(\theta)$. Then, two more circuit evaluations $f(\theta + s)$ and $f(\theta - s)$ are carried out on the same circuit to get the gradient. Finally, the circuit is trained to minimize the cost of parameter updates using a classical computer. Since a classical computer is used for optimization, the approach is considered as a hybrid classical-quantum approach. In the following section, the details

of the experiments performed on numerical data using the proposed techniques are given with performance comparison.

4.3 Experiments on Numerical Data for Classification

In this section, we present the details of the experiments performed on numerical datasets using QC ANN for binary classification and QMCC for multi-class classification. We performed experiments using `default.qubit` simulator provided by *PennyLane* [89]. Experiments are performed on the Wisconsin Breast Cancer dataset using QC ANN. Multi-class benchmark numerical datasets for machine learning Iris [90], banknote authentication (BNA) [91], and wireless indoor localization [92] are classified using QMCC. Further, we present the performance evaluation of QC ANN and QMCC along with the major observations.

4.3.1 Description of the Numerical Datasets

The following benchmark numerical datasets for machine learning are used for the experiments. Breast Cancer Wisconsin (Original) dataset (WBCD) [93] is used to evaluate the performance of QC ANN. The datasets Iris [90], Banknote Authentication (BNA) [91], and Wireless Indoor Localization (WIL) [92] were used to evaluate the performance of QMCC.

WBCD: The WBCD from the UCI machine learning repository is a binary classification dataset. The attributes of the dataset consist of measurements for breast cancer cases. There are two class labels in the dataset *benign* and *malignant*. Dimensionality of the dataset is 32 with 569 instances of which 357 belongs to *benign* and 212 belongs to *malignant*.

Iris Dataset: The Iris dataset contains flower data with four attributes related to length and width of sepals and petals. Each class label in the dataset consists of 50 samples of Iris setosa, Iris virginica, and Iris versicolor. These measures were used to create a machine learning models to classify the species. The dataset is often used as a benchmark dataset for machine learning to test algorithms related to classification.

BNA Dataset: The BNA dataset involves data and attributes related to banknote to predict whether a given banknote is authentic. The numerical data given in the dataset is related to the features from the images of the banknotes. The dataset contains 1,372 rows with five numeric variables used for binary classification. BNA dataset is used in our experiments to demonstrate the fact that QMCC can also be used for binary classification.

WIL Dataset: The WIL dataset consists numerical data with signal strengths of seven

WiFi signals visible on a smartphone. There are 2000 rows and seven columns in the dataset with four rooms as class labels. We synthesized the dataset to suit our experiments on the designed quantum circuit.

4.4 Performance Analysis of QC ANN for Binary Classification

In this section, we present the results of the simulation performed to prove the efficiency of QC ANN. To implement classical ANN we used *keras* [94] library in *python*. QC ANN experimented on a quantum simulator provided by *PennyLane* [89]. *PennyLane* also provides methods [88] for optimizing rotational parameters of quantum gates with hybrid quantum-classical computations. All the experiments are performed on *Wisconsin Breast Cancer Diagnosis* (WBCD) dataset with two class labels.

Classical ANN designed for binary classification of WBCD consists of an *input layer* with 30 nodes, as the input dimension is 30. As the input values are 30, reduction in the the number of nodes is not possible in the input layer. Next, two *hidden layers* with 9 and 3 nodes, are used respectively. The output layer consists of 1 node as each data record is associated with a class label of the binary value. Hence, the total number of parameters to be optimized are 300 $[(30 \times 9) + (9 \times 3) + (3 \times 1) = 300]$. We used *sigmoid* activation function for determining the class label.

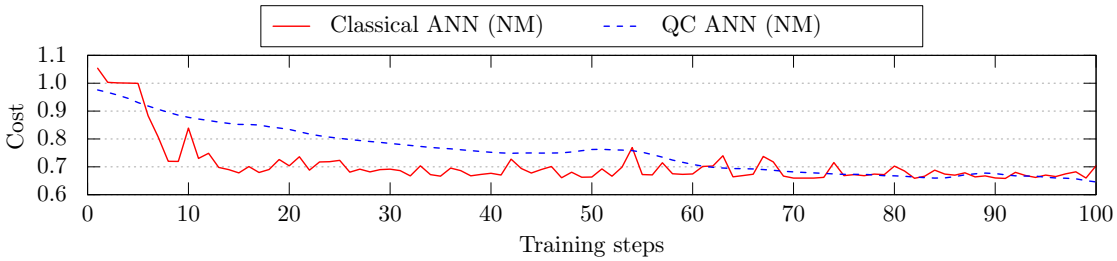


Figure 4.6: Cost Vs number of training steps of Classical ANN and QC ANN using NM.

Table 4.1: Details of hyperparameters used for training the networks in classical and quantum computing approach for ANN

Type	Value
Optimization (SGD)	Nestrov momentum (NM) and Adaptive momentum estimation (Adam)
Learning rate	0.01
Momentum	0.9
Kernel initialization	Standard normal distribution

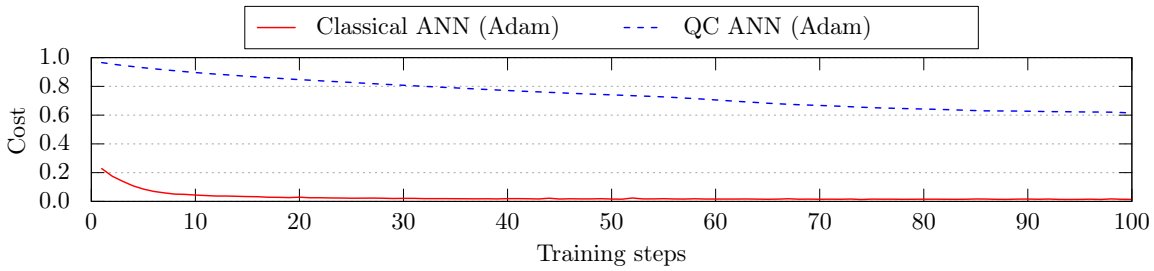


Figure 4.7: Cost Vs number of training steps of Classical ANN and QC ANN using Adam.

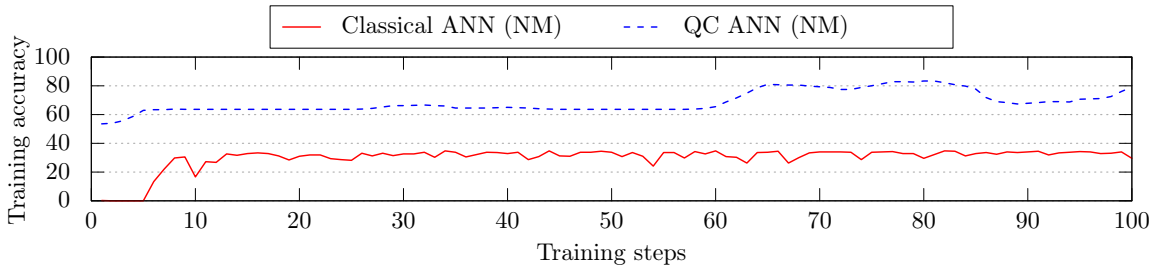


Figure 4.8: Training accuracy comparison of Classical ANN and QC ANN using NM.

The QC ANN is implemented using 5 qubits in the *input layer*. As there are 30 input values in the dataset, the values are embedded as amplitudes of quantum states. Therefore, it is required to use at least 5 qubits (as $2^5 = 32$). The amplitude embedding encoding scheme is used for the state preparation where the input data of 30 values are encoded as amplitudes of the quantum state with 32 superpositions. The other two values for the input are padded with constants to match the size of states in superposition. The amplitudes are represented by a state vector with a 32×1 dimension. The *hidden layer* consists of *Controlled NOT (CNOT)* gates for entanglement and $R_y(\theta)$ gates on each qubit. The $R_y(\theta)$ gate is a single-qubit rotation around $y - axis$ represented in the Bloch sphere (Fig. 1.3) through an angle of θ (radians). We used six such hidden layers and the last qubit (Fig. 4.2) is measured using *Pauli-Z* (σ_z) measurement for an expectation value. A *Pauli-Z* (σ_z) measurement operator on basis states is represented as $\langle 0| \sigma_z |0\rangle$ and $\langle 1| \sigma_z |1\rangle$. As the operations $\langle 0| \sigma_z |0\rangle$ and $\langle 1| \sigma_z |1\rangle$ give binary values 1 and -1, respectively, this measurement is suitable for binary classification of the input data. The total number of parameters to be optimized in this approach is 30 as there are six hidden layers with five single-qubit rotations in each layer.

The results show that QC ANN is outperforming the classical version of ANN for binary classification on *WBCD* dataset with a validation accuracy of 82.51% whereas the classical ANN is giving 39.86% validation accuracy using *Nesterov Momentum* (NM) [95].

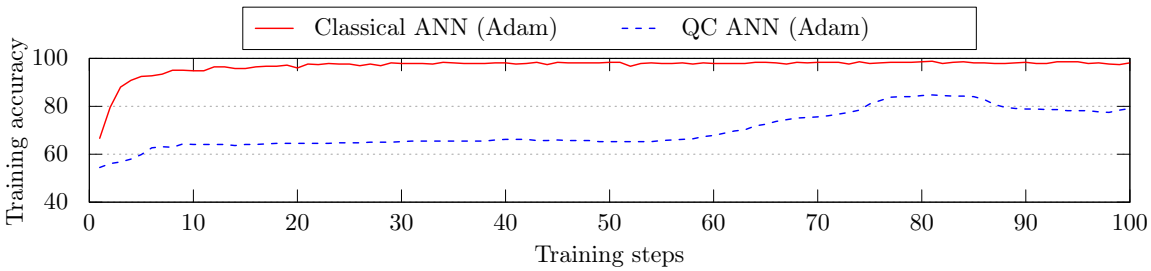


Figure 4.9: Training accuracy comparison of Classical ANN and QC ANN using Adam.

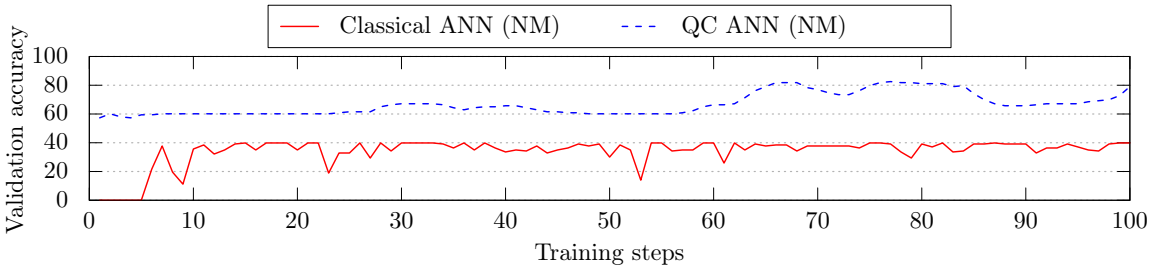


Figure 4.10: Validation accuracy comparison of Classical ANN and QC ANN using NM.

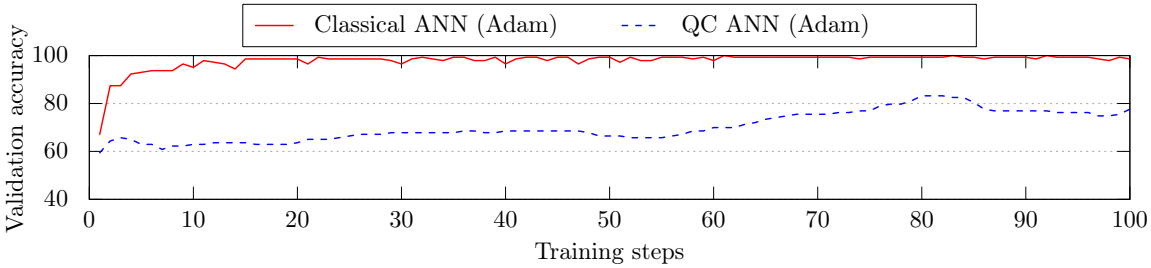


Figure 4.11: Validation accuracy comparison of Classical ANN and QC ANN using Adam.

Table 4.2: Accuracy comparison of Classical ANN and QC ANN on WBCD dataset.

Type	Classical ANN	QC ANN
Training steps	100	100
Parameters trained	300	30
Training accuracy (426 records)	34.09% (NM) 98.12% (Adam)	83.33% (NM) 84.74% (Adam)
Validation accuracy (143 records)	39.86% (NM) 98.60% (Adam)	82.51% (NM) 83.21% (Adam)

However, Classical ANN is giving good accuracy over QC ANN using *Adaptive Momentum Estimation* (Adam) [96]. Both NM and Adam are Stochastic Gradient Descent (SGD) optimization techniques. Comparatively, QC ANN is able to perform well with very few parameters.

Table 4.1 shows the details of hyperparameters used to train the models and a comparison of accuracy is given in Table 4.2. In Fig. 4.6, a comparison of the training cost for both classical ANN and QC ANN models using NM is provided. Mean-squared error is used as the cost function. The transition of the cost curve for the QC ANN is smooth, thus indicating that the model is able to learn the features from the data and can generalize the characteristics of the data efficiently in the model. However, the training cost for classical ANN is significantly less when compared to QC ANN using Adam, indicating that tuning hyperparameters is an important task in training an artificial neural network.

In Figs. 4.8 and 4.9, the training accuracy of both the models is compared. Figs. 4.10 and 4.11 show the plots for validation accuracy. We observe that with good training accuracy and validation accuracy, QC ANN is performing extremely well even when only fewer parameters are trained. Thus, using a quantum computer, we can train an ANN efficiently.

Quantum computers have vast potential to solve many complex problems that are difficult for a classical computer to solve. In this work, we show that the quantum computing approach for training ANN on a quantum computer performs exceptionally well when compared with classical ANN. QC ANN used only a few parameters to learn the complex and non-linear patterns in data. Hence, it is computationally efficient. The training approach used in this work can also be extended to train deep neural network models for

image classification. Also, the approach using a quantum computer in training a deep neural network is considered efficient in terms of computational space as we require only k qubits to represent 2^k input values of classical data. Thus, we achieved good results using a few qubits and fewer parameters for optimization.

4.5 Performance Analysis of QMCC for Multi-Class Classification

In this Section, we present the details of our experiments and quantum simulations. We experimented with the state preparation on the real quantum hardware provided by IBM on the IBM QX platform. The entire QMCC variational circuit is implemented on the `default.qubit` simulator provided by *PennyLane*.

IBM provides access to real quantum hardware through *Qiskit* [97] quantum computing framework on the IBM QX platform. The unitary operations for the state preparation are executed on the IBM QX platform using *Pennylane-Qiskit* [98] plugin. *Pennylane-Qiskit* plugin provides necessary packages to integrate *Qiskit* with *PennyLane*'s quantum machine learning capabilities. Hence, we can execute a Python program with quantum operations written using *PennyLane* package on realtime quantum hardware. The following sample code snippet is used to access the real quantum hardware.

```
import pennylane as qml
dev = qml.device('qiskit.ibmq', wires = 4,
backend = 'ibmq_16_melbourne', ibmqx_token = "XXX")
```

The method `qml.device` is used to initialize a quantum device or a quantum simulator for performing quantum operations with the number of qubits given as input to `wires`. The input parameter, `qiskit.ibmq`, is used to indicate that the quantum hardware on the IBM QX platform is used to perform quantum operations. The selection of quantum hardware for experimentation is given as input to `backend`. The IBM QX API token, necessary to perform quantum operations, can be obtained by creating a profile on the IBM QX. The token is given as input to `ibmqx_token`. All the qubits on the selected quantum hardware are initialized to $|0\rangle$ by default. After the selection of IBM's quantum device, `qml.Hadamard` and `qml.QubitUnitary` operations are used to create the state with the unitary operations as mentioned in Eqn. 4.2 and Eqn. 4.3, respectively. The results for the state preparation experiment on real quantum hardware are given in Section 4.5.

The entire QMCC variational circuit is implemented as a quantum node on the qubit simulator named `default.qubit`. The quantum node with one layer is programmed in

five modules: *State preparation*, *Gates on data qubits*, *Entanglement* Gates on ancilla qubits, and *Measurement*. The decorator `@qml.qnode(dev)` is used for creating a quantum node on the simulator. Gates in the variational circuit are implemented using `qml.RY`, `qml.RZ` operations and the entanglement between the qubits are created using `qml.CNOT` operation. The *measurement* operation on the ancilla qubits is performed on each wire using `qml.exp.PauliZ(n)`, where `n`$= 1, 2, \ldots k$ is the label of the wire. The output values from the measurement are passed through *softmax* function provided in Eqn. 4.5. The model is trained and the gate parameters are optimized classically using the optimization techniques `NesterovMomentumOptimizer` and `AdamOptimizer`, provided in the package `pennylane.optimze`, such that the model performs with the desired accuracy. *PennyLane* provides necessary interfaces to perform hybrid classical-quantum operations together on the classical computing facility and the quantum simulator. The details of the results for the simulations, along with discussions, are given in the following sections.

Dataset and preprocessing

The simulations and experiments were carried out using three benchmark datasets for machine learning algorithms. The datasets we used are the following: (i) *Iris dataset*, (ii) *Banknote Authentication (BNA) dataset*, and (iii) *Wireless Indoor Localization (WIL) dataset*. The *Iris dataset* consists of three class labels and 150 instances, *BNA dataset* consists of two class labels with 1372 instances, and the *WIL dataset* consists of three class labels with 1500 instances. *Min-Max scaling* is used on the datasets and then the values are rescaled between $[0, \frac{\pi}{2}]$ for R_y rotations. As stated before, *one-hot encoding* is used to relabel the class names. A single data vector from one of the 150 instances from *Iris dataset* is considered for the state

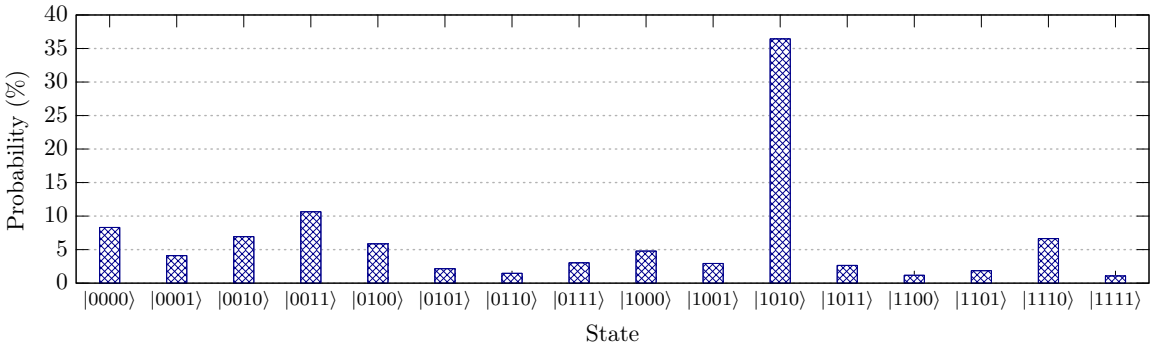


Figure 4.12: Experimental result of state preparation scheme on IBMQX platform with probabilities calculated as $|| \, |\psi\rangle \, ||_2^2$.

preparation experiment on *ibm_16_melbourne* [99] real quantum hardware on IBM QX. The

designed unitary matrix converts the initial state of a qubit to the desired state using qubit encoding scheme. The input consists of four elements from a data vector of *Iris dataset* and Fig. 4.12 shows the state obtained after the state preparation as a superposition of all the basis states as follows:

$c_1 |0000\rangle + c_2 |0001\rangle + c_3 |0010\rangle + c_4 |0011\rangle + c_5 |0100\rangle + c_6 |0101\rangle + c_7 |0110\rangle + c_8 |0111\rangle + c_9 |1000\rangle + c_{10} |1001\rangle + c_{11} |1010\rangle + c_{12} |1011\rangle + c_{13} |1100\rangle + c_{14} |1011\rangle + c_{15} |1110\rangle + c_{16} |1111\rangle$

where $c_1, c_2, \ldots, c_{16}$ are the amplitudes of the quantum states in superposition.

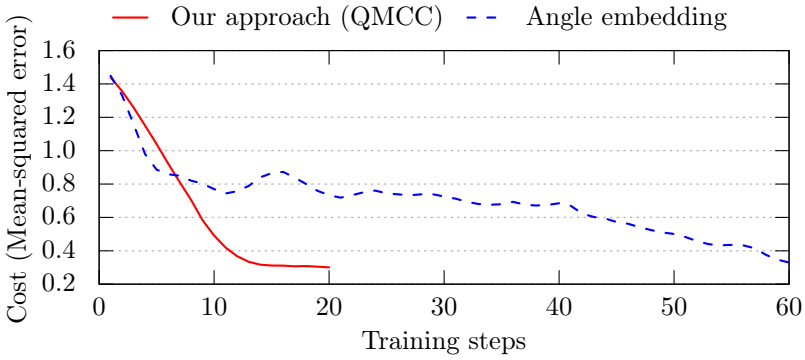


Figure 4.13: Cost plot for our approach (QMCC) and the model using angle embedding scheme.

The encoding scheme needs more computational space as it requires n qubits to encode a N-dimensional data-vector. However, as our approach uses only single-qubit rotations, the encoding scheme is efficient in terms of time. A comparison study between our approach and the model developed with angle embedding scheme [25] is carried out on a binary classification task using classes 1 and 3 in the *Iris dataset*. The class declaration is carried out based only on *Pauli-Z* measurement of a selected qubit. The parameters are updated by minimizing the *mean square error*.

Due to the ease of preparing a state as $\cos(x_n^d) |0\rangle + \sin(x_n^d) |1\rangle$, all the four features from each data vector of *Iris dataset* are encoded on each of the qubits. The model with an angle embedding scheme used two features and the state preparation consists sequence of multiple single-qubit rotations. As only two features are encoded on the angle embedding model, the cost for our approach is comparatively less as shown in Fig. 4.13. The cost for the angle embedding model with two features is reduced after 60 training steps. Whereas,

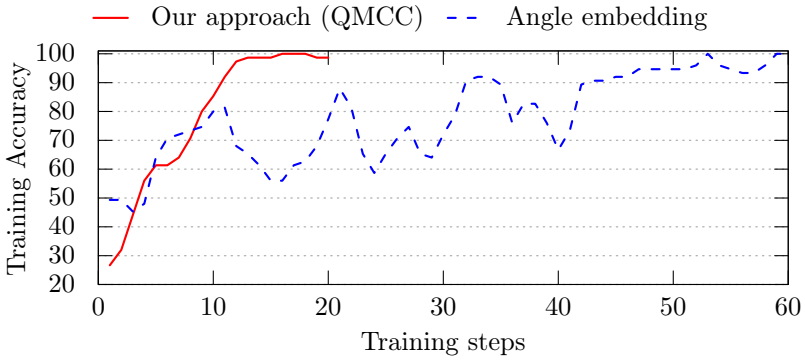


Figure 4.14: Training accuracy for our approach (QMCC) and the model using angle embedding scheme.

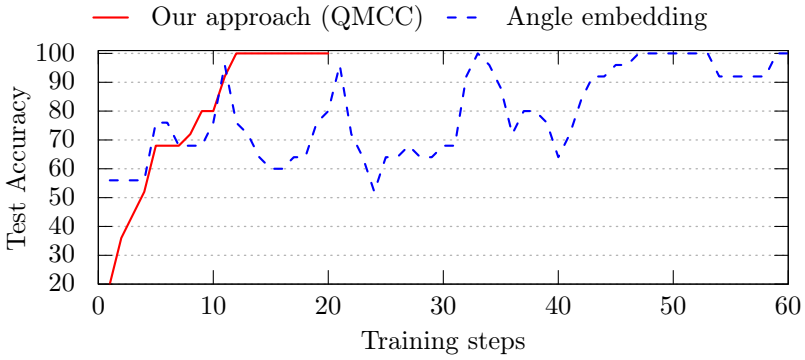


Figure 4.15: Test accuracy for our approach (QMCC) and the model using angle embedding scheme.

from Fig. 4.14 and Fig. 4.15 it is clear that the model using our approach for state preparation is able to learn fast in 20 training steps with four features encoded as a quantum state.

Variational Circuit and training

The circuit after state preparation consists of layered architecture as shown in Fig. 4.3. Each layer is configured with single qubit and two qubit gates. For *Gates on data qubits* part, parameterized $R_y(\theta)$ rotations are used to allow only real rotation of qubits. We used three $R_y(\theta)$ rotations in sequence for each of the first four qubits. The *Entanglement* part of the circuit is implemented using *CNOT* gates. Later, the ancilla qubits are passed through a sequence of $R_z(\theta)$, $R_y(\theta)$, and $R_z(\theta)$ each in *Gates on ancilla qubits* part of the layer. We used seven such layers. Finally, three ancilla qubits are measured individually using *Pauli-Z* measurement, and the obtained expectation values are given as input to the

softmax function to obtain predicted labels. At each training step t, we calculate the analytic gradient as in Eqn. (4.7).

On *Iris dataset*, cost value optimization is performed using Nesterov momentum optimization with parameters updated using the following equation

$$\boldsymbol{\theta}^{(t+1)} \leftarrow m\boldsymbol{\theta}^{(t)} + \eta^{(t)}\nabla J^{(t)} \tag{4.8}$$

where m is momentum that shifts the current input, η is step size, and $\nabla J^{(t)}$ is the analytic gradient. In our simulations, we used $m = 0.9$, $\eta = 0.01$ and the gate parameters ($\boldsymbol{\theta}$) are initialized using standard normal distribution. Cost value optimization on *BNA dataset* and *WIL dataset* is performed using Adam optimizer [100] with $\beta_1 = 0.9$, $\beta_2 = 0.9$, and $\eta = 0.01$.

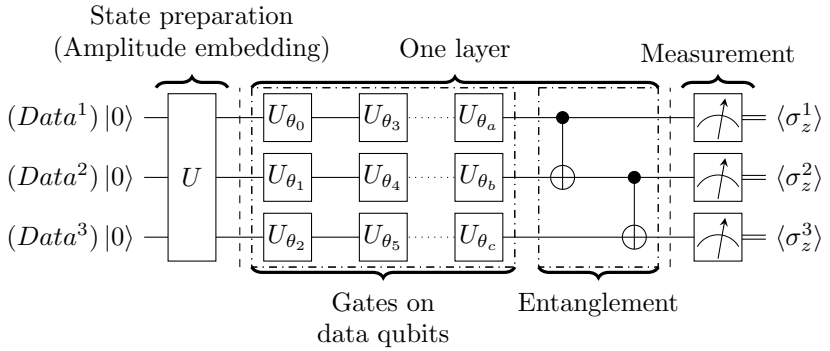


Figure 4.16: Illustration of amplitude embedding model (AEM) with one layer for classification of the three classes. The original implementation of the model consists a total of seven similar layers for comparison with our QMCC model.

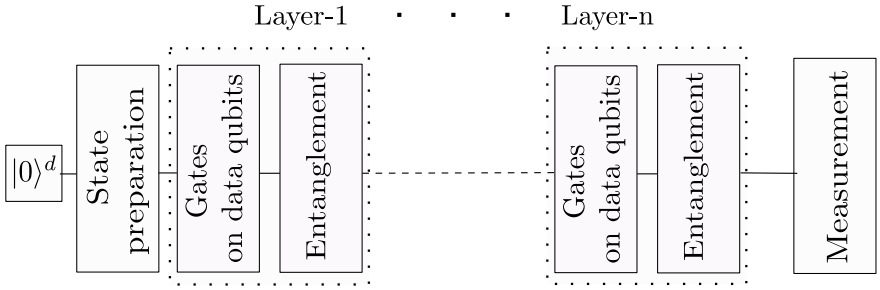


Figure 4.17: Generalized architecture of amplitude embedding model with n-layers.

The QMCC model is trained for 100 iterations to minimize the *cross entropy* loss on all the three datasets. Datasets are partitioned into 75% of data instances as training set and 25% are used for testing the model. We used the early-stopping technique [101] in

training to avoid over-fitting of the data. All numerical simulations are preformed on qubit simulator provided by *Pennylane*.

The results obtained from the QMCC model are compared with the results from the amplitude embedding model (AEM) built using amplitude embedding [102, 24] for the state preparation. For the comparison study, we made certain changes to our model, as shown in Fig. 4.16. The state preparation scheme in QMCC is replaced with amplitude embedding and the layer structure is the same as in QMCC with $R_y(\theta)$ gates followed by entanglement using *CNOT* gates. The measurement process for AEM is different from QMCC. As there are no ancilla qubits, only data qubits are measured. The generalized architecture for the amplitude embedding model is shown in Fig 4.17. The remaining process after measurement for class prediction using *softmax* and *cross entropy* loss minimization is the same as QMCC. The results presented in Table 4.3 show the comparison between QMCC model and AEM on all the three datasets.

Table 4.3: Comparison of QMCC and AEM

Type	AEM	QMCC
State preparation	Amplitude embedding	Designed unitary
Layers	7	7
Rotations on data qubits	R_y	R_y
Rotations on ancilla qubits	—	R_z, R_y, R_z
Training steps	100	100
Training accuracy on Iris dataset	58.92%	87.50%
Test accuracy on Iris dataset	57.89%	92.10%
Training accuracy on BNA dataset	73.17%	88.53%
Test accuracy on BNA dataset	69.67%	89.50%
Training accuracy on WIL dataset	46.84%	89.60%
Test accuracy on WIL dataset	46.93%	91.73%

The cost curve plot for *Iris dataset* with 150 instances is given in Fig. 4.18. The loss curves in the Fig. 4.18 represents the *cross-entropy* loss between the predicted labels and the actual labels of *Iris dataset* for 100 training steps. Training accuracy plot for *Iris dataset* given in Fig. 4.19 shows that our QMCC model can generalize the data in the early stage of the training process. The QMCC model shows the best training accuracy of 87.50%, and AEM shows the best training accuracy of 58.92%. In the testing phase, as given in Fig. 4.20, QMCC model classified *Iris dataset* into three class labels with the best test accuracy of 92.10% where as the best test accuracy for the AEM is 57.89%.

On *BNA dataset* with 1372 instances and two class labels, the QMCC model shows

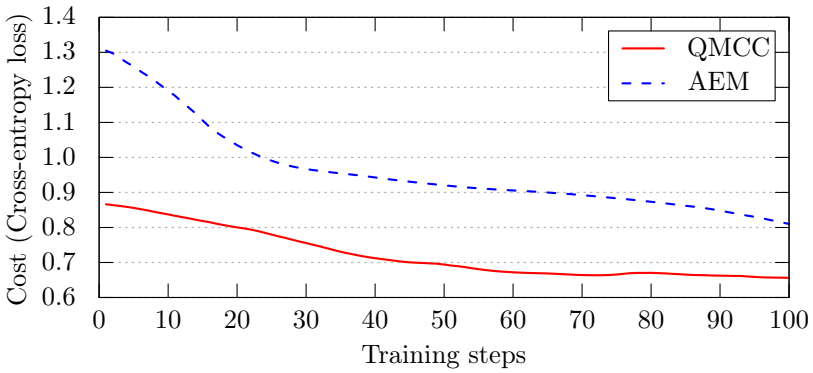


Figure 4.18: Cost comparison of QMCC and AEM on Iris dataset.

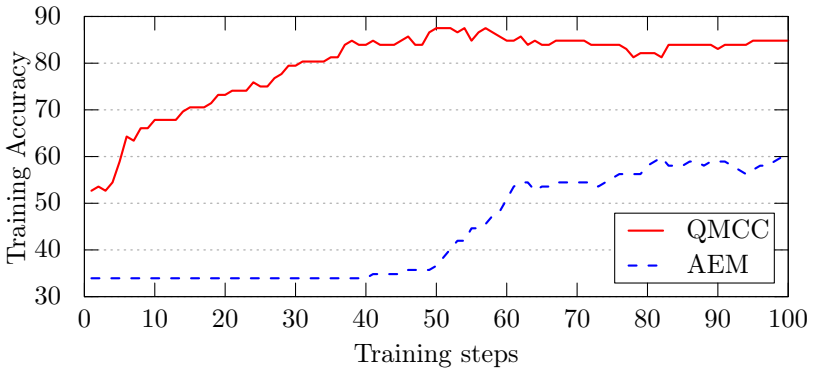


Figure 4.19: Comparison of training accuracy between QMCC and AEM on Iris dataset.

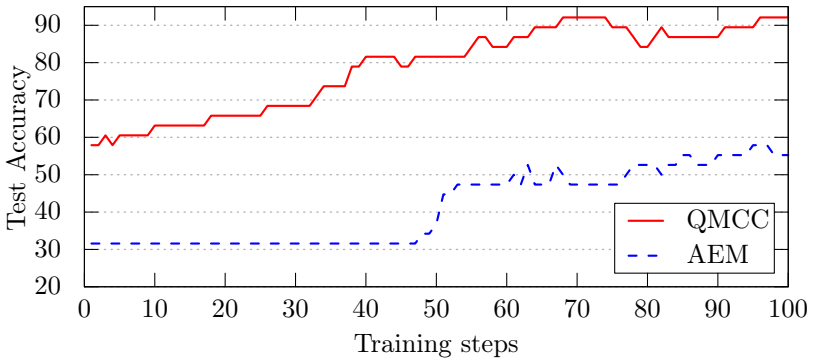


Figure 4.20: Comparison of test accuracy between QMCC and AEM on Iris dataset.

better performance than AEM. The cost curve plots for QMCC model and AEM are given in Fig. 4.21. The training loss for the QMCC model was reduced after 80 training steps in comparison with the AEM model. After the first ten training steps, the QMCC model shows the best training accuracy of 88.53%, whereas AEM shows the best training accuracy of 73.17% only. The training accuracy plot for both the models is given in Fig. 4.22. The

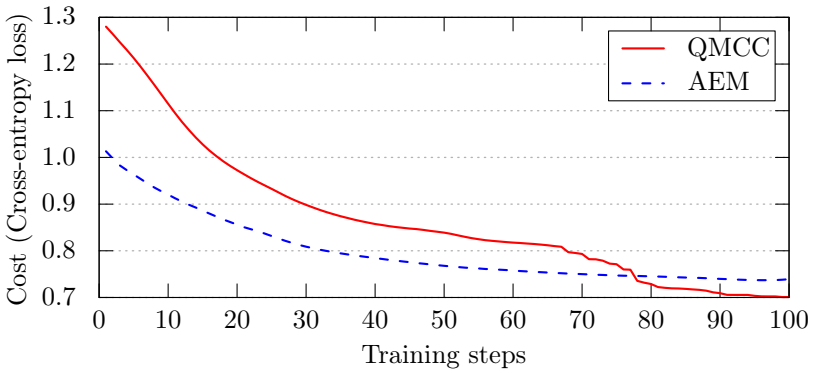


Figure 4.21: Cost comparison between QMCC and AEM on BNA dataset.

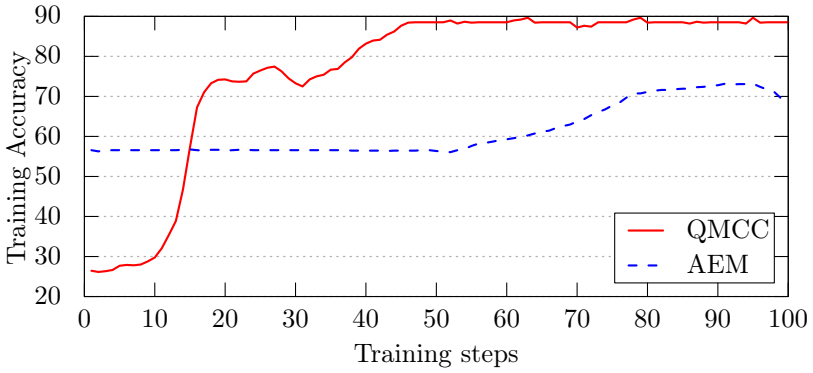


Figure 4.22: Comparison of training accuracy between QMCC and AEM on BNA dataset.

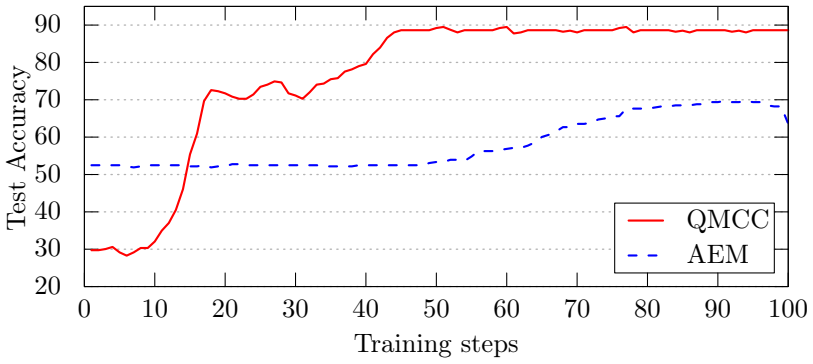


Figure 4.23: Comparison of test accuracy between QMCC and AEM on BNA dataset.

best test accuracy for the QMCC model is 89.50% and, similarly, the best test accuracy of AEM is only 69.67% on *BNA dataset* as can be seen from Fig 4.23.

The *WIL dataset* is preprocessed and 1500 instances with four attributes and three class labels are used for our quantum simulations. The training loss for the QMCC model is very less when compared to the AEM, as given in Fig. 4.24. In Fig. 4.25, the training accuracy

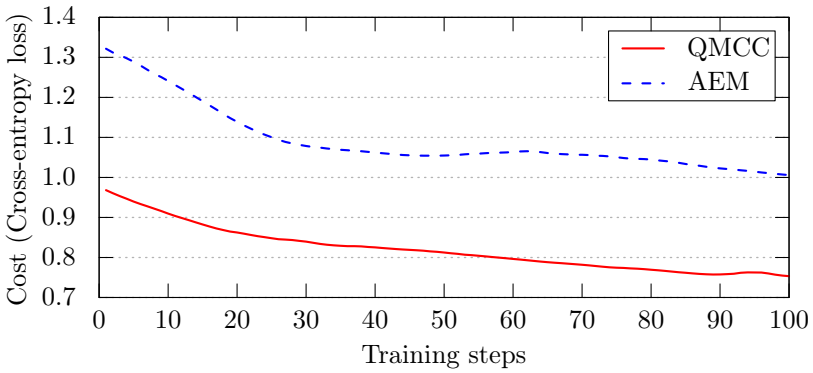


Figure 4.24: Cost comparison between QMCC and AEM on WIL dataset.

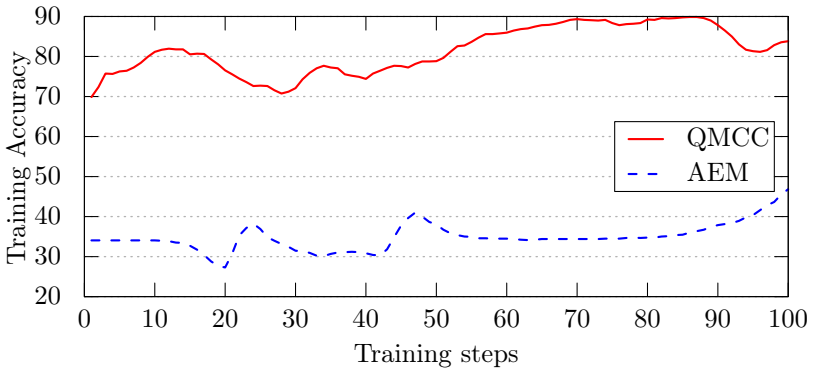


Figure 4.25: Comparison of training accuracy between QMCC and AEM on WIL dataset.

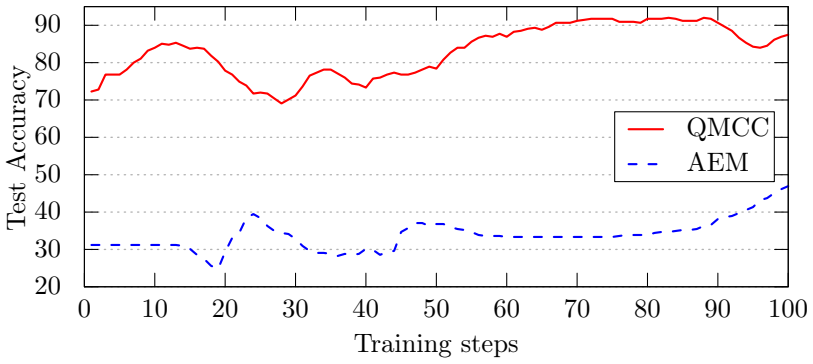


Figure 4.26: Comparison of test accuracy between QMCC and AEM on WIL dataset.

curve for the QMCC model shows that the model can learn the patterns in the input with the best training accuracy of 89.60%. The best training accuracy of AEM is only 46.84%. QMCC model can classify the larger dataset with the best test accuracy of 91.73%, thereby, outperforming AEM. The test accuracy plot is given in Fig. 4.26, whereas, the best test accuracy for AEM is 46.93% only.

The above results show that the QMCC model outperforms the AEM on all three datasets. The proposed model can also be extended to classify datasets with more than three class labels based on the availability of qubits for processing.

Quantum computing is considered to be one of the possible future options for dealing with highly complex data processing problems that cannot be dealt with classical computing approach. Quantum computers can represent data as high-dimensional superposition in a quantum state. Hence, quantum circuits can be used to build hybrid models that detect features in data using both classical and quantum operations. The huge amounts of data existing today, can be handled efficiently using a quantum computer due to the use of qubits for data processing.

In this work, we proposed a Quantum Multi-Class Classifier (QMCC) using the variational circuit approach for classification on a quantum computer. Existing quantum classifiers perform only binary classification. The results obtained from experiments conducted on IBMQX show that the unitary matrix designed for the state preparation is compatible with the real quantum hardware. Our QMCC approach stands unique as we used separate data qubits to encode classical data and ancilla bits for measurement. The proposed QMCC model uses classical machine learning techniques such as *softmax* function and *cross entropy* loss effectively to obtain results. The QMCC model is implemented on a qubit simulator with seven qubits considering the dimension of the datasets used for the quantum machine learning simulations. Alternatively, our approach can be expanded to classify multiple classes considering the availability of qubits. We studied our QMCC model using the three datasets, *Iris dataset*, *BNA dataset*, and *WIL dataset* and obtained classification accuracy of 92.10%, 89.50%, and 91.73%, respectively.

4.6 Major Observations

The performance of the hybrid models on benchmark numerical datasets proves the efficacy of quantum computing in enhancing machine learning algorithms. Even though both the hybrid models are performing well on numerical datasets, the architecture of the models is not entirely suitable for analyzing spatial big data, especially satellite images. The major observations on the two models are as follows:

1. Data encoding is an important step for any quantum computer application. However, there exists no single encoding technique that suits all data types.

2. Non-linearity is not present in the quantum circuits of the hybrid models as the same

qubits and, therefore, the same operations are used in all layers.

3. Depth of the quantum circuits increases drastically as more layers are added when processing large amounts of data.

4. Efficient hybrid models are required to handle huge amounts of data such as image data, as the data consists of huge dimensionality.

As per the observations above, handling spatial big data is difficult with the same circuit design and implementation process. Hence, as the next step, we propose and study hybrid models to address the problem of non-linearity in the parameterized quantum circuits and develop models to enhance the training process of machine learning models.

4.7 Summary

This chapter presents two models designed for supervised learning on numerical data. We first proposed a quantum computing approach for ANN to study the impact of applying quantum computing techniques. We extended the idea by designing a new quantum circuit and a data encoding process for a multi-class classification circuit. Our results show that QC ANN performed binary classification with an accuracy of 82.51% on WBCD. Also, multi-class classification is performed using QMCC on three benchmark datasets for machine learning iris dataset with an accuracy of 92.10%, BNA dataset with an accuracy of 89.50%, and WIL dataset with an accuracy of 91.73%.

Chapter 5

Hybrid QML Architectures for Spatial Data Analytics

"No problem is too small or too trivial if we can really do something about it."

– RICHARD FEYNMAN

Spatial data refers to any data with a spatial identifier that refers to a position on the earth. Spatial data is essential and can be analyzed using machine learning algorithms. The following sections provide information about spatial data and its analysis. There exist several sources for spatial data. Some of these sources include: GPS, GPS-enabled devices, satellite remote sensing, aerial surveying, radar, LiDAR, sensor networks, digital cameras, and RFIDs. Data from all such sources give various opportunities to explore and discover useful information.

Spatial data analysis has a wide range of applications [103, 104, 105, 106, 107, 108, 109, 110, 111]. Analyzing Spatial data can be very advantageous to different sectors [3]. Considering the insurance industry, optimal portfolios can be designed for dynamic insurance pricing based on different features ranging from topographical features to social risks in a particular geographical area. Emergency responders can be positioned in a particular area during natural havoc base on real-time spatial data available from the affected area. Images from drones can be analyzed for taking different decisions. In the agriculture sector, food production can be improved by giving proper instruction to farmers after analyzing weather data. In the commercial sector, retailer analytics can be carried out to find customer behavior as a function of spatial regions thereby, making business decisions. In our work, we focus on satellite image scene classification.

In the previous chapter, we highlighted the limitations of existing hybrid models in handling spatial data. In this chapter, we performed the work to overcome the limitations of the existing models and also to enhance the machine learning models. The main focus is

to process huge amounts of spatial data, particularly satellite images using hybrid models and enhance existing deep learning models. The two scenarios presented in this chapter allow for the processing of spatial data efficiently using hybrid models. The proposed models reduce the training parameters of the deep learning models, thereby, enhancing the training process. In the following, we explain the details of the designed models.

5.1 Quantum-Enhanced Deep Neural Networks Architecture

Deep learning algorithms gained prominence in analyzing images for real-time applications such as detection of objects, segmentation of instances, semantic segmentation, and classification of image scenes. However, deep learning models for image classification, such as convolutional neural networks (CNNs), require extensive computational facilities. Also, training such models with multiple layers becomes complex as many trainable parameters are to be optimized. Quantum computing emerged as a research area to handle complex problems using quantum-mechanical properties for computation on a quantum computer. In this work, we primarily focus on designing a hybrid quantum-classical deep learning model for image scene classification. We propose a novel hybrid architecture that uses quantum computation for feature extraction and classical computation for scene classification. In the hybrid architecture, we use quantum measurement-based features to obtain the quantum representations of images. The obtained quantum representations of images are used to train and build a classical deep learning model for image scene classification. Our experiments performed on *ibm_santiago* quantum computer shows that the proposed model is suitable for implementation on noisy intermediate scaled quantum computers (NISQs). Our experimental results show that the proposed three-layered hybrid architecture models can classify data efficiently using trainable parameters approximately 27%–30% less than the state-of-the-art models on satellite image datasets. Hence, the complexity of training the deep learning models reduces as the number of parameters to be optimized reduces. Using the designed architecture, the deep learning model can classify data with an overall accuracy of 95.89%, 86.13%, and 79.32% on UC Merced Land-Use [112], AID [113], and NWPU-RESISC45 [114] datasets, respectively for image scene classification.

The major contributions of our work are as follows:

- Designed a three-level hybrid quantum-classical architecture for the multi-class scene classification of satellite images using quantum computation and classical computa-

tion together.

- Extracted quantum measurement-based features from satellite images to obtain quantum representations of images by measuring qubits in a quantum circuit. The quantum circuit is suitable to implement on present-day NISQs or a quantum simulator.
- Performed experiment on *ibm_santiago* quantum computer provided online by *IBM Q Experience* [8], to prove the feasibility of implementation of the method to obtain quantum representations on NISQs.
- Quantum representations of the images are used to design and build fully connected deep neural network models, and such models are trained with minimum computational resources.
- Detailed analysis of the results using hybrid quantum-classical models compared with the state-of-the-art deep learning models for image scene classification on three benchmark datasets.

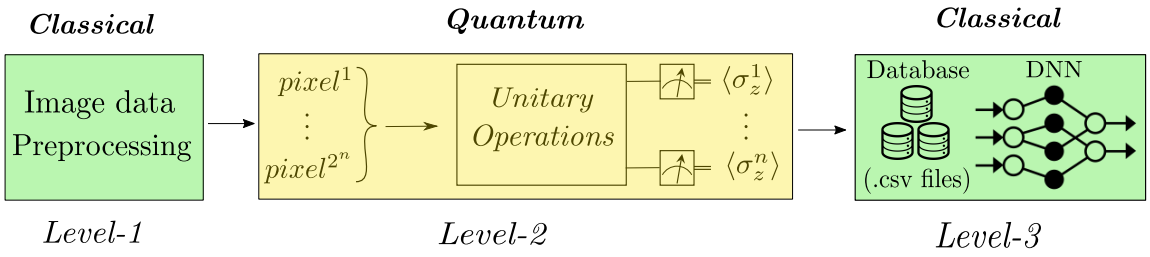


Figure 5.1: Three-level hybrid quantum-classical architecture.

5.1.1 Three-Level Hybrid Quantum-Classical Architecture

The proposed hybrid quantum-classical architecture (HybridQC) consists of three satellite image scene classification levels. In *Level-1*, image data preprocessing is performed to preprocess the images and make them suitable for giving as input to the quantum circuit in *Level-2*. The pixel values of each row from the input images are processed by a set of unitary operations in *Level-2* as shown in Fig 5.1. The qubit measurement output values from *Level-2* are stored as comma-separated values (*.csv files*) for every image. Each row of a *.csv file* contains the information [*Image id, Measurement values, Class label*] for every

image in a dataset. A classical deep learning model is developed using a fully connected deep neural network (DNN) in the *Level-3*. The model is trained using the quantum image representations stored in the *.csv files*. The details of the operations performed at each level are given in the following sections.

Level-1: Image Data Preprocessing

The first step of the image data preprocessing in *Level-1* is to resize each satellite image from datasets to the nearest power of 2. For example, an image of size 600×600 is resized to 512×512 because 512, which is 2^9, is the nearest power of 2 for 600. Every image is converted from the original Red, Green, and Blue (RGB) color space to a grayscale in the second step. The two steps are crucial to facilitate the qubit encoding scheme used in the *Level-2* for encoding image data into qubits to process the pixel values and extract quantum measurement-based features.

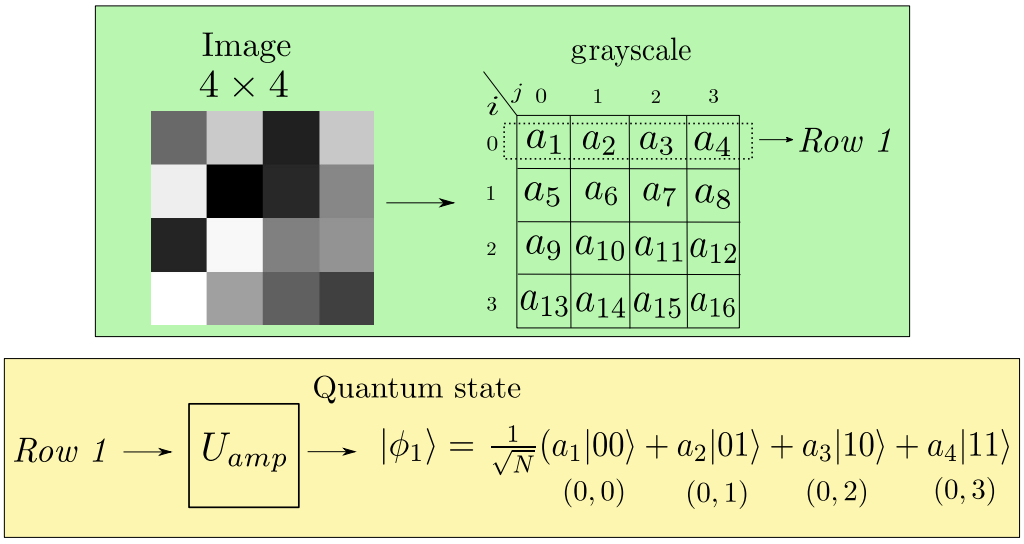


Figure 5.2: Illustration of grayscale and position of pixel to quantum state mapping.

Level-2: Quantum Representations of Images

After preprocessing in the *Level-1*, quantum measurement-based features are extracted from the images in the *Level-2* to create quantum representations of images. As a quantum computer uses qubits to process the data, images are encoded into qubits using amplitude embedding scheme [24]. The controlled rotations of R_y and R_z gates, along with $CNOT$

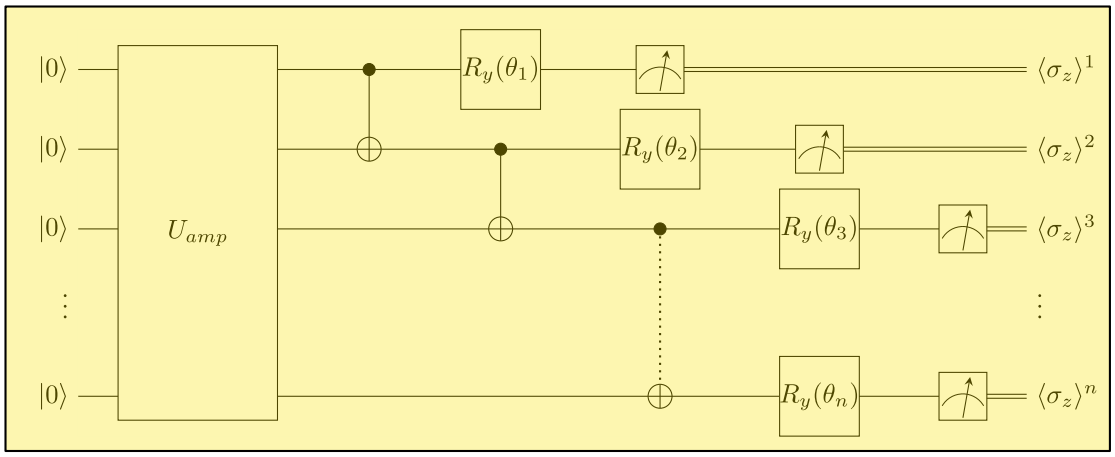


Figure 5.3: Illustration of quantum circuit for an input of 2^n values.

(Controlled-NOT) gates, are used to encode the pixel data as amplitudes of the superposition of quantum states. The idea of amplitude embedding is to transform the initial state of a quantum system with n qubits into the desired state $|\psi_n\rangle$ using the uniformly controlled rotations. Using the amplitude embedding scheme, 2^n normalized classical data points are represented as amplitudes of a n-qubit state $|\psi_n\rangle$:

$$|\psi_n\rangle = \sum_{i=0}^{2^n} a_i \, |i\rangle \tag{5.1}$$

where a_i is the i^{th} data point and $|i\rangle$ the i^{th} computational basis state. Thus, the exponential advantage is obtained as 2^n pixel values are encoded using n qubits.

As the image data is encoded into n qubits, the pixel values are represented in a Hilbert space. The state $|\phi_1\rangle$ in Fig. 5.2 represents the grayscale value and position of the pixel to be represented by the quantum state for the 4×4 image where U_{amp} contains the unitary operations [24] for amplitude embedding. For example, if $\{a_1, a_2, a_3, a_4\} = \{1, 2, 3, 4\}$, then the normalized values $\{\frac{1}{\sqrt{30}}, \frac{2}{\sqrt{30}}, \frac{3}{\sqrt{30}}, \frac{4}{\sqrt{30}}\}$ are encoded into quantum states as amplitudes using a quantum circuit. The superposition of quantum states after the U_{amp} operations is:

$$|\phi_1\rangle = \frac{1}{\sqrt{30}}|00\rangle + \frac{2}{\sqrt{30}}|01\rangle + \frac{3}{\sqrt{30}}|10\rangle + \frac{4}{\sqrt{30}}|11\rangle \tag{5.2}$$

Thus, input values are encoded into a quantum state using amplitude embedding. The entire quantum circuit used to extract the quantum representations from an image of size $2^n \times 2^n$, is given in Figure 5.3. In the quantum circuit, U_{amp} is used for generating the

amplitude embedded state. After the U_{amp} operation, the CNOT gate is operated on every pair of qubits for strong entanglement between all the qubits in the quantum circuit. The entanglement operation is followed by $R_y(\theta_i)$ operation on the qubits, where i represents the qubit number in the quantum circuit. For a systematic operational procedure, an average of a set of input pixel values is considered as θ_i. However, any value of θ_i ($\in \mathbb{R}$) can be used in the circuit.

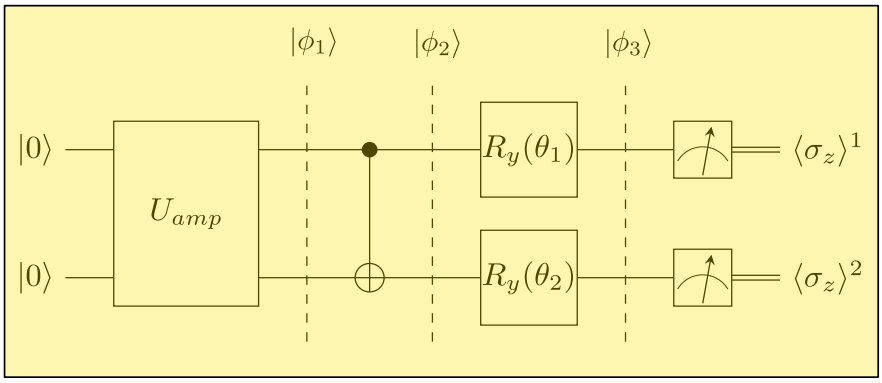


Figure 5.4: Illustration of quantum circuit for an input of 2^2 values.

The complete sequence of the unitary operations on the input pixel data is described below. For simplicity, we use *Row 1* of the grayscale from Fig. 5.2 as input to the quantum circuit given in Fig. 5.4.

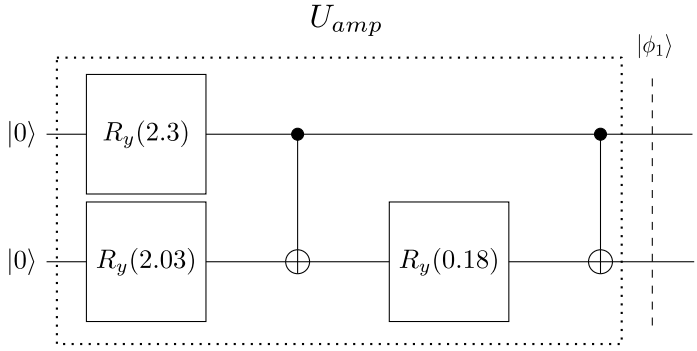


Figure 5.5: Illustration of amplitude encoding.

For example, if $\{a_1, a_2, a_3, a_4\} = \{1, 2, 3, 4\}$, then the normalized values

$$\left\{\frac{1}{\sqrt{30}}, \frac{2}{\sqrt{30}}, \frac{3}{\sqrt{30}}, \frac{4}{\sqrt{30}}\right\}$$

are encoded into quantum states as amplitudes using the quantum circuit given in Fig. 5.5. The superposition of quantum states after the U_{amp} operations is:

$$|\phi_1\rangle = \frac{1}{\sqrt{30}}|00\rangle + \frac{2}{\sqrt{30}}|01\rangle + \frac{3}{\sqrt{30}}|10\rangle + \frac{4}{\sqrt{30}}|11\rangle. \tag{5.3}$$

Thus, the input values are encoded into a quantum state using amplitude embedding. There exist different qubit encoding methods used in different contexts. Encoding classical data values into qubits is a very important step in quantum processing.

After applying U_{amp} on *Row-1* of the grayscale, the values a_1, a_2, a_3, and a_4 are embedded into amplitudes of quantum state in superposition as:

$$|\phi_1\rangle = \frac{1}{\sqrt{N}}\left(a_1|00\rangle + a_2|01\rangle + a_3|10\rangle + a_4|11\rangle\right) \tag{5.4}$$

where $N = (a_1)^2 + (a_2)^2 + (a_3)^2 + (a_4)^2$ is the normalization constant. Next, the state is acted upon by $CNOT$ gate and transforms $|\phi_1\rangle$ to $|\phi_2\rangle$:

$$|\phi_2\rangle = \frac{1}{\sqrt{N}}\left(a_1|00\rangle + a_2|01\rangle + a_4|10\rangle + a_3|11\rangle\right). \tag{5.5}$$

Now, $R_y(\theta)$ rotations acted on both the qubits with $\theta_1 = (\frac{a_1+a_2}{2})$ and $\theta_2 = (\frac{a_3+a_4}{2})$, transforms $|\phi_2\rangle$ to $|\phi_3\rangle$:

$$|\phi_3\rangle = A_{00}|00\rangle + A_{01}|01\rangle + A_{10}|10\rangle + A_{11}|11\rangle, \tag{5.6}$$

where

$$\begin{pmatrix} A_{00} \\ A_{01} \\ A_{10} \\ A_{11} \end{pmatrix} = \frac{1}{\sqrt{N}} \begin{pmatrix} \left(a_1\cos\frac{\theta_1}{2} - a_3\sin\frac{\theta_1}{2}\right)\cos\frac{\theta_2}{2} - \left(a_2\cos\frac{\theta_1}{2} - a_4\sin\frac{\theta_1}{2}\right)\sin\frac{\theta_2}{2} \\ \left(a_1\cos\frac{\theta_1}{2} - a_3\sin\frac{\theta_1}{2}\right)\sin\frac{\theta_2}{2} + \left(a_2\cos\frac{\theta_1}{2} - a_4\sin\frac{\theta_1}{2}\right)\cos\frac{\theta_2}{2} \\ \left(a_1\sin\frac{\theta_1}{2} + a_3\cos\frac{\theta_1}{2}\right)\cos\frac{\theta_2}{2} - \left(a_2\sin\frac{\theta_1}{2} + a_4\cos\frac{\theta_1}{2}\right)\sin\frac{\theta_2}{2} \\ \left(a_1\sin\frac{\theta_1}{2} + a_3\cos\frac{\theta_1}{2}\right)\sin\frac{\theta_2}{2} + \left(a_2\sin\frac{\theta_1}{2} + a_4\cos\frac{\theta_1}{2}\right)\cos\frac{\theta_2}{2} \end{pmatrix}. \tag{5.7}$$

In the final step, *Pauli-Z* (σ_z) measurement is performed on the qubits to obtain the quantum measurement-based features. Let the state of a two qubit system before mea-

surement be $|\mathbf{S}\rangle$, where

$$|\mathbf{S}\rangle = \frac{1}{\sqrt{\alpha^2 + \beta^2 + \gamma^2 + \delta^2}} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix}. \tag{5.8}$$

After *Pauli-Z* (σ_z) measurement where

$$(\sigma_z) = Z \otimes Z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

. The state $|\mathbf{S}\rangle$ of qubit collapses and the value obtained is:

$$Tr(|\mathbf{S}\rangle\langle\mathbf{S}|\sigma_z) = \frac{(|\alpha|^2 - |\beta|^2 - |\gamma|^2 + |\delta|^2)}{(\alpha^2 + \beta^2 + \gamma^2 + \delta^2)} \tag{5.9}$$

where Tr denotes the trace operation. *Pauli-Z* measurement on a qubit always gives a value between $[-1, 1]$. Hence, for the four values of input pixel data: *Row 1* $= \{a_1, a_2, a_3, a_4\}$, we obtain two values for quantum measurement on two qubits using the *Pauli-Z* measurement. The same quantum circuit is used to obtain the quantum measurement-based features for the remaining rows of the grayscale. Thus, for the input image of 4 rows $\times$ 4 values $= 16$ pixel values, a total of 4 rows $\times$ 2 measurement values $= 8$ values are obtained as quantum representations of the image. All the 8 values are stored as comma-separated values along with *Image Id* and *Class label*. The quantum representations of the images are used to train the fully connected deep neural network in the *Level-3*. The details of the quantum simulations performed on the images of the benchmark datasets to obtain quantum representations are given in Section 5.4.

***Level-3*: Classical Deep Learning Model**

The *Level-3* of the hybrid architecture is to develop a classical deep learning model using the quantum representations of the images obtained in the *Level-2*. The classical deep learning model is designed using a fully connected deep neural network with three types of layers:

1. *Input Layer*: A single layer with a number of nodes equal to the number of features

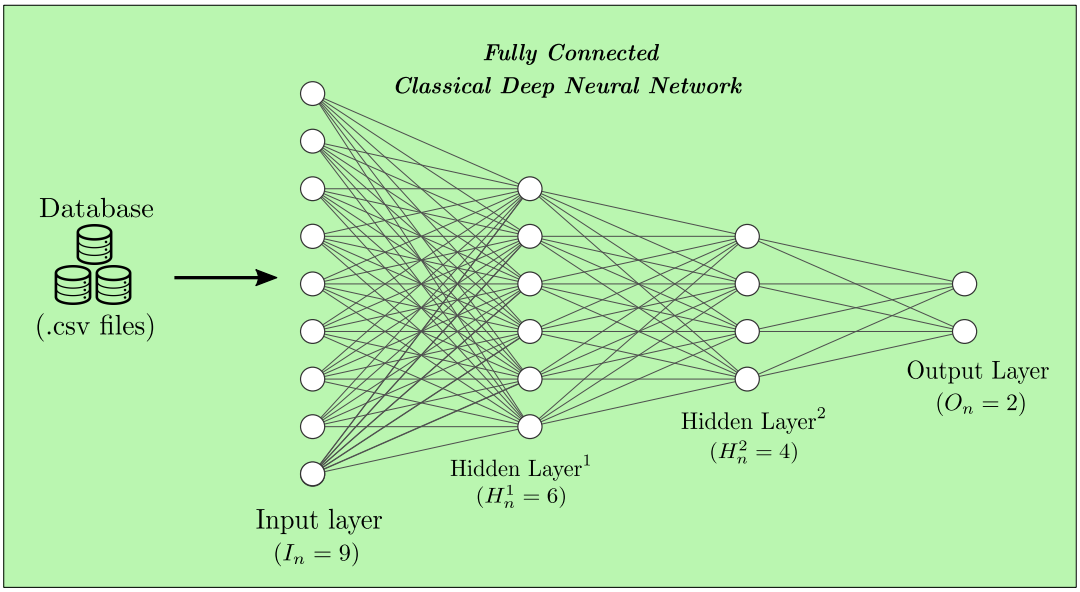


Figure 5.6: Illustration for the architecture of a classical deep learning model.

given as input.

2. *Hidden Layers*: Multiple layers with a variable number of nodes.

3. *Output Layer*: A single layer with a number of nodes equal to the number of class labels in a dataset.

Fig. 5.6 shows an example of a classical deep learning model for classification. The model consists of an *input layer* with 9 nodes, *Hidden Layer*^{1} with 6 nodes, *Hidden Layer*^{2} with 4 nodes, and an *Output Layer* with 2 nodes. The total trainable parameters of the model can be calculated as $[9 \times 6 + 6 \times 4 + 4 \times 2] = 86$. The model is trained for multiple steps to converge the training loss and improve overall accuracy.

Table 5.1: Details of number of nodes used in each layer of fully connected deep neural network for different datasets

Dataset	Input Layer (I_n)	Hidden Layers (H_n)	Output Layer (O_n)
UC Merced Land-Use	I_n = 2048	H_n^1 = 880 H_n^2 = 64	O_n = 21
AID	I_n = 4608	H_n^1 = 256 H_n^2 = 64	O_n = 30
NWPU-RESISC45	I_n = 3548	H_n^1 = 512 H_n^2 = 64	O_n = 45

In this work, we used three different models for the classification of the three benchmark datasets. Table 5.1 contains the details of the number of nodes used in each layer for the three different datasets. Table 5.2 shows the total number of quantum measurement-based features extracted per image along with the number of qubits used for each of the datasets. The implementation details of the designed classical deep learning model are given in Section 5.4. Thus, a hybrid quantum-classical model that uses quantum computation and classical computation is developed for satellite image scene classification.

Table 5.2: Details of number of quantum measurement-based features extracted along with number of qubits used for each of the datasets

Dataset	Images	Size after *Level 1*	Qubits used	Measurement-based features per image
UC Merced Land-Use	2100	256×256	8	2048
AID	10,000	512×512	9	4608
NWPU-RESISC45	31,500	256×256	7	3584

5.2 Hybrid Quantum-Classical Convolutional Neural Network

In recent years, convolutional neural network models gained prominence in sensor signal processing ranging from inertial sensors to image sensors. Convolutional neural networks (CNNs) are used in many real-time high-impact applications such as hand tremor detection, fingerprint detection, and target classification [115, 116, 117]. Target or scene classification is a significant task that aims to process the high-resolution images and find an object or scene in the images. There is a potential application of satellite image scene classification in various fields such as detection of natural calamities, vegetation mapping, military applications, urban planning, and environment monitoring [46].

Deep learning proved to be efficient for different satellite image processing and scene classification tasks [44]. Feature extraction with CNNs uses multiple layers with trainable parameters for remote sensing image scene classification. However, training such methods requires large datasets to attain good classification accuracy. Also, CNN models use a large number of trainable parameters to learn the features that lead to complexity in training. Quantum computing techniques [118] can be used to design and develop methods that improve the performance of machine learning models for applications in remote sensing such as scene classification [119, 120].

Sensor signal processing with CNNs is an area of concern as rapidly growing data (col-

lected from sensor signals) affects the performance of CNNs for classification [121]. For example, satellite image data collected for analysis consists of various labels with fewer samples for each label. Data augmentation can help deal with such datasets. Images are modified using augmentation techniques such as rotation and noise-addition to increase the size of datasets. Also, data augmentation is proved to be effective in avoiding overfitting of the models and improving accuracy [122]. However, no one augmentation technique fits all types of data, and thus there is a need for more techniques. Another major concern of CNNs is the training process, as the models require many parameters during the training phase.

In this work, we propose image processing techniques using both hybrid quantum and classical computations to enhance the training process of CNNs and also improve classification accuracy. The significant contributions of the work are as follows.

1. Designed a data augmentation technique using a quantum circuit and classical computation.

2. Experimented with the quantum circuit on *ibm santiago* quantum computer provided by *IBM QX* [84].

3. Designed a hybrid quantum-classical CNN model to improve scene classification accuracy with reduced training parameters.

4. Detailed analysis of results in comparison with the other models.

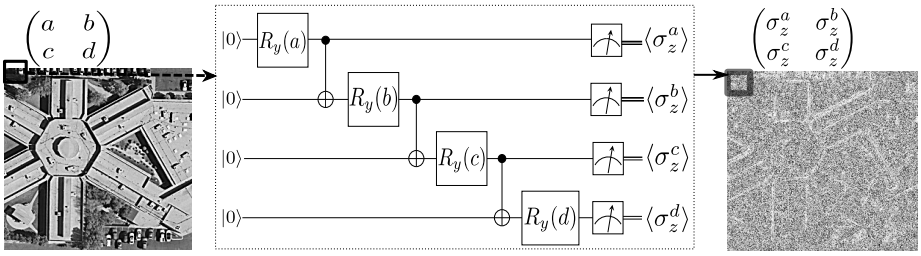


Figure 5.7: Quantum circuit for data augmentation.

5.2.1 Quantum Circuit for Data Augmentation

In the NISQ era, quantum processing units (QPUs) at present consist of a limited number of qubits for computation [26, 84] and hence, we designed a quantum circuit with four qubits for the data augmentation purpose. As shown in Fig. 5.7, R_y gates are used with

entanglement between the qubits using *CNOT* gates. However, different quantum operations can also be used in the circuit based on the requirement. In our work, we used the UC Merced Land-Use dataset [112] with 21 classes with 100 images for each class.

The proposed data augmentation technique is used to enhance the dataset size as follows. All images from the size 256×256 are converted to grayscale as processing RGB images is difficult with only limited qubits available. Each 256×256 image is divided into a 2×2 sub-block, and four values are given as an input to the rotational gates of the quantum circuit with four qubits.

As shown in Fig. 5.7, each pixel value from the 2×2 sub-block is encoded into the rotational gate ($R_y(\theta)$) of the quantum circuit. The initial state of all qubits in the quantum circuit is $|0\rangle$. The rotational gate R_y rotates the qubit in the *Bloch sphere* at an an angle of θ, where $\theta \in$ pixel values. The entangled state of all the qubits is given as

$$\begin{aligned} |\psi\rangle = \cos(a/2)\cos(b/2)\cos(c/2)\cos(d/2)\,|0000\rangle + \\ \cos(a/2)\cos(b/2)\cos(c/2)\sin(d/2)\,|0001\rangle + \\ \cdots - \sin(a/2)\cos(b/2)\cos(c/2)\sin(d/2)\,|1110\rangle \\ + \sin(a/2)\cos(b/2)\cos(c/2)\cos(d/2)\,|1111\rangle\,. \end{aligned} \tag{5.10}$$

Finally, *Pauli-Z* (σ_z) measurement is performed on the qubits to obtain a value between [-1, 1]. The quantum measurement values are scaled back to [0, 255] to obtain a grayscale representation. Thus, all the sub-blocks of the 256x256 image are processed using a quantum circuit to obtain a quantum image representation. In the quantum circuit, measurement of one qubit influences the measurement of the entangled qubit, and hence values of q_{image} are randomized. Thus, different outputs can be obtained for the same image if multiple measurements are performed.

Interpolation is used to blend (b_{image}) the obtained quantum image representation (q_{image}) with the classical image (c_{image}). The interpolation equation is given as

$$b_{image} = c_{image} * (1.0 - \alpha) + q_{image} * \alpha \tag{5.11}$$

where α is a constant value between 0 and 1 that decides the overlap between q_{image} and c_{image}. In our work, we used α as 0.5 to create a blended image with 50% of each of the images. The interpolated images can be used to train deep learning models.

Qubits are fundamentally different from classical bits as they are prone to noise (decoherence) from electromagnetic fields and material defects [26, 22]. Hence, transforming

the images using a quantum circuit adds inherent quantum noise to q_{image}. Thus, interpolating a quantum image with a classical one adds inherent noise that cannot be exactly generated using a classical computer. In the training phase, adding noise to the input data reduces generalization error, and therefore, the models can learn better [123, 122]. The proposed quantum circuit can also be implemented on real quantum hardware as given in Section 5.5.

5.2.2 Hybrid quantum-classical CNN Architecture

This section describes the details of the hybrid quantum-classical model proposed to reduce the parameters used for training CNNs.

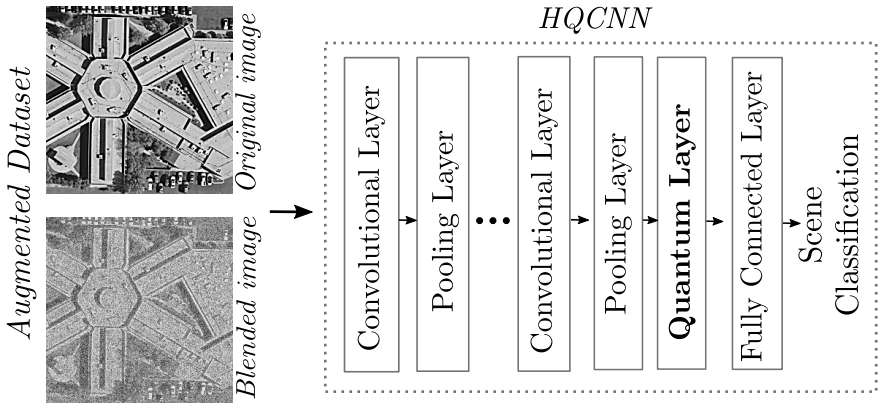


Figure 5.8: Illustration of computational layers in HQCNN.

A convolutional neural network (Vanilla CNN) is a deep learning model with an input layer, convolutional layers, pooling layers, and finally, a fully connected layer. The problem with CNN models is that they require vast amounts of training samples and extensive computational facilities as many parameters are to be trained.

Recently, Quantum neural networks (QNN) [30] are derived from the concept of variational quantum circuits and machine learning. Deep learning algorithms are designed as QNNs using parameterized quantum circuits. Classical data is encoded into qubits with a *quantum loader* using efficient encoding schemes [118]. Qubit encoding schemes such as amplitude encoding [24] can encode 2^n data points into n qubits as a superposition of different quantum states. Thus, huge amounts of classical data can be handled efficiently using as *quantum loader*.

Each layer in the quantum neural network consists of *parameterized quantum gates* with combinations of the rotational gate, phase gates, and entanglement [57]. The parameters of

the quantum gates are optimized using classical optimization techniques. The parameters are optimized until the quantum *measurement* gives the desired output.

In our work, we combine the vanilla CNN (VCNN) and QNN together to obtain a hybrid quantum-classical CNN model (HQCNN) for image scene classification. As shown in Fig. 5.8, the model consists of a combination of classical CNN layers and quantum layers in between the CNN layers. Each quantum layer is a quantum circuit consisting of a quantum loader and parameterized quantum gates, followed by measurement operations.

The output values from the pooling layer can be encoded into qubits using encoding schemes such as amplitude encoding. As data encoding into qubits provides exponential scaling, there results a huge reduction in the trainable parameters of the model. The quantum loader is followed by parameterized quantum gates that consist of rotational gates with rotational angles as trainable parameters. Finally, a measurement operation is performed on the qubits of the quantum circuit. The output values of the quantum layer are given as input to the fully connected layer.

Convolutional layers and pooling layers of CNNs effectively extract features from images. QNNs are efficient in handling huge amounts of data and provide exponential scaling. Thus, we combine the ability CNNs and QNNs to process the information for the scene classification task. We consider that high-performance computing environments consist of CPUs, GPUs, and QPUs to process information in the future.

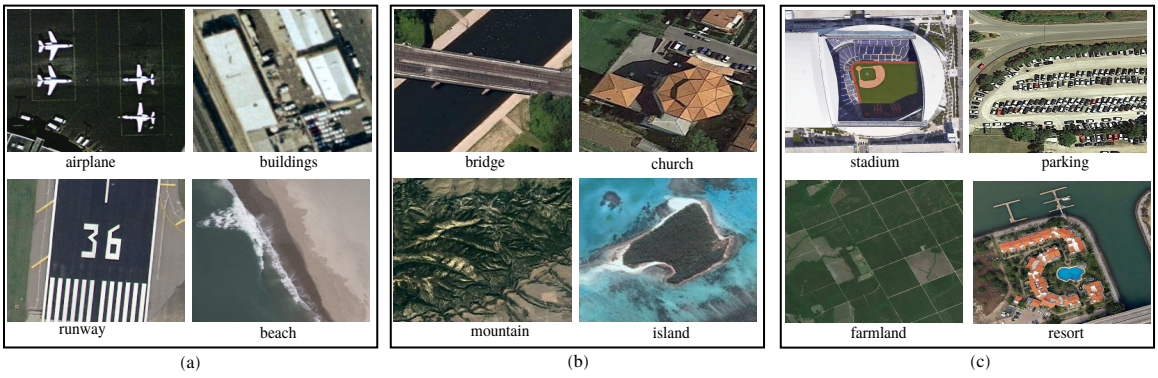


Figure 5.9: Examples of satellite images from: (a) UC Merced Land-Use dataset, (b) NWPU-RESISC45 dataset, (c) AID.

5.3 Experiments on Spatial Data for Classification

In this section, we present the details of the three datasets used for demonstrating the effectiveness of the proposed architecture. The three benchmark datasets used for remote sens-

ing scene classification are: UC-Merced Land-Use [112], Aerial Image Dataset (AID) [113], and NWPU-RESISC45 [114]. Fig. 5.9 shows the example images from the datasets, and the details of the datasets are given below.

Land-Use Dataset by UC Merced (UC Merced Land-Use): The Land-Use dataset by the University of California Merced (UC Merced) consists of 2100 aerial scene RGB color images with a resolution of 256×256 pixels. The images are divided into 21 land-use classes with 100 images per class label.

Aerial Image Dataset (AID): AID consists of 10,000 aerial scene RGB color images with a resolution of 600×600 pixels and the images are divided into 30 scene types with 200-400 images per class label.

Northwestern Polytechnical University - Remote Sensing Image Scene Classification (NWPU-RESISC45): NWPU-RESISC45 is a complex dataset with 31,500 images with a resolution of 256×256 for 45 scene classes for Remote Sensing Image Scene Classification (RESISC) with 700 images per class label created by Northwestern Polytechnical University (NWPU).

All the images from the three datasets are converted to grayscale and processed row-wise using a quantum circuit. The detailed methodology followed to implement the hybrid architectures is given in the following sections along with the the details of the quantum simulations performed on the three benchmark datasets.

5.4 Performance Analysis of HybridQC for Satellite Image Scene Classification

The quantum circuit described in Section 5.1.1, to obtain quantum representations of images, is implemented using `PennyLane` [89]. Qubits are simulated using the `default.qubit` simulator provided by `PennyLane`. The unitary operations (U_{amp}) for amplitude encoding of the input values is implemented using `AmplitudeEmbedding` method provided in `pennylane.templates.embeddings`.

The entanglement between the qubits is implemented using `pennylane.CNOT` operation. `pennylane.RY` operation is used to perform the rotations on qubits after the entanglement. Finally, using `pennylane.expval(pennylane.PauliZ)` operation, the qubits are measured with *Pauli-Z* measurement.

The details of the strategy followed on each dataset to obtain quantum representations of the images are given in the following sections.

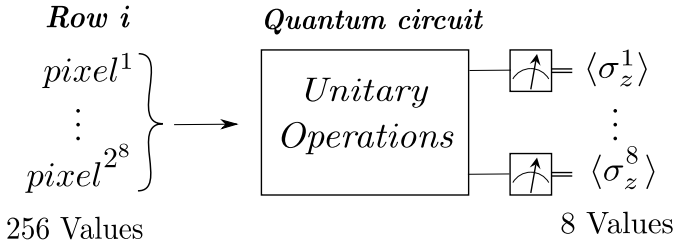


Figure 5.10: Illustration of quantum measurement-based feature extraction on UC Merced Land-Use dataset.

Quantum Simulations on UC Merced Land-Use Dataset

All the UC Merced Land-Use dataset images are converted to grayscale, and thus each row of the pixel matrix for an image consists of 256 values. As shown in Fig. 5.10, the 256 values (2^8) from each row of 256×256 matrix are given as input to the quantum circuit with 8 qubits. All the 256 values are encoded as amplitudes of 256 quantum states in superposition.

An entanglement operation is performed in the next step between all the pairs of 8 qubits using the CNOT gate. Then, rotational gate $R_y(\theta_i)$, where $i = \{1, 2, \ldots, 8\}$, is implemented on the 8 qubits. The input data of 256 values are divided into eight sets of 32 values each. Then, θ_i values are calculated as the average of 32 values in each set. Hence, we obtain eight values as input to $R_y(\theta_i)$ gate, and each gate is applied on the eight qubits of the circuit individually. Finally, *Pauli-Z* measurement ($\langle\sigma_z\rangle$) is performed on qubits to obtain quantum representations of the image. Hence, for each 256×256 image, 2048 (256 rows $\times$ 8 output values) quantum measurement-based features are extracted using eight qubits.

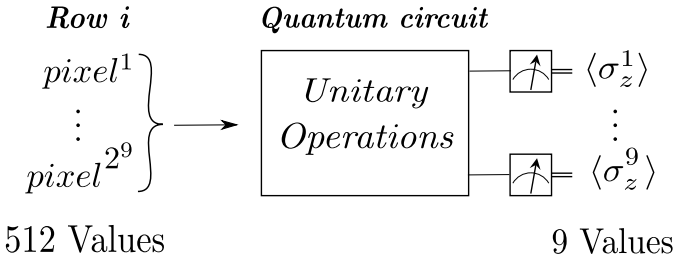


Figure 5.11: Illustration of quantum measurement-based feature extraction on AID.

Quantum Simulations on AID dataset

Aerial Image Science Dataset (AID) images are converted to grayscale, and the resultant size of each image is 600×600. As 512 is the nearest power of 2 for 600, all the images are resized to 512×512. As shown in Fig. 5.11, the 512 values from each row of 600×600 matrix are given as input to the quantum circuit with 9 qubits. The input data of 512 values are encoded as amplitudes of 512 quantum states in superposition.

An entanglement operation is performed in the next step between all the pairs of 9 qubits using the CNOT gate. Then, rotational gate $R_y(\theta_i)$, where $i = \{1, 2, \ldots, 9\}$, is implemented on the 9 qubits. The input data of 512 values are reduced to 504 values by removing the last eight values. Then, the 504 values are divided into nine sets of 56 values each. Then, θ_i values are calculated as the average of 56 values in each set. Hence, we obtain nine values as input to $R_y(\theta_i)$ gate, and each gate is applied on the nine qubits of the circuit individually. Finally, *Pauli-Z* measurement ($\langle\sigma_z\rangle$) is performed on qubits to obtain quantum representations of the image. Hence, for each 512×512 image, 4608 (512 rows $\times$ 9 output values) quantum measurement-based features are extracted using nine qubits.

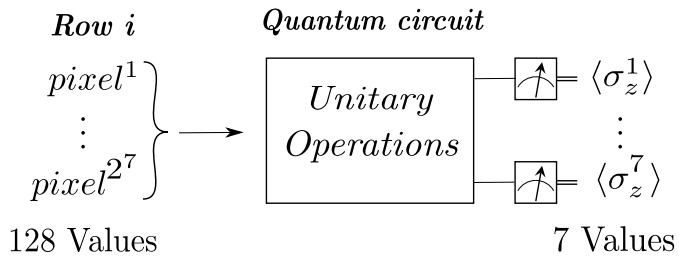


Figure 5.12: Illustration of quantum measurement-based feature extraction on NWPU-RESISC45 dataset.

Quantum Simulations on NWPU-RESISC45 dataset

The images of the NWPU-RESISC45 dataset are of size 256×256, and each image is converted to grayscale. Even though the image size is similar to the UC Merced Land-Use dataset, NWPU-RESISC45 is a large dataset with 45 scene class labels. Hence, more features are to be collected to train the deep learning model for classifying all the 45 scene classes. Collecting fewer quantum measurement-based features leads to underfitting of the deep learning model used for NWPU-RESISC45 dataset classification. Hence, a different

approach is followed to collect more quantum measurement-based features for quantum representations of images.

Each row of the image matrix with 256 values is further divided into two halves, each with 128 values. Then, a quantum circuit with seven qubits is used to encode the first 128 values, as shown in Fig. 5.12. The input data of 128 values are encoded as amplitudes of 128 quantum states in superposition.

An entanglement operation is performed in the next step between all the pairs of 7 qubits using the CNOT gate. Then, rotational gate $R_y(\theta_i)$, where $i = \{1, 2, \ldots, 7\}$, is implemented on the 7 qubits. The input data of 128 values are reduced to 126 values by removing the last two values. Then, the 126 values are divided into seven sets of 18 values each. Then, θ_i values are calculated as the average of 18 values in each set. Hence, we obtain seven values as input to $R_y(\theta_i)$ gate, and each gate is applied on the seven qubits of the circuit individually. Finally, *Pauli-Z* measurement ($\langle\sigma_z\rangle$) is performed on qubits to obtain quantum representations of the image. Thus, we obtain seven output values for the 128 input values. The same procedure is repeated for the next 128 values of the second half of the first row (256 values). Hence, for each 256×256 image, 3584 (256 rows (7 + 7) output values) quantum measurement-based features are extracted using seven qubits.

5.4.1 Observations and Discussion

In this section, we describe in detail the experiments performed on *ibm_santiago* quantum computer to obtain the quantum representations. Also, the details of performance evaluation of the proposed hybrid quantum-classical model (HybridQC) in comparison with the state-of-the-art models are given along with results and discussion.

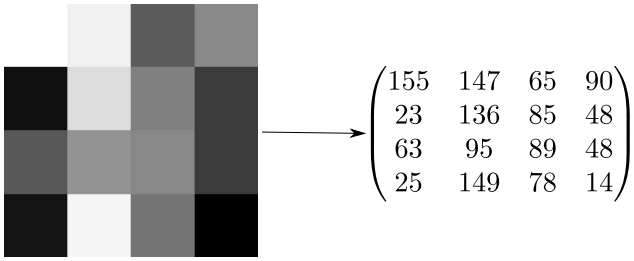


Figure 5.13: Grayscale values of the image used to experiment on *ibm_santiago*.

Experiment on *ibm_santiago* quantum computer

At present, a minimal number of qubits are available in NISQs for computation. As HybridQC requires less number of qubits for processing in the *Level-2*, the proposed quantum processing is suitable to implement on NISQs. The details of the experiment performed on NISQs to obtain quantum representations for the image in Figure 5.13 are given in the following.

Table 5.3: Configuration details of *ibm_santiago* quantum computer

Type	Value
Total Qubits	5
Processor	Falcon r4L
Basic gates	*CX (CNOT), ID, RZ, SX, X*
Avg. CNOT Error	6.192e-3
Avg. Readout Error	1.214e-2

The real quantum hardware *ibm_santiago*, provided online by *IBM Q Experience* [8], is used to perform the experiment. Table 5.3 shows the configuration details of *ibm_santiago* quantum computer.

`pennylane-qiskit` plugin provided by *PennyLane* is used to write python code to perform the experiments. As the number of values in each row is four, as shown in Figure 5.13, we use only two qubits from *ibm_santiago* quantum computer for the experiment. The following is the sample code to obtain quantum representations from the first row of the image matrix.

```
#PennyLane code for implementation on ibm_santiago#
1 import numpy as np
2 import pennylane as qml
3 from pennylane.templates import AmplitudeEmbedding

4 dev = qml.device('qiskit.ibmq', wires=2,
backend='ibmq_santiago',ibmqx_token="XXXX")

5 pixels = [155, 147, 65, 90]

6 @qml.qnode(dev)
7 def circuit(vals=None):
8   qml.templates.embeddings.AmplitudeEmbedding
```

```
(vals,wires=[0, 1], normalize = True)
9    qml.RY(np.average([vals[0], vals[1]]), wires=0)
10   qml.RY(np.average([vals[1], vals[2]]), wires=1)
11   return [qml.expval(qml.PauliZ(0)),
                qml.expval(qml.PauliZ(1))]

12 quantum_representations = circuit(pixels)
```

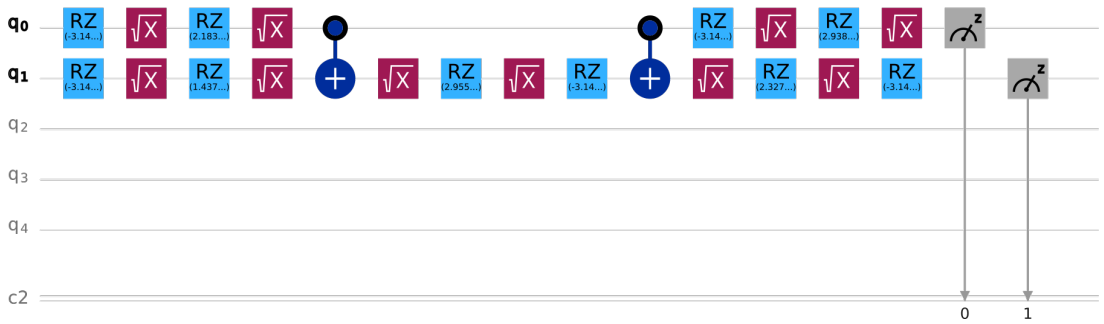


Figure 5.14: Quantum circuit for the experiment on *ibm_santiago* with the basic gates.

The description of the sample code is as follows.

- Line 1 to Line 3 is used to import the necessary packages to perform the experiments.
- In Line 4, the quantum computer to perform the quantum operations is declared. For the experiment, *ibm_santiago* quantum computer is selected using `backend` and accessed using the API token generated on *IBM Q Experience*. The token obtained from *IBM Q Experience* is given as an input to `ibmqx_token`.
- The variable `pixels` in Line 5 is declared with the pixel values of the first row of the image in Figure 5.13.
- `@qml.qnode(dev)` in Line 6 is a decorator used to indicate that the method `circuit` in Line 7 is to be executed on the quantum device declared in `dev`.
- `AmplitudeEmbedding` method declared in Line 8 is used to encode the input values in `pixels` into qubits.
- The operations in Line 9 and Line 10 are used to input an average of pixel values into R_y gate as described in Section 5.1.1.

- Finally, in Line 11, `PauliZ` measurement is performed on the qubits.
- `quantum_representations` in Line 12 contains the output values of the measurement values from the `circuit` method.

The quantum operations from the method `circuit` in the above code are performed on *ibm_santiago* quantum computer using the basic gates given in Table. 5.3. The corresponding quantum circuit on *ibm_santiago* is given in Figure 5.14. The *Pauli-Z* measurement values from the quantum circuit stored in the `quantum_representations` variable are considered as the quantum representations of the input `pixels`. In a similar way, quantum representations of different images can be obtained by using NISQs for processing at *Level-2* in the proposed architecture. The obtained quantum representations are used to train the classical deep learning models.

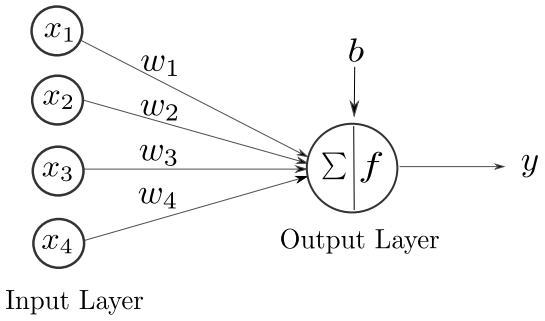


Figure 5.15: Illustration of parameters of a neural network.

HybridQC Comparison

In this work, we used the Keras framework [94] to implement the classical deep learning models using quantum representations of images for training. 80% of the data from the .csv files is used for training and 20% is used for testing the models. The hyperparameters such as nodes and the number of hidden layers are adjusted during the training process on the three datasets to produce the best results. The stochastic gradient descent (SGD) optimizer with Nesterov momentum is used to optimize the training weights with momentum = 0.9 and learning rate = 0.001. The rectified linear unit (ReLU) activation function is used at the hidden layers, and activation at the output layer is performed using the softmax function. Data augmentation and Dropout regularizations are used to avoid overfitting of the models. The models are trained until the categorical cross-entropy loss is converged. All the quantum simulations to extract measurement-based quantum features and exper-

iments for the proposed approach are performed on a computing node with CentOS 7.5 operating system, one 2.6 GHz 28-core Intel Xenon E5-2690 V4 CPU, and 128GB memory.

The trainable parameters of the classical deep learning models are the edge weights of the deep neural network nodes, along with a bias added at each layer. Figure 5.15 represents a sample network to describe the parameters. The sample network consists of only one input layer and an output layer. The input layer consists of four nodes, and the output layer is with one node. The values $x_1, x_2, x_3,$ and x_4 represents the input. The output value y is calculated as

$$y = f(w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b) \tag{5.12}$$

where f is an activation function and b is the bias. The trainable parameters are the weights $w_1, w_2, w_3, w_4,$ and b. The neural network is trained with the data until the weights are optimized such that desired result y is obtained.

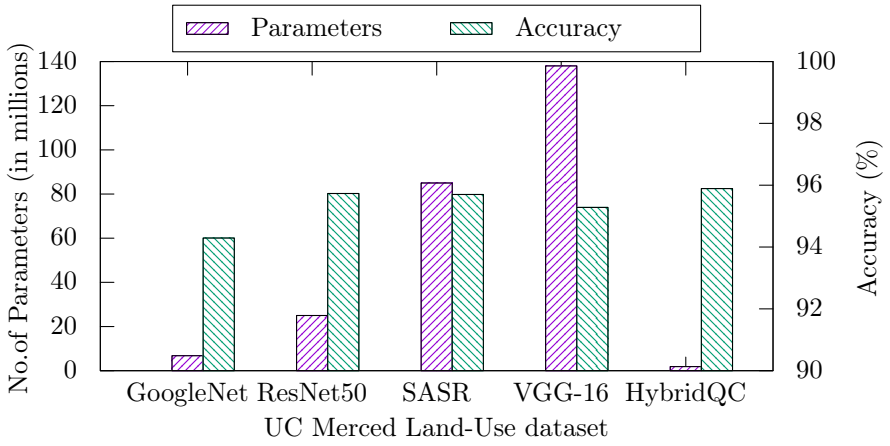


Figure 5.16: Comparison of parameters and accuracy of different models with the proposed HybridQC model on UC Merced Land-Use dataset.

Table 5.4: Comparison of overall accuracy (in %) and total trainable prameters (in millions (M)) for the proposed hybrid QC apporach with the other state-of-the-art models

Method	*UC Merced Land-Use*	Trainable parameters	*AID*	Trainable parameters	*NWPU RESISC45*	Trainable parameters
GoogLeNet	94.29%	∼6.79M	86.34%	∼6.79M	78.74%	∼6.79M
ResNet50	95.73%	∼25M	91.83%	∼25M	79.42%	∼25M
SARS	95.70%	∼85M	87.74%	∼85M	78.92%	∼85M
VGG-16	95.28%	∼138M	89.57%	∼138M	79.58%	∼138M
HybridQC	**95.89%**	**∼1.85M**	**86.13%**	**∼1.92M**	**79.32%**	**∼1.87M**

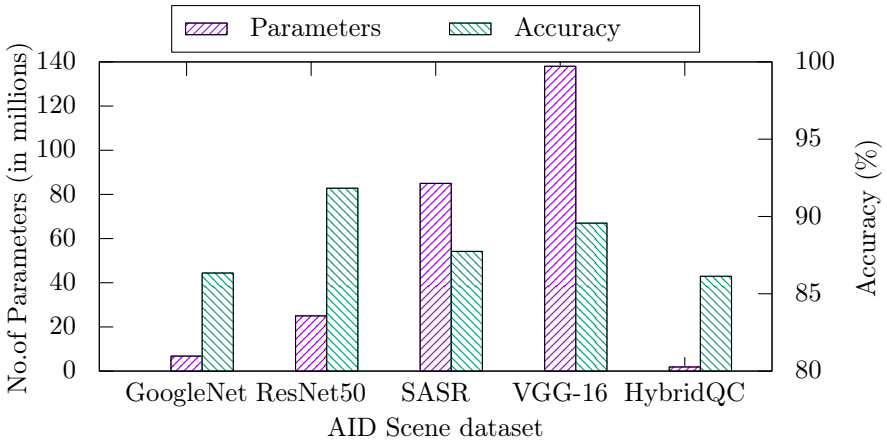


Figure 5.17: Comparison of parameters and accuracy of different models with the proposed HybridQC model on AID Scene dataset.

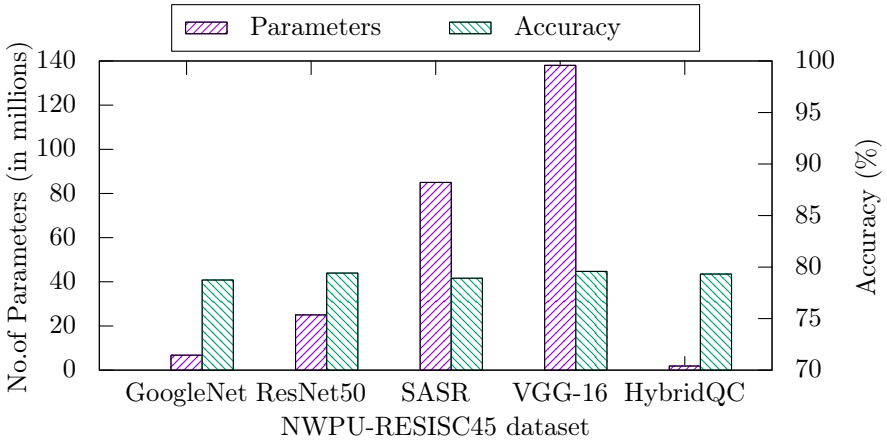


Figure 5.18: Comparison of parameters and accuracy of different models with the proposed HybridQC model on NWPU-RESISC45.

The proposed hybrid quantum-classical model (HybridQC) is compared with the state-of-the-art deep learning models [124, 125] for image classification. We compare HybridQC with VGGNet [126], GoogleNet [127], ResNet [128], and semantic-aware scene recognition model (SA) [129] to evaluate the performance and effectiveness. For NWPU-RESISC45 dataset and AID, 20% of the images from the original dataset are randomly selected for the training purpose on state-of-the-art models.

Fig. 5.16 shows that the proposed HybridQC model outperforms all other models on the UC Merced Land-Use dataset with an overall accuracy of 95.89%. Also, on the other datasets, HybridQC competes well with the state-of-the-art models as shown in Fig. 5.17 and Fig. 5.18. The results also show that the proposed approach uses very less number of

trainable parameters for the deep learning models.

The overall accuracy comparison of the proposed model with the state-of-the-art models is given in Table 5.4 along with a total number of trainable parameters for each model. As the number of trainable parameters reduced with the proposed approach, the computational requirement to implement the models is less compared to the state-of-the-art-models. Hence, the difficulty of training the models reduces with the decrease in the number of trainable parameters. The proposed architecture is also considered to be computationally time-efficient, as the model building and experimentation using the proposed approach can result in drastic time reduction. Thus, quantum enhancement can be performed to the deep neural network architectures for image scene classification.

Quantum computing is gaining prominence due to a quantum computer's ability to handle huge amounts of data with exponential speed. In this thesis, we also propose a hybrid quantum-classical computing architecture for satellite image scene classification. Quantum representations of images are extracted using a quantum circuit, and the extracted representations are used to train the deep learning models. Quantum image representations can be considered unique as the images are encoded into qubits to represent the data in a high dimensional Hilbert space and processed using quantum gates. Our results show that existing deep learning architectures for image scene classification can be enhanced using the proposed hybrid quantum-classical architecture. The performance of the proposed hybrid quantum-classical approach is evaluated using three benchmark satellite image datasets. The experimental results show that the proposed model achieves good overall accuracy compared to the state-of-the-art models for scene classification. Also, the proposed approach uses fewer trainable parameters than the state-of-the-art models.

5.5 Performance Analysis of HQCNN for Satellite Image Scene Classification

This section presents the details of the experiments performed to study the impact of data augmentation on different architectures. The experiments and simulations are performed on multiple synthesized datasets obtained from the UC Merced land-use dataset. The experiments and simulations to evaluate the performance of the proposed hybrid quantum-classical model is also given in Section 5.5.2.

5.5.1 Data preprocessing

The UC Merced land-use dataset is synthesized into multiple datasets with 3, 5, 10, and 21 scene classes, respectively. All the input images to the quantum circuit given in Section 5.2.1 are converted to grayscale. Data augmentation is performed using the proposed QDA to enlarge the volume of the dataset. The performance comparison of the proposed quantum data augmentation (QDA) is performed with the two most widely used classical image augmentation techniques on all the synthesized datasets. The first classical data augmentation technique (CDA_1) is performed using image rotation with rotation angle $90°$. The second technique (CDA_2) is performed using salt and pepper noise where the pixels are randomly replaced with 1 or 0.

5.5.2 Performance evaluation

This section presents the details of the performance evaluation of the proposed HQCNN in comparison with VCNN and QNN on CDA_1, CDA_2, and QDA. CNN models in our experiments are implemented using the TENSORFLOW library in Python. QNN and HQCNN are implemented using PENNYLANE [89], a quantum machine learning library in Python. PENNYLANE provides a *QNN module* that can be used to implement both QNNs and hybrid models. We also implemented the quantum circuit for data augmentation on *IBMQX* using *pennylane-qiskit* plugin. The following is the sample code to execute quantum operations on *ibm_santiago*.

```
1 import pennylane as qml
2 dev = qml.device('qiskit.ibmq', wires=4,
backend='ibmq_santiago',ibmqx_token="XXXX")
```

Line 1 is used to import the pennylane package, and Line 2 is to select the device on the IBM QX via the pennylane-qiskit plugin. The token can be generated by creating a user account on the IBM QX. The quantum operations are executed on the selected device.

Table 5.5: Comparison of training and validation accuracy of different models.

Scenes	VCNN						**HQCNN**						QNN					
	CDA_1		CDA_2		QDA		CDA_1		CDA_2		**QDA**		CDA_1		CDA_2		QDA	
	Train	Val	Train	Val	Train	Val	Train	Val	Train	Val	Train	Val	Train	Val	Train	Val	Train	Val
3 classes	98.43	87.50	99.04	92.32	99.31	97.86	99.68	86.93	93.32	86.66	**96.03**	**88.17**	96.22	73.41	94.77	75.21	97.28	75.55
5 classes	97.82	81.45	97.24	90.14	98.12	95.06	98.24	82.29	93.96	85.35	**98.83**	**86.98**	86.66	63.02	86.25	62.15	89.02	65.14
10 classes	96.21	73.25	97.57	89.93	97.76	92.58	97.04	78.53	91.22	81.79	**98.67**	**85.65**	79.11	58.89	80.44	55.82	81.69	62.78
21 classes	89.97	70.64	96.73	81.71	95.56	86.29	88.19	78.52	89.54	80.11	**97.36**	**86.55**	72.68	52.87	76.58	53.42	80.27	55.98

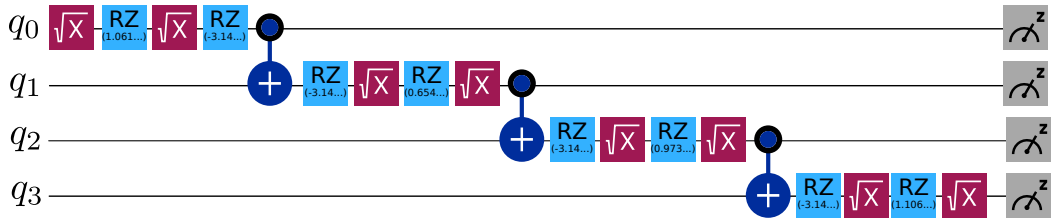


Figure 5.19: Quantum circuit for data augmentation on *ibm_santiago*.

Table 5.6: Experimental details of different models for 3 class labels

	VCNN	**HQCNN**	QNN
Processor	CPU+GPU	CPU + GPU + QPU	CPU + GPU + QPU
Software	TensorFlow	PennyLane	PennyLane
Type of parameters	Parameters in CNN layers	Parameters in CNN layers and Rotation angles in Quantum layers	Rotation angles in Quantum layers
Total parameters	≈800K	≈**36K**	≈560K

The standard ratio for the train-test split of the datasets, 80:20, is used in our study. In turn, the training set is divided again into a train-validation group of 80:20. Training is performed to converge the loss function (categorical cross-entropy loss).

Fig. 5.19 shows the quantum circuit from IBMQX implemented using basic gates of *ibm_santiago* for the pixel values [155, 147, 65, 90]. Table 5.5 provides the comparison of training accuracy and validation accuracy during the training process for all the models. The results show a huge difference between the training and validation accuracy using CDA_1 and CDA_2 in comparison with QDA. For example, the differences in the training accuracy and validation accuracy of VCNN for 21 classes using CDA_1 and CDA_2 are 19.33% and 15.02%, respectively. In comparison, the difference using the proposed DA is 9.27%. Also, the training accuracy using the proposed DA is more (95.56%) when compared to CDA_1 and CDA_2. Thus, the proposed data augmentation technique allows the models for better generalization.

Also, the test results from Fig. 5.20 show that the proposed HQCNN model outperforms all the other models in terms of overall accuracy. HQCNN can classify all the 21 classes in the UC Merced Land-Use dataset with an accuracy of 85.28%. Also, the results show that the proposed data augmentation technique gives better accuracy than the traditional and widely used classical image data augmentation techniques.

Quantum computation techniques can be used to enhance the existing machine learning models. The proposed data augmentation is a combination of quantum and classi-

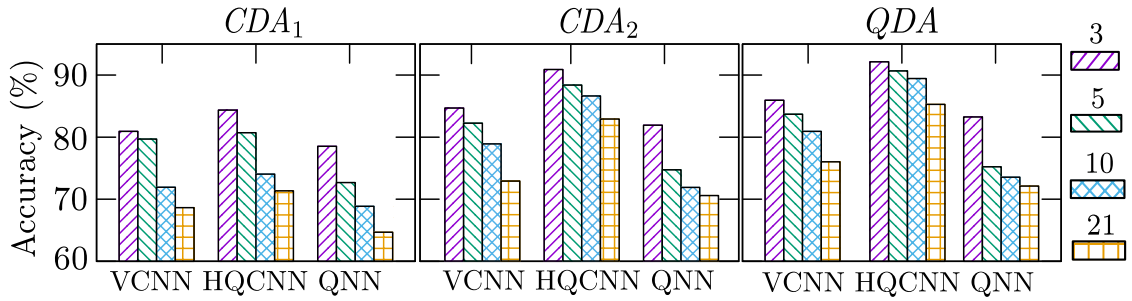


Figure 5.20: Test accuracy comparison of the models.

cal operations. Transformation of images using quantum circuit operations adds inherent noise to the blended images. Hence, the performance of the models increased when trained using the proposed augmentation technique.

As HQCNN processes data using both CNN layers and quantum layers, the model outperformed the other models in terms of parameters used for training the models, as shown in Table 5.6. The hybrid models can combine the efficiency of classical computation and quantum computation. The test accuracy of HQCNN using the proposed QDA on datasets with class labels 3, 5, 10, and 21 is 92.13%, 90.67%, 89.45%, and 85.28%, respectively. Hence the results show that the proposed quantum computation techniques enhanced the performance of the models.

The rapid development of sensor technology and data generation requires new data handling techniques. Deep learning models such as CNNs are efficient for the purpose and also require large datasets with a good number of samples for each class label to avoid overfitting. In this chapter, we proposed a data augmentation technique using quantum computation to enhance the size of datasets. Data augmentation is performed by processing classical images using a quantum circuit and blending the quantum output with the original image. We also proposed a new hybrid model that combines the efficiency of CNNs with the quantum computer's ability to process data with exponential scaling. The proposed model is suitable for future high-performance computing environments with classical and quantum processors. Our proposed data augmentation technique performed well in terms of accuracy on all the models. Also, the proposed hybrid quantum-classical CNN model outperforms all the other models.

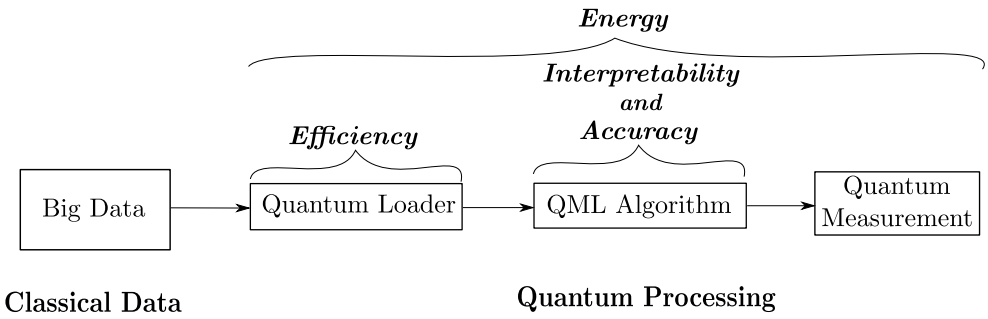


Figure 5.21: Identified advantages of the proposed hybrid approaches.

5.6 General Observations

The following overall advantages as shown in Fig. 5.21are observed at each stage of our research in big data processing using quantum machine learning.

1. *Efficiency*: Classical data can be encoded into qubits efficiently using quantum loaders. Encoding schemes such as amplitude encoding uses only n qubits to encode 2^n classical values as amplitudes of quantum states.

2. *Interpretability*: Quantum machine learning algorithms are interpretable as algorithm is designed as a quantum circuit. The quantum operations in the circuit can be interpreted to observe the information processing in the quantum machine learning algorithm.

3. *Accuracy*: Classical machine learning algorithms can be enhanced using quantum computational techniques to improve performance accuracy. Specifically, hybrid computation can be used to improve the accuracy of the machine learning models in real-time high-impact applications.

4. *Energy (expected)*: As quantum computational hardware can provide exponential scaling, overall energy utilization in solving a complex problem on a classical computer can be reduced using a quantum computer. Critical problems can be partially solved on a quantum computer, and the results can be postprocessed using classical computations. Hence, performing huge computations on high-performance classical hardware can be avoided, and energy can be saved.

5.7 Summary

This chapter presented the details of different quantum machine learning techniques designed for supervised learning on spatial data. We first developed quantum-enhanced deep neural network architecture for satellite image scene classification. We observed that data augmentation plays a crucial role in achieving good classification accuracy during the experimentation. Hence we proposed a quantum data augmentation technique to increase the size of the dataset. As CNNs consist of huge parameters for training, introducing models such as HybridQC and HQCNN reduced the trainable parameters for scene classification on the spatial data. From the overall general observations, quantum computing techniques can be used to enhance the machine learning models. In the next chapter, we present the details of quantum processing techniques used to process SAR images for deep learning.

Chapter 6

Quantum Processing of Synthetic Aperture Radar Images for Deep Learning

"What we usually consider are impossible are simply engineering problems . . .
there's no law of physics preventing them."
– MICHIO KAKU

Remote sensing applications generally require intense processing of satellite images for analysis. Active remote sensing, such as Synthetic Aperture Radar (SAR), produces images with higher spectral information when compared to optical sensors. Synthetic Aperture Radar (SAR) images are difficult to comprehend as they look different compared to optical Earth Observation images. SAR is a type of radar that is generally used to take 2D images or reconstructions of 3D objects from their 2D images such as terrain surfaces and water bodies [130]. SAR imagery comes under the category of active data collection because the sensor produces energy waves to illuminate the target surface and then records the amount of energy reflected after interacting with the target surface. Unlike optical image sensors, the SAR signal is responsive to surface characteristics such as structure, moisture, and depth.

SAR sensors are capable of producing images with microwave frequencies with the help of a processing method that mimics a long antenna aperture [131]. SAR sensor is generally fixed on a rigid platform that is in motion, such as a spacecraft or a high-altitude aircraft, and the movement of a radar antenna is used over a target surface [132]. However, the visual complexity in the case of SAR images leads to difficulty in comprehending and processing the image data. Hence, deep learning models face difficulty in processing SAR images for remote sensing applications [133]. This chapter presents the details of quantum processing techniques developed to process SAR images for deep learning.

6.1 Quantum Processing Techniques for SAR Images

Every ML system is a combination of code that consists of a model/algorithm and data. Hence, two approaches are used to build an ML system. The first is a model-centric approach where a model is tuned to improve performance. The second is data-centric, an approach where data used for training the models is systematically processed to improve the model's efficiency. For many years, model-centric approaches have been followed where the model hyperparameters are tuned during training to improve the efficiency of the models.

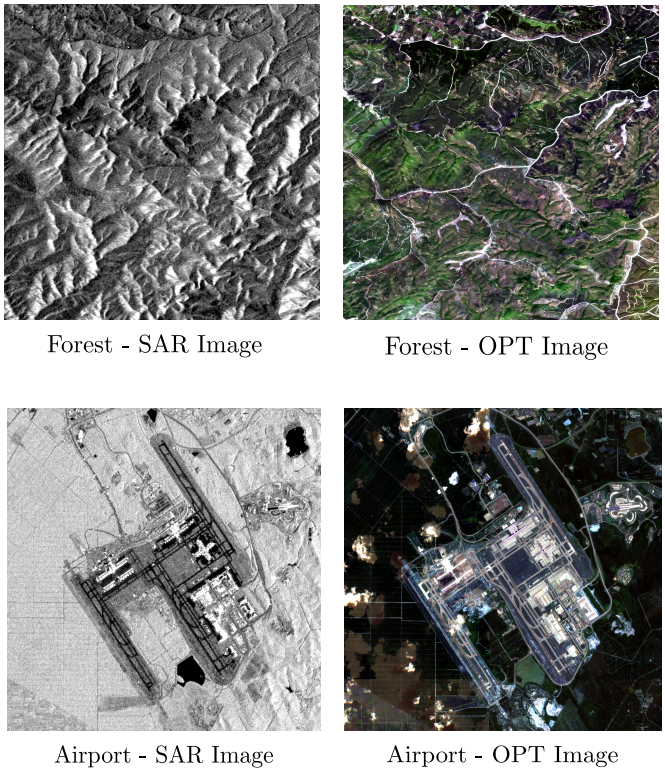


Figure 6.1: Comparison of SAR and OPT images from Google Earth Engine.

We used quantum computing for a data-centric approach and proposed new techniques to process the SAR images. As shown in Fig. 6.1, SAR images are different from optical (OPT) images. The visually observable features are more in OPT images than SAR images. Hence, deep learning models such as CNNs face challenges in dealing with SAR images [133]. We designed two quantum processing techniques (QPTs) where SAR images are systematically processed using quantum circuits. In the following sections, we describe the quantum operations involved in the designed QPTs.

6.1.1 Quantum Processing Technique-1

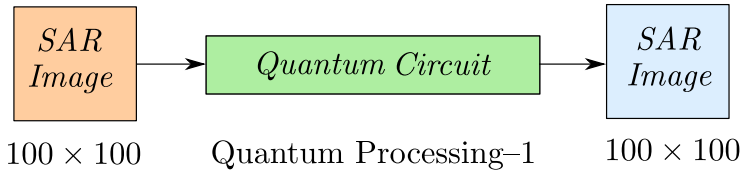


Figure 6.2: Illustration of QPT-1.

In the first quantum processing technique (QPT-1) given in Fig. 6.2, we resized the images to 100×100 size before processing. In the next step, each pixel value of the SAR image is given as input (θ) for the rotational angle of $R_y(\theta)$ gate as shown in the Fig. 6.3. We used a four-qubit circuit to process the SAR images. The pixel values $\{a, b, c, d\}$ are given as input to the rotational gates of the quantum circuit. Iteratively, all the pixel values of each image are processed using the same circuit.

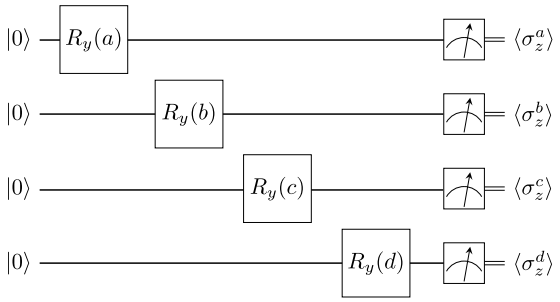


Figure 6.3: Illustration of quantum circuit for QPT-1.

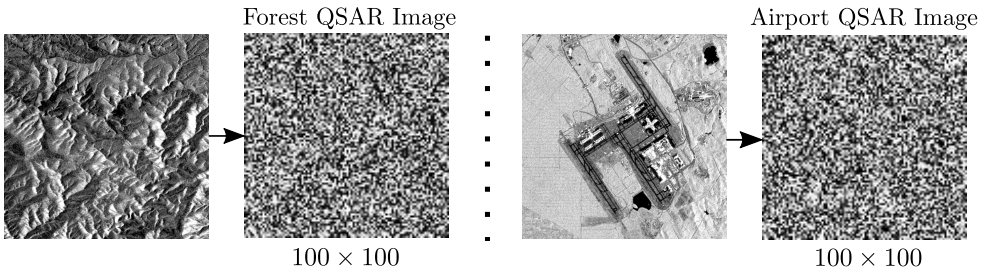


Figure 6.4: QSAR images using QPT-1.

After the rotational gate operation, we perform *Pauli-Z* measurement operation to obtain the output values between $[-1, 1]$. The output values are rescaled to $[0, 255]$ to obtain a grayscale image from the measurement output values. Thus, a processed image is obtained using quantum processing. Fig. 6.4 shows the quantum processed SAR (QSAR) images using QPT-1.

6.1.2 Quantum Processing Technique-2

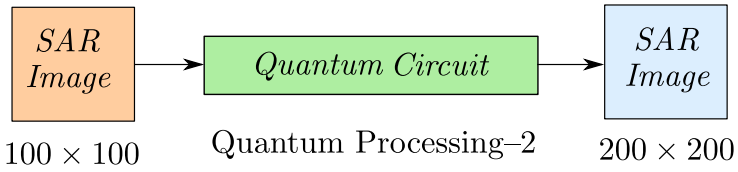


Figure 6.5: Illustration of QPT-2.

As shown in Fig. 6.5, the second quantum processing technique is used to increase the size of the image using quantum processing. Each pixel value of the image is repeated on the four qubits using the R_y operation as shown in Fig. 6.6. All the four output values from the *Pauli-Z* are used to create a 2×2 block for each input pixel. Hence, for an input size of 100×100 image, we obtain a 200×200 QSAR image.

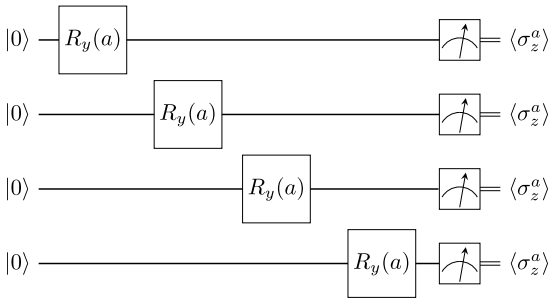


Figure 6.6: Illustration of quantum circuit for QPT-2.

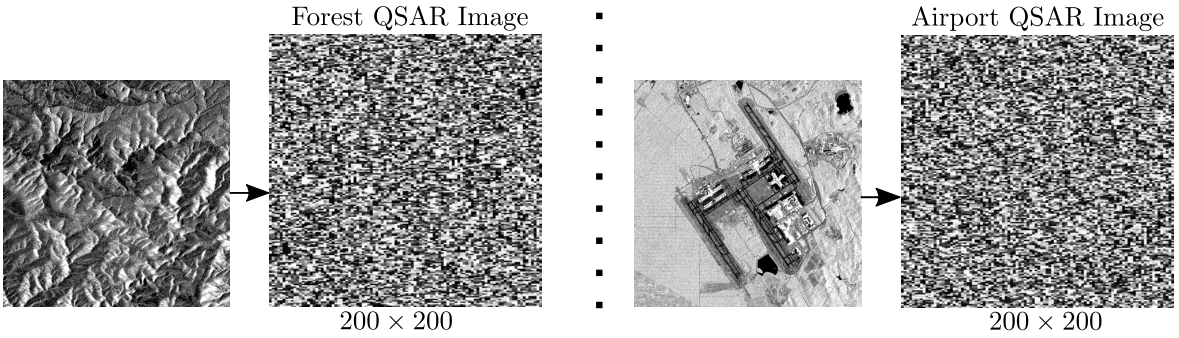


Figure 6.7: QSAR images using QPT-2.

As mentioned in the previous chapters, there exists a limited availability of the qubits on QPUs. Hence, we used four qubit circuits for both QPT-1 and QPT-2. Fig. 6.7 shows the QSAR images using QPT-2. The following section provides the details of the quality metrics of the images created using quantum processing techniques compared to the original SAR images.

6.1.3 Quality metrics of images using QPTs

Quantum processing of images is a novel approach where SAR images are processed using a quantum circuit, thereby, obtaining a new image from the qubit measurement output values. In the following, we calculate the quality metrics of QSAR images using the most widely used metrics [134].

Relative Bias

The absolute difference between the mean of the original SAR image pixels and the QSAR images divided by the mean of the original SAR image is considered as relative bias (RB) [135]. The ideal value of RB is 0.

Relative Variance

Relative Variance (RV) is calculated by subtracting the variance of the QSAR image from the variance of the original SAR image, divided by the variance of the original SAR image [135]. The ideal value of RV is 0.

Universal Image Quality Index

A universal image quality index (UIQI) is also called a Q-index. In UIQI, QSAR image distortion is modeled as a combination of loss of correlation, luminance distortion, and contrast distortion [136]. The output value ranges from -1 to 1, and the value must be as high as possible.

Entropy

The richness of information in the QSAR image data can be calculated using entropy [137]. Entropy is used to estimate the information contained in the image processed using QPTs. The calculation of entropy (E) is given as

$$E = -\sum_{i=0}^{L-1} p_i \log p_i \tag{6.1}$$

where L represents the gray scale levels. The value p_i represents the ratio of the pixels with gray scale value equivalent to i to the total number of pixels in the image. The higher the entropy value higher is the information in the QSAR image.

Table 6.1: Quality metrics comparison of QPTs with original SAR images

	Relative Bias	Relative Variance	Universal Image Quality Index	Entropy	Spatial Frequency
Original SAR	0.000	0.000	1.000	5.223	24.534
QPT-1 with Despeckling	0.137	1.610	0.686	**5.324**	16.635
QPT-1 without Despeckling	0.138	1.616	0.687	4.816	17.998
QPT-2 with Despeckling	0.137	1.610	0.667	**5.324**	**28.752**
QPT-2 without Despeckling	0.138	1.616	0.670	4.816	**31.105**

Spatial Frequency

Spatial frequency (SF) measures the overall activity level in the QSAR image. SF is expressed in terms of row frequency and column frequency [138]. The value of SF must be as high as possible.

Table 6.1 shows the quality metrics calculated on the sample images collected from Google Earth Engine. Despeckling refers to the process of removing speckle noise from the images. Quality metrics of the QSAR images processed using QPT-2 techniques are extremely well in terms of entropy and spatial frequency. In the next section, we present the performance analysis of the deep learning models in terms of classification accuracy using QSAR images.

6.2 Performance Analysis of DL models using QSAR Images

In this section, we evaluate the performance of DL models using the images obtained from quantum processing (QPT-1 and QPT-2). We performed the experiments on the OpenSARUrban collection [139, 140] that provides image patches of urban SAR scenes. We synthesized the dataset into multiple datasets of two classes, four classes, six classes, and ten classes. Fig. 6.8 shows the sample images from different classes in the OpenSARUrban dataset.

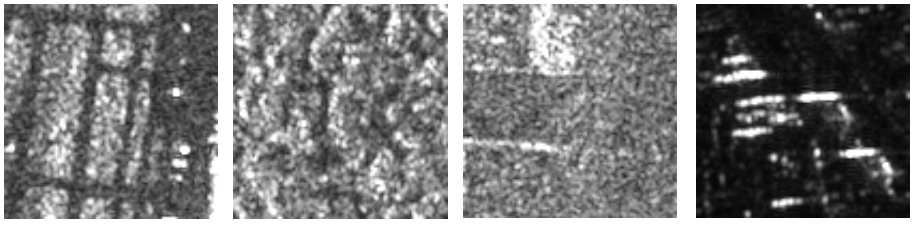


Figure 6.8: SAR images from OpenSARUrban dataset.

Table 6.2: Accuracy (%) of deep learning models on QPTs

	Images				
	SAR	QPT-1	SAR+QPT-1	QPT-2	SAR+QPT-2
2 classes	**73.08**	58.87	69.70	66.68	70.64
4 classes	**69.55**	58.64	61.13	61.89	68.04
6 classes	66.23	56.33	60.98	60.43	**66.79**
10 classes	63.59	54.77	59.65	59.07	**64.17**

All the deep learning models are implemented with CNN architecture using *TensorFlow* package. Table 6.2 shows the accuracy of the deep learning models on different images for the synthesized OpenSARUrban dataset. Using original SAR images along with QSAR images from QPT-2 for training the DL models, outperformed all the other categories. We used *ibm_santiago* quantum computer provided by IBM QX to check the compatibility of circuits used for QPT-1 and QPT-2. Fig. 6.9 and Fig. 6.10 show the output of the quantum circuits implemented using the basic quantum gates of *ibm_santiago* on sample pixel values.

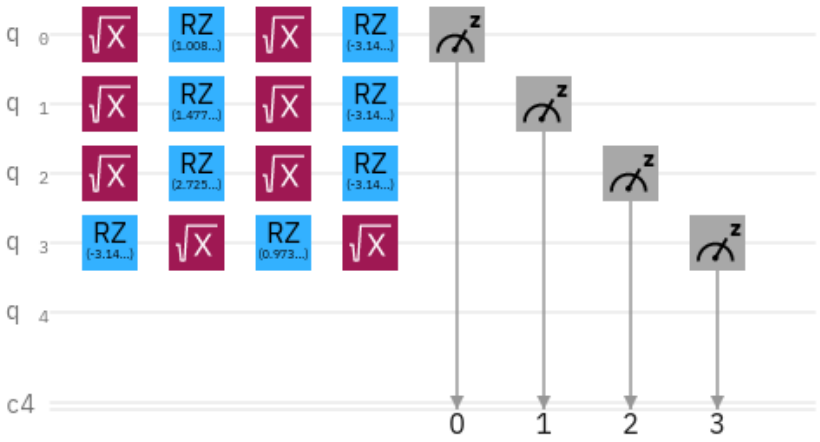


Figure 6.9: Quantum circuit for QPT-1 on ibm_santiago.

6.3 Summary

This chapter presented the importance of the data-centric approach to building machine learning models. As DL models face many challenges in processing SAR images, we proposed two quantum processing techniques that can enhance the performance of DL models by systematically processing SAR images. We observed that the quality metrics of the

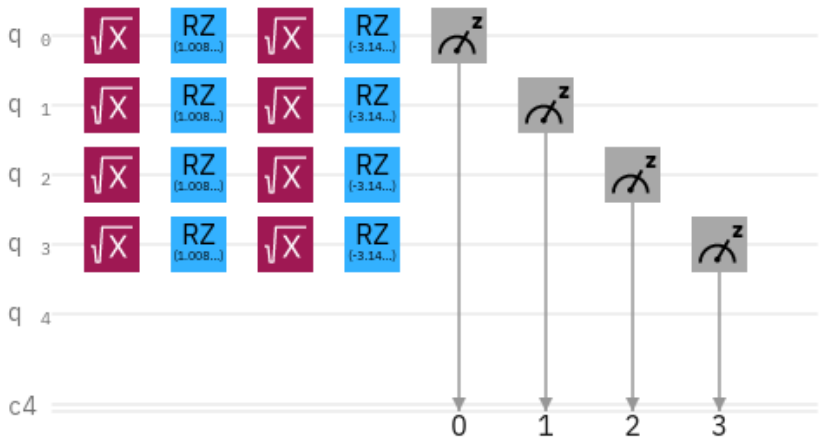


Figure 6.10: Quantum circuit for QPT-2 on ibm_santiago.

QSAR images improved after the quantum processing. Our results showed the improvement in the overall classification accuracy of the DL models using both SAR and QSAR images for training the models.

Chapter 7

Conclusions and Future Directions

"Often when you think you're at the end of something,
you're at the beginning of something else."
– FRED ROGERS

This thesis work investigated some of the open questions in the context of the evolution of quantum machine learning for big data analytics. In particular, we observed the impact of quantum computational techniques in handling big data for supervised learning. We found that the hybrid quantum-classical approach can overcome some of the limitations of existing quantum computers such as availability of qubits. Also, quantum and classical computations together can efficiently handle machine learning problems related to classical spatial big data.

In the following, we summarize the major conclusions of the thesis. An ANN is proposed with qubits as artificial neurons to study the impact of quantum computing in machine learning. The proposed quantum computing approach for ANN (QC ANN) efficiently encoded data into a quantum state using the amplitude encoding method. QC ANN can perform binary classification with more accuracy than classical ANN on the benchmark dataset. Next, a quantum loader is proposed using single-qubit encoding to encode each value of classical data into one qubit. A variational circuit with rotational gates and *CNOT* gates is designed for multi-class classification (QMCC). We observed that using more quantum operations in the circuit for processing also increased the accuracy of the model. We observed that there is a lack of non-linearity in both QC ANN and QMCC. Non-linearity is an important aspect that helps to construct a generalized model using machine learning. Hence, we separated the feature extraction process from the quantum machine learning algorithm and used the features to train a classical non-linear model in the next step. Extracting quantum representation of the classical data helped the classical machine learning techniques to construct a generalized model with fewer parameters

using HQCNN. The combined power of quantum computation to process data in a high-dimensional Hilbert space with the efficiency of deep neural networks is used to construct a generalized model for classification tasks. We then proposed a quantum circuit to add noise into the data, and adding noise to enhance the dataset is a popular technique in data augmentation. Further in the work, quantum operations are used in between CNN layers to enhance the training process of CNNs. We also presented a detailed performance analysis and comparative study to prove the efficiency of the models designed in our research. Finally, we proposed and evaluated a data-centric approach to building ML models using quantum processing on SAR images.

7.1 Future Directions

In this section, a few possible future extensions of the work done in this thesis for different models are discussed.

Data-specific qubit encoding Information encoding on qubits is a crucial step for any quantum information processing method. Due to the rapid generation of big data, a lot of scope exists to identify data-specific qubit encoding schemes. Most importantly, qubit encoding schemes for images differ from qubit encoding schemes for numerical data. There is a need to develop efficient qubit encoding schemes for different data types.

Hybrid models for numerical data The future direction of the hybrid models for numerical data presented in the thesis includes a study of the advantages of quantum computing on deep learning when working on datasets of large size and higher dimensions. Further research can be carried out to improvise the quantum circuit or design a new circuit suitable for large-scale data implementation on a quantum computer. Also, identifying the application of hybrid models on numerical data is crucial.

Non-linear hybrid models for spatial data The future scope of the work presented in the thesis is to study the impact of different quantum circuits on quantum representations of spatial data, such as images. The future direction of the work can be developing more complex hybrid quantum-classical models for sensor signal processing. Also, information processing and the impact of the fusion of quantum-processed SAR images and optical images can be performed using quality metrics for evaluation. A study can also be performed on the accuracy improvements of DL models using the fused images.

Non-linear quantum models for big data There exist a lot of scope to develop and explore the impact of non-linear quantum models for quantum machine learning on big data. As the quantum computational power in the future increases, there is an immediate scope to study and develop non-linear quantum models for large-scale data processing. The future direction also includes identifying fixed circuits to solve a common problem in specific applications.

Bibliography

[1] J. R. David Reinsel, John Gantz, "The digitization of the world from edge to core," *Framingham: International Data Corporation*, p. 16, 2018. [Online]. Available: http://cloudcode.me/media/1014/idc.pdf

[2] J.-G. Lee and M. Kang, "Geospatial big data: challenges and opportunities," *Big Data Research*, vol. 2, no. 2, pp. 74–81, Jun. 2015.

[3] R. Kune, P. K. Konugurthi, A. Agarwal, R. R. Chillarige, and R. Buyya, "The anatomy of big data computing," *Software: Practice and Experience*, vol. 46, no. 1, pp. 79–105, Oct. 2016.

[4] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE transactions on knowledge and data engineering*, vol. 26, no. 1, pp. 97–107, Jun. 2013.

[5] How much data is created every day in 2022? [Online]. Available: https://techjury.net/blog/how-much-data-is-created-every-day/#gref

[6] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data," *Neurocomput.*, vol. 237, no. C, pp. 350–361, May 2017.

[7] L. Eeckhout, "Is moore's law slowing down? what's next?" *IEEE Micro*, vol. 37, no. 04, pp. 4–5, Aug. 2017.

[8] G. García-Pérez, M. A. Rossi, and S. Maniscalco, "Ibm q experience as a versatile experimental testbed for simulating open quantum systems," *npj Quantum Information*, vol. 6, no. 1, pp. 1–10, 2020.

[9] "Google moves toward quantum supremacy with 72-qubit computer." [Online]. Available: https://www.sciencenews.org/article/google-moves-toward-quantum-supremacy-72-qubit-computer

[10] M. Ohzeki, "Breaking limitation of quantum annealer in solving optimization problems under constraints," *Scientific reports*, vol. 10, no. 1, pp. 1–12, Feb. 2020.

[11] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Physical review letters*, vol. 113, no. 13, p. 130503, 2014.

[12] J. D. Hidary and J. D. Hidary, *Quantum computing: an applied approach*. Springer, 2019, vol. 1.

[13] R. P. Feynman, "Simulating physics with computers," *International journal of theoretical physics*, vol. 21, no. 6, pp. 467–488, 1982.

[14] P. Brumer and J. Gong, "Born rule in quantum and classical mechanics," *Phys. Rev. A*, vol. 73, p. 052109, May 2006. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.73.052109

[15] M. Schuld and F. Petruccione, *Supervised Learning with Quantum Computers*. Springer, 2018, vol. 17.

[16] M. Fingerhuth, T. Babej, and P. Wittek, "Open source software in quantum computing," *PloS one*, vol. 13, no. 12, p. e0208561, 2018.

[17] Y. S. Weinstein, M. Pravia, E. Fortunato, S. Lloyd, and D. G. Cory, "Implementation of the quantum fourier transform," *Physical review letters*, vol. 86, no. 9, p. 1889, 2001.

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

[19] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Phys. Rev. Lett.*, vol. 103, p. 150502, Oct 2009.

[20] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum algorithms for supervised and unsupervised machine learning," *arXiv preprint arXiv:1307.0411*, 2013.

[21] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.

[22] S. Resch and U. R. Karpuzcu, "Benchmarking quantum computers and the impact of quantum noise," *ACM Comput. Surv.*, vol. 54, no. 7, Jul. 2021.

[23] O. Gamel, "Entangled bloch spheres: Bloch matrix and two-qubit state space," *Physical Review A*, vol. 93, no. 6, p. 062320, 2016.

[24] M. Möttönen, J. J. Vartiainen, V. Bergholm, and M. M. Salomaa, "Transformation of quantum states using uniformly controlled rotations," *Quantum Info. Comput.*, vol. 5, no. 6, p. 467–473, 2005.

[25] M. Schuld and F. Petruccione, *Information Encoding*. Springer International Publishing, 2018, pp. 139–171.

[26] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018.

[27] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik, "The theory of variational hybrid quantum-classical algorithms," *New Journal of Physics*, vol. 18, no. 2, p. 023023, 2016.

[28] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature communications*, vol. 5, p. 4213, 2014.

[29] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii, "Quantum circuit learning," *Physical Review A*, vol. 98, no. 3, p. 032309, 2018.

[30] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, "Parameterized quantum circuits as machine learning models," *Quantum Science and Technology*, vol. 4, no. 4, p. 043001, 2019.

[31] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, p. 195, 2017.

[32] I. Kerenidis, *Quantum Machine Learning: Algorithms and Applications*. QC Ware, Practical Quantum Computing Conference (Q2B 2019), vol. Keynote Talk. [Online]. Available: https://q2b.qcware.com/

[33] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum algorithms for supervised and unsupervised machine learning," 2013.

[34] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum principal component analysis," *Nature Physics*, vol. 10, no. 9, pp. 631–633, Sep 2014.

[35] N. Wiebe, D. Braun, and S. Lloyd, "Quantum algorithm for data fitting," *Phys. Rev. Lett.*, vol. 109, p. 050505, Aug 2012.

[36] A. M. Childs, R. Kothari, and R. D. Somma, "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision," *SIAM Journal on Computing*, vol. 46, no. 6, pp. 1920–1950, 2017.

[37] D. Michie, D. J. Spiegelhalter, C. Taylor *et al.*, "Machine learning," *Neural and Statistical Classification*, vol. 13, no. 1994, pp. 1–298, 1994.

[38] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[39] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[40] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[41] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[42] Q. Zou, L. Ni, T. Zhang, and Q. Wang, "Deep learning based feature selection for remote sensing scene classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 11, pp. 2321–2325, 2015.

[43] X. Lu, X. Zheng, and Y. Yuan, "Remote sensing scene classification by unsupervised representation learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5148–5157, 2017.

[44] Q. Liu, R. Hang, H. Song, and Z. Li, "Learning multiscale deep features for high-resolution satellite image scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 1, pp. 117–126, 2018.

[45] X. Zheng, Y. Yuan, and X. Lu, "A deep scene representation for aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 7, pp. 4799–4809, 2019.

[46] Y. Gu, Y. Wang, and Y. Li, "A survey on deep learning-driven remote sensing image scene understanding: Scene classification, scene retrieval and scene-guided object detection," *Applied Sciences*, vol. 9, no. 10, p. 2110, 2019.

[47] K. Nogueira, O. A. Penatti, and J. A. Dos Santos, "Towards better exploiting convolutional neural networks for remote sensing scene classification," *Pattern Recognition*, vol. 61, pp. 539–556, 2017.

[48] W. Zhang, P. Tang, and L. Zhao, "Remote sensing image scene classification using cnn-capsnet," *Remote Sensing*, vol. 11, no. 5, p. 494, 2019.

[49] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais *et al.*, "Deep learning and process understanding for data-driven earth system science," *Nature*, vol. 566, no. 7743, pp. 195–204, 2019.

[50] L. Ma, Y. Liu, X. Zhang, Y. Ye, G. Yin, and B. A. Johnson, "Deep learning in remote sensing applications: A meta-analysis and review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 152, pp. 166–177, 2019.

[51] M. Paoletti, J. Haut, J. Plaza, and A. Plaza, "Deep learning classifiers for hyperspectral imaging: A review," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 158, pp. 279–317, 2019.

[52] Q. Nguyen and M. Hein, "Optimization landscape and expressivity of deep cnns," in *International conference on machine learning*. PMLR, 2018, pp. 3730–3739.

[53] F. Bach, "Breaking the curse of dimensionality with convex neural networks," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 629–681, 2017.

[54] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *arXiv preprint arXiv:1710.09282*, 2017.

[55] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, "A survey of methods for explaining black box models," *ACM computing surveys (CSUR)*, vol. 51, no. 5, pp. 1–42, 2018.

[56] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[57] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, 10th ed. New York, NY, USA: Cambridge University Press, 2011.

[58] P. Wittek, *Quantum machine learning: what quantum computing means to data mining*. Academic Press, 2014.

[59] M. Schuld, M. Fingerhuth, and F. Petruccione, "Implementing a distance-based classifier with a quantum interference circuit," *EPL (Europhysics Letters)*, vol. 119, no. 6, p. 60002, 2017.

[60] E. Grant, M. Benedetti, S. Cao, A. Hallam, J. Lockhart, V. Stojevic, A. G. Green, and S. Severini, "Hierarchical quantum classifiers," *npj Quantum Information*, vol. 4, no. 1, p. 65, 2018.

[61] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, "Supervised learning with quantum-enhanced feature spaces," *Nature*, vol. 567, no. 7747, p. 209, 2019.

[62] F. Tacchino, C. Macchiavello, D. Gerace, and D. Bajoni, "An artificial neuron implemented on an actual quantum processor," *npj Quantum Information*, vol. 5, no. 1, p. 26, 2019.

[63] V. Dunjko, J. M. Taylor, and H. J. Briegel, "Quantum-enhanced machine learning," *Physical review letters*, vol. 117, no. 13, p. 130501, 2016.

[64] M. Schuld and N. Killoran, "Quantum machine learning in feature hilbert spaces," *Phys. Rev. Lett.*, vol. 122, p. 040504, 2019.

[65] R. Mengoni and A. Di Pierro, "Kernel methods in quantum machine learning," *Quantum Machine Intelligence*, vol. 1, no. 3, pp. 65–71, 2019.

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

[67] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, "Quantum boltzmann machine," *Physical Review X*, vol. 8, no. 2, p. 021050, 2018.

[68] S. Lloyd and C. Weedbrook, "Quantum generative adversarial learning," *Physical review letters*, vol. 121, no. 4, p. 040502, 2018.

[69] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, "Quantum machine learning: a classical perspective," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2209, p. 20170551, 2018.

[70] O. A. Von Lilienfeld, "Quantum machine learning in chemical compound space," *Angewandte Chemie International Edition*, vol. 57, no. 16, pp. 4164–4169, 2018.

[71] Y. Cai, X. Lu, and N. Jiang, "A survey on quantum image processing," *Chinese Journal of Electronics*, vol. 27, no. 4, pp. 718–727, 2018.

[72] P. Xu, Z. He, T. Qiu, and H. Ma, "Quantum image processing algorithm using edge extraction based on kirsch operator," *Optics express*, vol. 28, no. 9, pp. 12 508–12 517, 2020.

[73] M. Islam, M. Chowdhury, Z. Khan, and S. Khan, "Hybrid quantum-classical neural network for cloud-supported in-vehicle cyberattack detection," *IEEE Sensors Letters*, vol. Early Access Article, 2021.

[74] C. Li, K. K. Tokgoz, M. Fukawa, J. Bartels, T. Ohashi, K.-i. Takeda, and H. Ito, "Data augmentation for inertial sensor data in cnns for cattle behavior classification," *IEEE Sensors Letters*, vol. 5, no. 11, pp. 1–4, Oct. 2021.

[75] N. Dawar, S. Ostadabbas, and N. Kehtarnavaz, "Data augmentation in deep learning-based fusion of depth and inertial sensing for action recognition," *IEEE Sensors Letters*, vol. 3, no. 1, pp. 1–4, Jan. 2019.

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

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

[78] M. Henderson, S. Shakya, S. Pradhan, and T. Cook, "Quanvolutional neural networks: powering image recognition with quantum circuits," *Quantum Machine Intelligence*, vol. 2, no. 1, pp. 1–9, 2020.

[79] A. Mari, T. R. Bromley, J. Izaac, M. Schuld, and N. Killoran, "Transfer learning in hybrid classical-quantum neural networks," *Quantum*, vol. 4, p. 340, 2020.

[80] L. Gyongyosi and S. Imre, "Training optimization for gate-model quantum neural networks," *Scientific reports*, vol. 9, no. 1, pp. 1–19, 2019.

[81] L. Gyongyosi and S. Imre, "State stabilization for gate-model quantum computers," *Quantum Information Processing*, vol. 18, no. 9, pp. 1–22, 2019.

[82] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, "Circuit-centric quantum classifiers," *Phys. Rev. A*, vol. 101, p. 032308, Mar 2020. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.101.032308

[83] S. V. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sensing of Environment*, vol. 62, no. 1, pp. 77–89, 1997. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0034425797000837

[84] G. García-Pérez, M. A. C. Rossi, and S. Maniscalco, "Ibm q experience as a versatile experimental testbed for simulating open quantum systems," *npj Quantum Information*, vol. 6, no. 1, p. 1, Jan. 2020.

[85] T. Draper and S. Kutin, "Qpic: Quantum circuit diagrams in latex," 2016. [Online]. Available: https://github.com/qpic/qpic

[86] E. Stoudenmire and D. J. Schwab, "Supervised learning with tensor networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 4799–4807.

[87] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing*. Springer, 1990, pp. 227–236.

[88] M. Schuld, V. Bergholm, C. Gogolin, J. Izaac, and N. Killoran, "Evaluating analytic gradients on quantum hardware," *Physical Review A*, vol. 99, no. 3, p. 032331, 2019.

[89] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, and N. Killoran, "Pennylane: Automatic differentiation of hybrid quantum-classical computations," *arXiv preprint arXiv:1811.04968*, 2018.

[90] "UCI Machine Learning Repository: Iris Data Set," Aug 1988. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Iris

[91] "UCI Machine Learning Repository: Banknote Authentication Data Set," Aug 2013. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/banknote+authentication

[92] "UCI Machine Learning Repository: Wireless Indoor Localization Data Set," Aug 2017. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Wireless+Indoor+Localization

[93] W. N. S. William H. Wolberg and O. L. Mangasarian, "UCI machine learning repository," 1995.

[94] F. Chollet *et al.*, "Keras: The python deep learning library," *ascl*, pp. ascl–1806, 2018.

[95] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, 2013, pp. 1139–1147.

[96] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[97] Q. c. IBM Research, "Qiskit," March 2017, [Online; accessed 28. Aug. 2020]. [Online]. Available: https://github.com/Qiskit

[98] PennyLaneAI, "pennylane-qiskit," November 2018, [Online; accessed 28. Aug. 2020]. [Online]. Available: https://github.com/PennyLaneAI/pennylane-qiskit

[99] "IBM Q 16 Melbourne. ibmq-device-information," Aug 2019. [Online]. Available: https://github.com/Qiskit/ibmq-device-information/blob/master/backends/melbourne/V1/version_log.md

[100] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[101] L. Prechelt, "Early stopping-but when?" in *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[102] M. Schuld and F. Petruccione, *Quantum Information*. Springer International Publishing, 2018, pp. 75–125.

[103] K. Johnston, J. M. Ver Hoef, K. Krivoruchko, and N. Lucas, *Using ArcGIS geostatistical analyst*. Esri Redlands, 2001, vol. 380.

[104] A. Dahlgren, "Geographic accessibility analysis-methods and application," Ph.D. dissertation, Real Estate Science, Department of Technology and Society, Lund University, 2008.

[105] M. Ghahramani, M. Zhou, and C. T. Hon, "Mobile phone data analysis: A spatial exploration toward hotspot detection," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 351–362, 2019.

[106] M. Molinari, M.-R. Fida, M. K. Marina, and A. Pescape, "Spatial interpolation based cellular coverage prediction with crowdsourced measurements," in *Proceedings of the 2015 ACM SIGCOMM Workshop on Crowdsourcing and Crowdsharing of Big (Internet) Data*. ACM, 2015, pp. 33–38.

[107] P. Raghubir, V. G. Morwitz, and A. Chakravarti, "Spatial categorization and time perception: Why does it take less time to get home?" *Journal of Consumer Psychology*, vol. 21, no. 2, pp. 192 – 198, 2011.

[108] D. L. Bandalos and S. J. Finney, "Factor analysis: Exploratory and confirmatory," in *The reviewer's guide to quantitative methods in the social sciences*. Routledge, 2018, pp. 98–122.

[109] M. Chen, D. Liu, K. Qian, J. Li, M. Lei, and Y. Zhou, "Lunar crater detection based on terrain analysis and mathematical morphology methods using digital elevation models," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 7, pp. 3681–3692, 2018.

[110] J. F. Roddick and M. Spiliopoulou, "A bibliography of temporal, spatial and spatio-temporal data mining research," *ACM SIGKDD Explorations Newsletter*, vol. 1, no. 1, pp. 34–38, 1999.

[111] Y. Zheng and X. Zhou, *Computing with spatial trajectories*. Springer Science & Business Media, 2011.

[112] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, 2010, pp. 270–279.

[113] G.-S. Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, L. Zhang, and X. Lu, "Aid: A benchmark data set for performance evaluation of aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3965–3981, 2017.

[114] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: Benchmark and state of the art," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1865–1883, 2017.

[115] L. Tong, J. He, and L. Peng, "Cnn-based pd hand tremor detection using inertial sensors," *IEEE Sensors Letters*, vol. 5, no. 7, pp. 1–4, Jul. 2021.

[116] I. Goel, N. B. Puhan, and B. Mandal, "Deep convolutional neural network for double-identity fingerprint detection," *IEEE Sensors Letters*, vol. 4, no. 5, pp. 1–4, May. 2020.

[117] H. Zhu, W. Wang, and R. Leung, "Sar target classification based on radar image luminance analysis by deep learning," *IEEE Sensors Letters*, vol. 4, no. 3, pp. 1–4, Mar. 2020.

[118] M. Schuld and F. Petruccione, *Machine Learning with Quantum Computers*, 2nd ed. Springer Nature Switzerland: Springer, 2021.

[119] A. Chalumuri, R. Kune, S. Kannan, and B. S. Manoj, "Quantum-enhanced deep neural network architecture for image scene classification," *Quantum Information Processing*, vol. 20, no. 11, p. 381, Nov. 2021.

[120] D. A. Zaidenberg, A. Sebastianelli, D. Spiller, B. Le Saux, and S. L. Ullo, "Advantages and bottlenecks of quantum machine learning for remote sensing," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. IEEE, Oct. 2021, pp. 5680–5683.

[121] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, Mar. 2019.

[122] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, Mar. 2021.

[123] C. M. Bishop, "Training with noise is equivalent to tikhonov regularization," *Neural computation*, vol. 7, no. 1, pp. 108–116, Jan. 1995.

[124] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: Benchmark and state of the art," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1865–1883, 2017.

[125] D. Zeng, M. Liao, M. Tavakolian, Y. Guo, B. Zhou, D. Hu, M. Pietikäinen, and L. Liu, "Deep learning for scene classification: A survey," *arXiv preprint arXiv:2101.10531*, 2021.

[126] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015.

[127] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[128] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[129] A. López-Cifuentes, M. Escudero-Viñolo, J. Bescós, and Á. García-Martín, "Semantic-aware scene recognition," *Pattern Recognition*, vol. 102, p. 107256, 2020.

[130] M. Kirscht and C. Rinke, "3d reconstruction of buildings and vegetation from synthetic aperture radar (sar) images." in *MVA*. Citeseer, 1998, pp. 228–231.

[131] L. Cutrona, "Synthetic aperture radar," *Radar handbook*, vol. 2, pp. 2333–2346, 1990.

[132] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, "A tutorial on synthetic aperture radar," *IEEE Geoscience and remote sensing magazine*, vol. 1, no. 1, pp. 6–43, 2013.

[133] X. X. Zhu, S. Montazeri, M. Ali, Y. Hua, Y. Wang, L. Mou, Y. Shi, F. Xu, and R. Bamler, "Deep learning meets sar," *arXiv preprint arXiv:2006.10027*, 2020.

[134] S. C. Kulkarni and P. P. Rege, "Pixel level fusion techniques for sar and optical images: A review," *Information Fusion*, vol. 59, pp. 13–29, 2020.

[135] L. Wald, T. Ranchin, and M. Mangolini, "Fusion of satellite images of different spatial resolutions: Assessing the quality of resulting images," *Photogrammetric engineering and remote sensing*, vol. 63, no. 6, pp. 691–699, 1997.

[136] Z. Wang and A. C. Bovik, "Image and multidimensional signal processing-a universal image quality index," *IEEE Signal Processing Letters*, vol. 9, no. 3, pp. 81–84, 2002.

[137] N. Yin and Q.-g. Jiang, "Feasibility of multispectral and synthetic aperture radar image fusion," in *2013 6th International Congress on Image and Signal Processing (CISP)*, vol. 2. IEEE, 2013, pp. 835–839.

[138] Y. Zheng, E. A. Essock, B. C. Hansen, and A. M. Haun, "A new metric based on extended spatial frequency and its application to dwt based fusion algorithms," *Information Fusion*, vol. 8, no. 2, pp. 177–192, 2007.

[139] J. Zhao, "Opensarurban," 2019. [Online]. Available: https://dx.doi.org/10.21227/3sz0-dp26

[140] J. Zhao, Z. Zhang, W. Yao, M. Datcu, H. Xiong, and W. Yu, "Opensarurban: A sentinel-1 sar image dataset for urban interpretation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 187–203, 2020.

List of Publications

Refereed Journals

1. **Avinash Chalumuri**, Raghavendra Kune, S. Kannan, and B. S. Manoj. "Quantum-Classical Image Processing for Scene Classification." IEEE Sensors Letters, vol. 6, no. 6, pp. 1-4, May 2022. DOI: https://doi.org/10.1109/LSENS.2022.3173253

2. **Avinash Chalumuri**, Raghavendra Kune, S. Kannan, and B. S. Manoj. "Quantum-enhanced deep neural network architecture for image scene classification." Quantum Information Processing, Springer Nature, Vol. 20, Issue. 11, 1-21, November 2021. DOI: https://doi.org/10.1007/s11128-021-03314-7

3. **Avinash Chalumuri**, Raghavendra Kune, and B. S. Manoj. "A hybrid classical-quantum approach for multi-class classification." Quantum Information Processing, Springer Nature, Vol. 20, Issue. 3, 1-19, March 2021.
DOI: https://doi.org/10.1007/s11128-021-03029-9

Refereed Conferences

1. **Avinash Chalumuri**, Raghavendra Kune, and B. S. Manoj. "Training an Artificial Neural Network Using Qubits as Artificial Neurons: A Quantum Computing Approach", in The Third International Conference on Computing and Network Communications (CoCoNet'19) December 2019, Elsevier Procedia Computer Science 171 (2020), pp. 568-575.
DOI: https://doi.org/10.1016/j.procs.2020.04.061

List of Other Related Publications

1. Sathwik Reddy Majji, **Avinash Chalumuri**, Raghavendra Kune, and B. S. Manoj, "Quantum Processing in Fusion of SAR and Optical Images for Deep Learning: A

Data-Centric Approach," in IEEE Access, vol. 10, pp. 73743-73757, July 2022.
DOI: https://doi.org/10.1109/ACCESS.2022.3189474

2. Sathwik Reddy Majji, **Avinash Chalumuri**, Raghavendra Kune, and B. S. Manoj. “Data Acquisition and Utilization of Quantum Processed SAR and Optical Images for Scene Classification", in IEEE TechRxiv Preprint, May 2022.
DOI:https://doi.org/10.36227/techrxiv.19672845.v1

Index

A

accuracy,50, 52, 79, 80, 85
amplitude embedding,20, 21, 35, 51, 52, 62, 72
analytic gradient,41
analytics tools,2, 31
angle embedding,20, 21
ANN,13, 27, 32
artificial intelligence,2

B

back-propagation,27
basis embedding,20
basis states,7
basis states,5
Bell state,11
big data,1, 11, 17
binary classification,35
binary operators,7
Bloch sphere,6
BNA,42
Born rule,5

C

characteristics,1, 2, 31
classical computer,4
classical data,18
classical processing,18, 19
classification,40, 43, 47, 71, 72, 81
CNN,27, 29, 59, 67, 70
CNOT,8, 11
computer science,3
Controlled NOT,8
convolutional neural network,70

D

data augmentation,30
data-centric,88
deep learning,27, 59, 65, 66
deep neural network,61
Dirac notation,4

E

eigenvalues,9
Entanglement,5
entanglement,3, 8, 10
entropy,91
EPR pair,11
expectation value,9

G

gate parameters,22

genomics,2

Google,3

H

Hadamard gate,6, 11

Hermitian operator,9

HHL algorithm,26

Hilbert space,5

HQCNN,14, 24, 67, 70, 81

hybrid model,23, 25, 26, 36, 59, 60

HybridQC,14, 24, 60

I

IBM,3, 36, 48, 76, 83

ibm_santiago,59, 76

image,2, 71, 81

information processing,3

Iris,42

K

kernel methods,29

L

linear algebra,12, 26

literature,2

M

machine learning,2, 11, 27, 31

measurement,9, 24, 59

micro blogging,2

model-centric,88

Moore's law,3

N

network,2

NISQ,12

numerical dataset,42

O

one-hot encoding,40

OpenSARUrban,92

optical,88

P

parameterized quantum circuit,23, 40

parameters,33–35, 46, 79, 80, 83

paramterized quantum circuit,23

paramterized quantum gates,22

Pauli-Z,9

PennyLane,42, 76

postporcessing,24

preprocessing,20, 48, 61

Q

QC ANN,13, 24, 33

QDA,14, 24

QMCC,13, 24, 36

quality metrics,91

quantum bit,3

quantum circuit,8, 10, 63, 68, 74, 83

quantum computer,3, 4, 25, 31

quantum computing,3, 29, 32

quantum data,19

quantum gates,5

quantum image processing,29

quantum information theory,3

quantum loader,20

quantum machine learning,12, 17, 19, 22, 25
quantum mechanics,3
quantum neural network,30
quantum processing,18
quantum processing technique,88
quantum processing units,4
quantum simulation,72
quantum simultion,73
qubit,3
qubit encoding,20

R

relative bias,91
relative variance,91
Richard P. Feynman,3

S

SAR,87
single-qubit gates,6
softmax,41
spatial,2
spatial data,1, 27, 58, 71
spatial frequency,92
state preparation,39
storage,2
superposition,3, 4, 10
supervised learning,11, 28, 32

T

transistors,3
Two-qubit gates,7

U

UIQI,91
unitary operator,5

V

variational circuit,23, 36, 37, 50
video,2
voice,2

W

WBCD,42
WIL,42