UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA DE PRODUÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

DIEGO ANDRÉS AICHELE FIGUEROA

USE OF QUANTUM ALGORITHMS FOR CLASSIFICATION OF ROLLING BEARING DAMAGE

Recife
2023

DIEGO ANDRÉS AICHELE FIGUEROA

USE OF QUANTUM ALGORITHMS FOR CLASSIFICATION OF ROLLING BEARING DAMAGE

Master thesis presented to Programa de Pós Graduação em Engenharia de Produção of Universidade Federal de Pernambuco as part of the requirements of the master‘s degree.

Concentration Area: Operations Research.

Advisor: Prof. Dr. Márcio Jose Das Chagas Moura, DSc.

Recife

2023

Catalogação na fonte:
Bibliotecário Carlos Moura, CRB-4/1502

A288u Aichele Figueroa, Diego Andrés.

Use of quantum algorithms for classification of rolling bearing damage. / Diego Andrés Aichele Figueroa. – 2023.

82 f.: il.

Orientador: Prof. Dr. Márcio José das Chagas Moura.

Dissertação (mestrado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia de Produção. Recife, 2023.

Inclui referências e apêndices.

1. Engenharia de produção. 2. Aprendizado de máquina quântico. 3. Conjuntos de dados de rolamentos. 4. Gerenciamento de prognóstico e saúde. 5. Diagnóstico de falhas. I. Moura, Márcio José das Chagas (orientador). II. Título.

UFPE

658.5 CDD (22. ed.) BCTG/2023-247

DIEGO ANDRÉS AICHELE FIGUEROA

USE OF QUANTUM ALGORITHMS FOR CLASSIFICATION OF ROLLING BEARING DAMAGE

Master thesis presented to Programa de Pós Graduação em Engenharia de Produção of Universidade Federal de Pernambuco, Centro de Estudos e Ensaios em Risco e Modelagem Ambiental, as part of the requirements of the master's degree attainment. Concentration area: Operations Research.

Approved in: 22/08/2023.

To my wife Ivana

To my dogs Igor, Maju and Lucas

ACKNOWLEDGEMENTS

I am grateful to my family for their unconditional support and the company of every loved one during this stage.

I express my gratitude to Professor Márcio for guiding me throughout the course of the thesis, challenging me to explore quantum computing. I also appreciate his teachings in the classroom, as they enable each student to bring out their best.

I thank all the members of CEERMA for their responses and assistance during my master's program and stay in Brazil.

I extend my thanks to the Brazilian educational system for supporting all students on their learning journey.

Es impresionante que el rostro de una persona amada, el rostro de alguien con quien hemos vivido, a quien creemos conocer, tal vez el único rostro que seríamos capaces de describir, que hemos mirado durante años, desde una distancia mínima –es bello y en cierto modo terrible saber que incluso ese rostro puede liberar de pronto, imprevistamente, gestos nuevos. Gestos que nunca antes habíamos visto. Gestos que acaso nunca volveremos a ver.

(Alejandro Zambra, Formas de volver a casa)

ABSTRACT

In Production Engineering, there is an area of research on reliability, maintenance, and risks in Production Systems. In this area, the objectives include maintenance policies aimed at predicting component failures and consequently minimizing unexpected downtimes of complex systems. Reliability engineering has successfully utilized machine learning to predict and categorize equipment and machine states. In this thesis, we aim to compare different machine learning algorithms and quantum machine learning in the context of reliability engineering. Specifically, we compare 8 models created with quantum machine learning, using three different bearing datasets: Case Western Reserve University, Machinery Failure Prevention Technology, and Paderborn University. These datasets mainly consist of accelerometer vibration data from the bearings, which must be processed to fit the different quantum machine learning algorithms. We focus on fault detection in rotating equipment, using quantum computing techniques to analyze sensor data installed on the equipment. Our approach considers the healthy state, inner ring faults, and outer ring faults of the bearings, with the aim of achieving fault detection. We compare established circuit designs, such as Real Amplitudes and Quantum Convolutional Circuits, as well as hybrid models that combine quantum circuits and neural networks using the library of TensorFlow with Cirq in python. We obtain results on classical computers using analytical calculations that simulate a quantum computer without experiencing the "quantum noise" delivered by the hardware. The most effective way to input data's key features is through a ZFeatureMap circuit, which assigns a qubit to each extracted feature from the data. Quantum convolutional circuits yield better results than other parameterized circuits, while hybrid models provide higher accuracy than their counterparts that do not utilize neural networks. The achieved accuracy rates on the training dataset are around 96%, suggesting that the parameterized quantum circuits used in this master's thesis yield stable results.

Keywords: quantum machine learning; bearing datasets; prognostic and health management; fault diagnosis.

RESUMO

Na Engenharia de Produção, existe uma área de pesquisa sobre confiabilidade, manutenção e riscos em Sistemas de Produção. Nesta área, os objetivos incluem políticas de manutenção com o objetivo de prever falhas em componentes e, consequentemente, minimizar os tempos de inatividade inesperados de sistemas complexos. A engenharia tem utilizado com sucesso a aprendizagem de máquina para prever e categorizar estados de equipamentos e máquinas. Nesta dissertação, o objetivo é comparar diferentes algoritmos de aprendizagem de máquina e de aprendizado de máquina quântico no contexto da engenharia de confiabilidade. Especificamente, se comparam 8 modelos criadas com aprendizado de máquina quântico, usando três conjuntos de dados de rolamentos diferentes: Case Western Reserve University, Machinery Failure Prevention Technology e Paderborn University. Esses conjuntos de dados contêm principalmente dados de vibração de acelerômetros dos rolamentos, que devem ser processados para se adaptarem aos diferentes algoritmos de aprendizado de máquina quântico. O foco é a detecção de falhas em equipamentos rotativos, utilizando técnicas de computação quântica para analisar dados de sensores instalados no equipamento. A abordagem considera estado saudável, falhas no anel interno e falhas no anel externo dos rolamentos, com o objetivo de detecção de falhas. Foram comparados circuitos estabelecidos, como Real Amplitudes e Circuitos Quânticos Convolucionais, bem como modelos híbridos que combinam circuitos quânticos e redes neurais usando a biblioteca do TensorFlow com o Cirq em Python. Os resultados obtidos em computadores clássicos usando cálculos analíticos que simulam um computador quântico, sem experimentar o "ruído quântico" entregue pelo hardware. A melhor maneira de inserir dados das principais características é através de um circuito ZFeatureMap, que dedica um qubit para cada característica extraída dos dados. Circuitos quânticos convolucionais produzem melhores resultados em comparação com outros circuitos parametrizados, enquanto modelos híbridos fornecem maior precisão do que seus equivalentes que não utilizam redes neurais. As taxas de precisão alcançadas no conjunto de dados de treinamento são em torno de 96%, sugerindo que os circuitos quânticos parametrizados usados nesta tese de mestrado estão fornecendo resultados estáveis.

Palavras-chave: aprendizado de máquina quântico; conjuntos de dados de rolamentos; gerenciamento de prognóstico e saúde; diagnóstico de falhas.

FIGURE INDEX

TABLE INDEX

INDEX

1 INTRODUCTION

Quantum computing represents a new paradigm in computing that uses the fundamental principles of quantum mechanics to perform calculations. With quantum computing, it is possible to solve different optimization, machine learning, finance or chemistry-based problems that are difficult to solve with classical computing. Operations on quantum computers are performed on quantum bits (or qubits) that represent the state-of-the-art in quantum computing (NIELSEN and CHUANG, 2000). However, current quantum devices do not have the necessary qubits and are not fault-tolerant enough to achieve these goals (PERAL GARCÍA, 2022). For this same reason, quantum computing has shown to be an area with important advances recently, where companies like IBM have developed quantum computers in 2021 with 127 qubits such as the Eagle, and with future plans to build a quantum computer with 433 qubits in 2022 (Osprey) and a quantum computer with 1,121 qubits by the end of 2023 (Condor) (IBM, 2021).

Different tools have also been built to perform simulations of quantum computers. In the case of IBM, Qiskit (QISKIT, 2020) is used with an integration with Pytorch, and for Google, Cirq is used combined to Tensorflow Quantum to perform quantum machine learning (CIRQ, 2020). This growth in technological level has occurred in recent years and has shown the theoretical advantage that distinguishes quantum from classical computing, such as improved speed and accuracy.

Prognostics and Health Management (PHM) is a cutting-edge integrated technology, which takes knowledge, information and data of system performance, control, operation and maintenance as input to detect the initiation of anomalies, isolate/diagnose the occurrence of failures, predict the health state of the system in the future and estimate its remaining useful life to dynamically support the maintenance decisions (HU and MIAO, 2022). Recently, machine learning, especially deep learning methods, have become popular in the PHM field and many review papers have considered how deep learning can enhance the PHM general performance (HU and MIAO, 2022). The launch of quantum computing gives rise to interesting areas of research that seek to leverage the principles of quantum mechanics to improve machine learning or vice versa. The aim is to boost classical Machine Learning (ML) algorithms by outsourcing difficult calculations to a quantum computer or optimizing quantum algorithms using classical

ML architectures – both fall under the diverse umbrella of quantum machine learning (QML) (QISKIT, 2021).

That is why the aim of this master thesis is to compare some of the different ML and QML algorithms that have emerged to date. In other words, the aim is to make a comparison at the simulation level with the different neural networks that can be made with QML. The above-mentioned datasets mainly contain vibration data (collected from accelerometers) of the bearings. Therefore, it will be necessary to preprocess the data to be able to adapt them to the different QML algorithms.

In this master thesis, we present our research on fault detection in rotary equipment, specifically bearings. A bearing facilitates the low-friction relative motion between two machine elements while simultaneously transmitting the forces between them as well as guiding and positioning them relative to each other. The operating behavior of the bearings has a significant effect on the operating characteristics of the overall machine, and then the suitability of the selected bearings for the planned application case must be investigated carefully (BRECHER, 2014). To perform this comparison in the field of reliability engineering, different sets of bearing data will be used, namely:

- Case Western Reserve University bearing dataset (CWRU)(SMITH and RANDALL, 2015).
- Machinery Failure Prevention Technology bearing dataset (MFPT) (BECHHOEFER, 2018).
- Paderborn University bearing dataset (PU) (LESSMEIER, 2020).

We focus on three different states of the bearings, namely healthy, inner ring fault, and outer ring fault. Our approach involves using quantum computing circuits, specifically the Real Amplitudes circuit and Quantum Convolutional Circuit (QCNN), to analyze data obtained from sensors installed in the bearings. Our methodology has the potential to explore new fault detection models in rotary equipment through the implementation and evaluation of proof-of-concept tests, which can assess their efficiency and accuracy in this specific context.

We examine the effectiveness of established circuit designs such as Real Amplitudes comprising alternating Y rotations and CX entanglements. The rotation y (Ry) gate is a single-qubit rotation through angle θ (radians) around the y-axis. The CX gate refers to a controlled-X gate. Real Amplitudes prepares quantum states comprising only Real Amplitudes, whereas the complex part is always 0 (SURYOTRISONGKO and MUSASHI, 2022) and QCNN model can be a solution in the field of physical and chemical classification not solved simply, and in addition to the existing Convolutional Neural Network (CNN) model, it can be a more effective and efficient learning model method (OH, 2020), as well as the effectiveness of hybrid models that combine quantum circuits and neural networks. In comparing the present work to that of PHM of Rotating Machinery via Quantum Machine Learning (SOUTO, MENDES, *et al.*, 2023), who also uses set of rotating equipment data for fault detection (CWRU), they develop their own circuit designs for hybrid models, rather than using established designs. Additionally, they only test hybrid models that combine quantum circuits and neural networks, while we test both exclusively quantum circuits and hybrid models. While we both use the same dataset of rotating equipment data, these methodological differences are important to consider when comparing the results of our studies.

Finally, the research encompasses an economic aspect, wherein studying new methods for damage detection can lead to the discovery of more ways to prevent accidents that may involve harm to the environment, equipment costs, and accidents involving individuals. This requires investment in more efficient maintenance and prevention methods. On the social front, the goal is to automate data analysis so that operators can focus on their work without losing time in the detection of damage or leaks. It is important to act ethically in this aspect since the idea is to support the operator rather than replace them, as decisions during accidents must be made by humans. Lastly, the environmental aspect depends on the factory's location and the area in which it operates. A equipment breakdown can result in leaks that can harm both the environment and the area surrounding the factory.

1.1 Objectives

1.1.1 General objectives

The general objective of this study is to compare existing quantum circuits with and without the assistance of neural networks, utilizing PHM datasets.

1.1.2 Specific objectives

According to the general objective, the specific targets of this work are:

- To analyze the CWRU, MFPT and PU bearing datasets that can be used for classification purposes.
- To analyze the Real Amplitudes Circuit and QCNN used for vibration data.
- To train the resulting data with the selected algorithms.
- To evaluate and visualize the results for relevant comparison.
- To compare the performance of quantum circuits and hybrid circuits.

1.2 Dissertation structure

The structure of the current investigation is as follows:

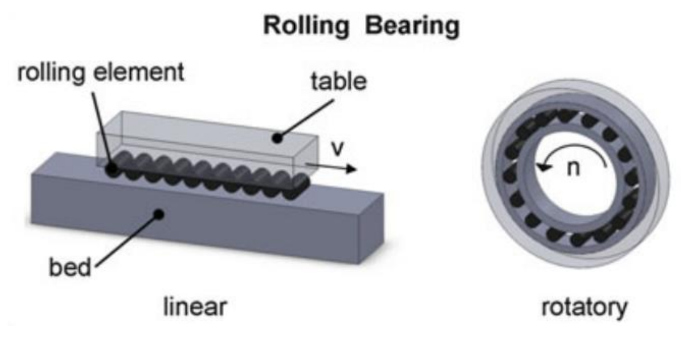
- Chapter 2 presents the theoretical framework, into topics such as Bearing datasets, Vibration Signals, Quantum Computing and existing circuits, and QML.
- Chapter 3 presents the methodological aspects of the study, providing explanations about the databases used for the analyses, the proposed QML models, and the training method.
- Chapter 4 presents the results derived from the QML models.
- Chapter 5 presents some concluding remarks of the study.

2 BACKGROUND

Bearings enable both rotational and translational motion between two machine parts. In addition to transferring radial and axial forces (in rotary bearings) and forces and moments perpendicular to the direction of motion (in linear bearings), they establish the relative positions of the machine elements supported by the bearings. Bearings are crucial components in all production machines and often play a significant role in their performance. For instance, in a machine tool, various types of bearings are used for different purposes: linear guiding is used for feed axes, rotary bearings are used for rotary axes, threaded spindle drives for linear axes, and bearings for main spindle shafts system (BRECHER, 2014). The bearing system is the component with the greatest influence on the lifetime of a spindle. Most commonly the motor is arranged between the two bearing systems (ABELE, ALTINTAS and BRECHER, 2010). The load capacity, stiffness, and speed capabilities of the bearings in a machine tool impact the cutting performance, and the stiffness also affects the achievable machining precision. Friction in bearings adds to the machine's energy consumption, and the lifespan of the bearings can affect the downtimes of the entire system (BRECHER, 2014).

Roller bearings are the most common type of bearing used. These use rolling elements placed between the machine parts being supported to create a rolling motion. Ideally, this results in low rolling friction, but in reality, there may be sliding-rolling friction due to factors such as the bearing design or operating conditions. Roller bearings (Figure 1) are widely used because of their relatively low cost, ease of use, standardization, and the range of properties that can be achieved with different designs. The main types of roller bearings used in machine tool main spindles are angular contact ball bearings, tapered roller bearings, and cylindrical roller bearings. Angular contact ball bearings, which have a contact angle of between 15 and 25 degrees, can carry radial and axial loads and have the highest speed capabilities of all roller bearings (BRECHER, 2014).

Figure 1 - Rolling bearing



Source: Bearing (BRECHER, 2014).

1.1 Bearing data sets

The different data sets to be here used are CWRU, MFPT and PU. The common information of these three datasets is that they are experiments conducted on bearings. The data includes information about their healthy state, inner ring failure, and outer ring failure, which are clearly distinguishable as they are captured under different scenarios. Furthermore, all three datasets are based on accelerometers, with the only difference being the frequency at which their samples are collected:

- CWRU bearing dataset (SMITH and RANDALL, 2015): The data were collected for normal bearings, single drive end point and fan end defects. The data were gathered at 12,000 samples/second and 48,000 samples/second for drive end bearing experiments.
- MFPT bearing dataset (BECHHOEFER, 2018): A set of bearing failure data is provided to facilitate research in bearing analysis. The data set includes data from a bearing test rig (nominal bearing data, outer race failure at various loads and inner race failure at various loads) and three real-world failures.
- PU bearing dataset (LESSMEIER, 2020): This data set from the Chair of Drive and Powertrain Technology, University of Paderborn, Germany is provided to allow and encourage collaboration in the field of bearing condition monitoring. The rotational speed of the drive system, the radial force on the test bearing and the load torque on the drive train are the main operating parameters.

2.1 Quantum computing

The bit is the fundamental concept of classical computing and classical information. Quantum computing and quantum information are built upon an analogous concept. In this section, the properties of single and multiple qubits will be introduced, comparing and contrasting their properties with those of classical bits (NIELSEN and CHUANG, 2000).

Quantum computing and quantum information are a wonderful theoretical discovery, but their central concepts, such as superpositions and entanglement, go against the intuition we get from the everyday world around us (NIELSEN and CHUANG, 2000). In spite of this, entanglement is a major resource for quantum communication and computation (ASHER PERES, 2003) which are linked qubits together in such a way that their states cannot be described independently. When qubits are entangled, the state of one qubit is intimately connected to the state of the other qubit, even if the two qubits are separated by a large distance. This correlation is a result of the strong quantum mechanical interactions between the qubits (NIELSEN and CHUANG, 2000).

Likewise, the quantum state of Schrödinger's legendary cat, doomed to be killed by an automatic device triggered by the decay of a radioactive atom, evolves into a superposition of "live" and "dead" states. This is a manifestly absurd situation for a real cat. The only meaning of such a quantum state is that of a mathematical tool for statistical predictions on the fates of numerous cats subjected to the same cruel experiment (ASHER PERES, 2003).

The quantum bit (qubit) is the basic information processing unit in quantum computing. The state of a qubit can be represented by a unit vector in a 2-dimensional Hilbert space spanned by the computational basis states $|0\rangle$ and $|1\rangle$ of the qubit. Specifically, the state $||\psi\rangle$ of a qubit can be expressed by $|\psi\rangle = \mathrm{a}|0\rangle + \mathrm{b}|1\rangle$where a, b ∈ C (C: Complex number) and $|a|^2 + |b|^2 = 1$. Without ambiguity, $|\psi\rangle$ can be written in the vector form $|\psi\rangle = (a, b)^T$ which is known as the state vector. For a system with n qubits, its state space is the tensor product of all its qubits and is a 2^n-dimensional Hilbert space. A state vector in this space is a vector with 2^n elements $|\psi\rangle = (a_0, \dots, a_{2^n-1})^T$.

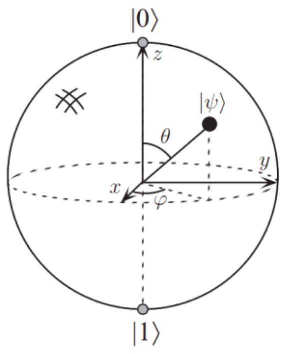
The sphere in Figure 2 is often called the Bloch sphere and provides a useful way to visualize the state of a single qubit and generally serves as an excellent test bed for ideas about quantum computing and quantum information. On the Bloch sphere, the states $|0\rangle$ and $|1\rangle$ form an orthonormal basis, it is part of the bra-ket notation, introduced by Dirac (QISKIT, 2020).

We can represent any 3D vector as a combination of these two states as shown in equation (1):

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

where the values of θ and ϕ form a point on the unitary three-dimensional sphere. At this point, the state of $|\psi\rangle$ has an interpretation of the superposition of states, but measuring a qubit will only give a value of 0 or 1. So the measurement of the state $|\psi\rangle$ will collapse from its superposition of $|0\rangle$ and $|1\rangle$ to the specific state consistent with the measurement result.

Figure 2 - Bloch sphere; representation of qubit



Source: (NIELSEN and CHUANG, 2000).

A Bloch sphere has 6 commonly known states that are assigned to the following symbols: $|0\rangle$, $|1\rangle$, $|+\rangle$, $|-\rangle$, $|+\mathrm{i}\rangle$ *and* $|-\mathrm{i}\rangle$ (Equations (2)-(7)):

$$|0\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{2}$$

$$|1\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{3}$$

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{4}$$

$$|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}\begin{bmatrix} -1 \\ 1 \end{bmatrix} \tag{5}$$

$$|+\mathrm{i}\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle) \tag{6}$$

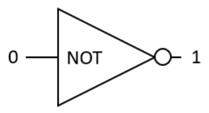
$$|-\mathrm{i}\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle) \tag{7}$$

2.2 Single Qubit Gates

Classical and quantum computation have many similarities in terms of how they are used to process and manipulate information. In classical computation, circuits consisting of wires and logic gates are used. The wires are used to carry information around the circuit, while the logic gates perform manipulations of the information, converting it from one form to another.

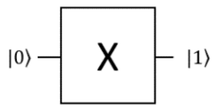
In the case of quantum computation, it is very similar. Wires are represented by qubits that store information, while gates are used to execute computational operations. Take, for example, classical single bit logic gates. The only non-trivial member of this class is the NOT gate (Figure 3), whose operation is defined by its truth table, where $0 \rightarrow 1$ and $1 \rightarrow 0$. In quantum computation, the Pauli X gate (Figure 4, in the next section we are going define this gate with more details) serves a similar function, having an analogous truth table and being used to invert the state of a qubit. However, unlike the classical NOT gate, the Pauli X gate also has a geometric interpretation in terms of the state space of a qubit (NIELSEN and CHUANG, 2000).

Figure 3 - Classical NOT gate.



Source: The author (2023)

Figure 4 - X Pauli Gate



Source: The author (2023)

In this section, we will talk about the most well-known and commonly used qubit gates, which are typically represented by a letter.

2.2.1 X Gate

The X gate, also known as the Pauli-X gate, is equivalent to flipping the state of a qubit, with $|0\rangle$ becoming $|1\rangle$ and vice versa ($|1\rangle \rightarrow |0\rangle$). It is represented by the matrix (NIELSEN and CHUANG, 2000):

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{8}$$

In the Dirac notation, if the quantum state of a qubit is $\alpha|0\rangle + \beta|1\rangle$, it is written with $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$ in a vector notation. Then, the corresponding output from the quantum gate is $X\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}$ (NIELSEN and CHUANG, 2000).

2.2.2 Y Gate

Y gate, also known as the Pauli-Y Gate, is the equivalent of $|0\rangle \rightarrow i|1\rangle$ and $|1\rangle \rightarrow i|0\rangle$. It is represented by the matrix (NIELSEN and CHUANG, 2000):

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{9}$$

So, if the quantum state of a qubit is $\alpha|0\rangle + \beta|1\rangle$, it is written with $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$ in a vector notation. Then, the corresponding output from the quantum gate is $Y\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} i\beta \\ i\alpha \end{bmatrix}$ (NIELSEN and CHUANG, 2000).

2.2.3 Z Gate

Z gate, also known as the Pauli-Z Gate, is the equivalent of $|0\rangle \rightarrow |0\rangle$ and $|1\rangle \rightarrow -|1\rangle$. It is represented by the matrix (NIELSEN and CHUANG, 2000):

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{10}$$

If the quantum state of a qubits is $\alpha|0\rangle + \beta|1\rangle$, then it is written with $\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$ in a vector notation. Thus, the corresponding output from the quantum gate is $Z\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha \\ -\beta \end{bmatrix}$. It is possible to use the Z gate in conjunction with measurement, but this will depend on the context and the specific purpose of the measurement (NIELSEN and CHUANG, 2000).

2.2.4 H Gate

H-gate is the equivalent of $H|0\rangle \rightarrow |+\rangle$ and $H|1\rangle \rightarrow |-\rangle$ gate (Hadamard Gate):

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{11}$$

This can be thought of as a rotation around the Bloch vector [1,0,1] (the line between the x and z-axis), or as transforming the state of the qubit between X and Z bases (NIELSEN and CHUANG, 2000).

2.2.5 U Gate

The U gate is a single qubit quantum gate used to apply a unitary transformation to a qubit. This gate is very useful in the construction of quantum algorithms, as it allows a wide range of transformations to be applied to a qubit. Some of the most common operations that can be performed with the U gate are rotating a qubit around the Z axis, rotating around the X axis, and applying a phase. Overall, the U gate is a very versatile and essential tool in the implementation of quantum algorithms.

This is a diagonal gate, where:

$$U1(\lambda) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{bmatrix} \tag{12}$$

As additional information, when $\lambda = \pi$ in $U1$ it's transform to Z gate. As shown in the next equation:

$$U1(\pi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = Z \tag{13}$$

2.3 Multiple qubits

Multiple qubits are a fundamental concept in quantum computing. They refer to the use of multiple quantum bits, or qubits, to store and process quantum information. However, unlike classical bits, qubits can exist in a superposition of states, meaning they can represent multiple values simultaneously. This property allows qubits to store and process information in ways that are not possible with classical bits (QISKIT, 2020).

Using multiple qubits allows for the creation of more complex quantum systems and enables the implementation of more powerful quantum algorithms. For example, multiple qubits can be entangled, which means they are correlated in a way that allows them to affect each other's state even when they are separated by large distances (NIELSEN and CHUANG, 2000). This property is a key aspect of quantum computing and enables the creation of quantum computers that are capable of solving certain problems much faster than classical computers.

Overall, multiple qubits are an essential part of quantum computing and hold great potential for revolutionizing many fields, including cryptography, drug discovery, and machine learning (ASHER PERES, 2003).

In general, multiple qubit gates are quantum operations involving more than one qubit and used to perform logical operations or unitary transformations on multiple qubit systems. Some of the most well-known multiple qubit gates include the CNOT gate, the Toffoli gate, and the SWAP gate.

2.3.1 CNOT gate

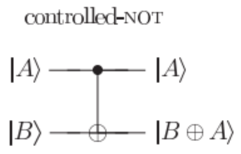
The CNOT (Control-NOT) gate is a two-qubit quantum gate used to perform logical operations on multi-qubit systems. It is called this because one of the qubits acts as a control and the other as the target bit to which the NOT operation is applied. The CNOT gate has the following instructions: if the state of the control qubit is $|0\rangle$, then the state of the target qubit is not modified; if the state of the control qubit is $|1\rangle$, then the state of the target qubit is inverted.

And acting on our 4D-state vector, it has the next matrix representation:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{14}$$

The CNOT gate, shown in Figure 5, is very important in quantum computing because it is one of the most basic and versatile gates that can be used to build more complex quantum algorithms. In addition, the CNOT gate is the basis for more complex gates, such as the Toffoli gate. In general, the CNOT gate is essential for the implementation of quantum algorithms, and it is a fundamental tool in quantum computing (QISKIT, 2020).

Figure 5 - CNOT circuit representation.

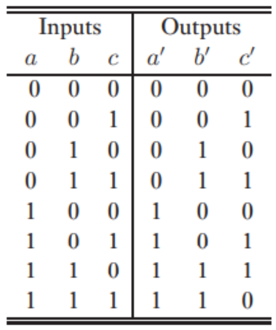


Source: Quantum Computing and quantum information, Page 21 (NIELSEN and CHUANG, 2000)

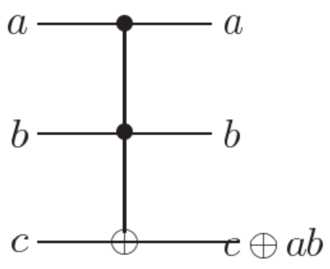
2.3.2 Toffoli gate

The Toffoli gate is a reversible gate that takes three qubits as input and has a truth table similar to that of the classical NOT gate. In general, the Toffoli gate is used to invert the state of a qubit based on the state of two control qubits. This allows the Toffoli gate to perform complex logical operations on multiple-qubit systems, making it very useful in the construction of quantum algorithms (Nielsen and Chuang, 2000).

Figure 6 - Truth table for the Toffoli gate, and its circuit representation.



Inputs			Outputs		
a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0



Source: Quantum Computing and quantum information, Page 29 (Nielsen & Chuang, 2000)

The matrix representation of Toffoli of three qubits (same circuits as shown in Figure 6):

$$Toffoli = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{15}$$

2.3.3 SWAP gate

The SWAP gate is a multi-qubit quantum gate that is used to swap the states of two qubits. This gate is useful in the construction of quantum algorithms and can be used to manipulate and control the state of two qubits precisely. The SWAP gate action can be described by the following matrix:

$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{16}$$

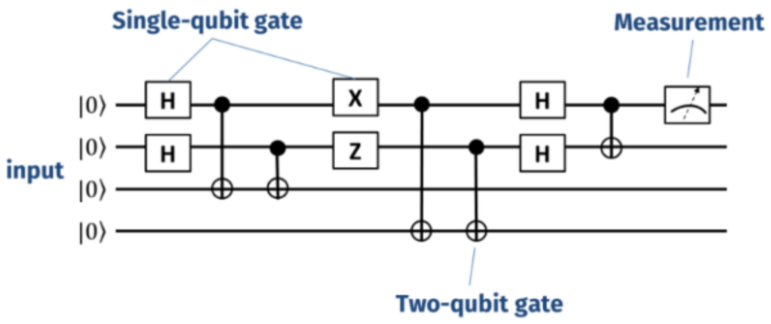
The SWAP gate is applied to two qubits, and its action is to swap the states of the two qubits. For example, if there are two qubits in the states $|0\rangle$ and $|1\rangle$, the SWAP gate would swap them so that they are in the states $|1\rangle$ and $|0\rangle$ respectively. The SWAP gate is reversible, which means it can be applied again to return to the original states (QISKIT, 2020).

2.4 Quantum circuits

The quantum circuit is composed of wires and gates, and it should be read from left to right. Each line in the circuit represents a wire in the quantum circuit, which does not necessarily correspond to a physical wire; it may instead represent the passage of time, or perhaps a physical particle such as a photon, moving from one place to another through space. It is conventional to assume that the input state to the circuit is a computational basis state, typically the state formed by $|0\rangle$ (NIELSEN and CHUANG, 2000).

In Figure 7, we can see the composition of the circuit with its inputs on the left side, and a single-qubit gate that only affects one wire, as well as the possibility of two-qubit gates that affect more than one wire.

Figure 7 - Example of a quantum circuit



Source: (SOUTO, MENDES, *et al.*, 2023)

In quantum computing, a variational quantum circuit (VQC) is a type of algorithm that is used to find the optimal value of a function over a set of variables. These algorithms are based on optimization theory and are commonly used to find approximate solutions to problems that are difficult to solve exactly (CIRQ, 2020).

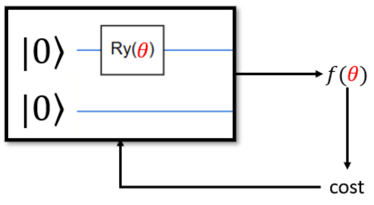
Variational quantum algorithms that are used for quantum machine learning rely on the ability to automatically differentiate parametrized quantum circuits with respect to underlying parameters (KYRIIENKO and ELFVING, 2021). Variational or parameterized quantum circuits are quantum algorithms that depend on free parameters. Like standard quantum circuits, they consist of three ingredients:

- Preparation of a fixed initial state (e.g., the vacuum state or the zero state).
- A quantum circuit $U(\theta)$ parameterized by a set of free parameters θ.
- Measurement of an observable $\hat{B}$ at the output. This observable may be made up from local observables for each wire in the circuit, or just a subset of wires.

A VQC consists of a series of quantum gates and qubits that are used to implement a specific mathematical function. The function is evaluated for different values of the variables, and an optimization method is used to find the optimal value of the function. VQC is often used for solving quantum optimization problems, such as finding the optimal solution to a linear programming problem or minimizing a cost function (PENNYLANE, 2022).

VQC is trained using a classical optimization algorithm that sends requests to the quantum device. The optimization process typically involves iteratively improving the values of the parameters θ at each step (PENNYLANE, 2022).

Figure 8 - Simplified diagram of a Variational Quantum Circuit



Source: Adapted from PennyLane (2022)

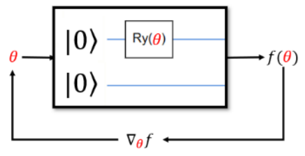
The output of a VQC is the expectation value of an observable measurement, which can be formally written as a parameterized "quantum function" $f(\theta)$ in the tunable parameters $\theta = \theta_1, \theta_2, \ldots$ As with any other function, partial derivatives of $f(\theta)$ can be defined with respect to its parameters.

2.4.1 Quantum gradient

A quantum gradient is the vector of partial derivatives of a quantum function $f(\theta)$:

$$f(\theta) = \begin{pmatrix} \delta_{\theta_1} f \\ \delta_{\theta_2} f \\ \delta_{\theta_3} f \end{pmatrix} \tag{17}$$

Figure 9 - Simplified diagram of a Quantum Gradient Descent



Source: Adapted from PennyLane (2022)

It turns out that the gradient of a quantum function f(θ) can in many cases be expressed as a linear combination of other quantum functions via parameter-shift rules. This means that quantum gradients can be calculated with quantum computers, enabling the use of quantum computing in gradient-based optimization such as gradient descent, which is widely used in machine learning (PENNYLANE, 2022).

The parameter-shift rule is an approach to measuring gradients of quantum circuits with respect to their parameters, which does not require ancilla qubits or controlled operations. The parameter-shift rule is a promising approach to evaluating gradients of parameterized quantum circuits on quantum hardware (CROOKS, 2019). The advantage (BANCHI and CROOKS, 2021) of this gradient is that parameter shift is exact, the formula for the gradient doesn't involve any approximations (PENNYLANE, 2022) (BANCHI and CROOKS, 2021). The shift does not have to be small or infinitesimal like in finite differences. Whereas finite differences can be applied to any function f(θ), the parameter-shift rule can only be used to determine $\frac{\mathrm{df}(\theta)}{\mathrm{d}\theta}$ if the function satisfies certain constraints. Suppose the next function f(θ):

$$\mathrm{f}(\theta) = \langle \phi | U_G{}^{t}(\theta)\, A\, U_G(\theta) | \phi \rangle, \tag{18}$$

Where $U_G(\theta)$ is the parameterized gate and ϕ is the initial state of the observable operator A. The parameter-shift rule states that if the generator of the gate G has only two unique eigenvalues, e_0 and e_1, then

$$\frac{d}{d\theta}f(\theta) = r\left[f\left(\theta + \frac{\pi}{4r}\right) - f\left(\theta - \frac{\pi}{4r}\right)\right], \tag{19}$$

Where the shift constant is $r = \frac{\alpha}{2}(e_1 - e_0)$. Compared to other approaches for evaluating circuit gradients, the parameter-shift rule has the advantage that it requires the performance of two circuits each of which being the same number of gates as the original circuit and does not require ancilla qubits (JORDAN, 2004).

2.4.2 Variational circuit

An ansatz is a proposed starting point for addressing a mathematical or physical problem, typically in the form of an equation, theorem, or value. It serves as a foundation for further investigation and potentially finding a solution to the problem at hand. In the context of quantum computing, an ansatz is a quantum circuit used to represent or model a quantum function (PENNYLANE, 2022).

2.4.2.1 Data encoding circuits (Data Embedding)

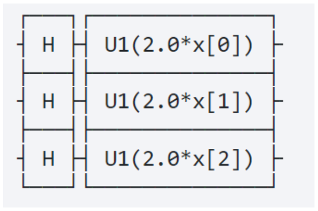
A data encoding circuit is a quantum circuit that is used to encode classical data into the quantum state of a quantum computer. This process, known as quantum state preparation, is an important step in many quantum algorithms and protocols (PENNYLANE, 2022).

Quantum feature maps provide a way to harness the computational power of quantum mechanics in machine learning problems. In machine learning, pattern recognition and image processing, a feature map starts from an initial set of measured data and builds derived values (also known as features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations (QISKIT, 2020). This idea leads to a simple quantum machine learning algorithm that makes no additional assumptions on data access and has rigorous and robust performance guarantees (LIU, 2020).

In the investigation conducted by SURYOTRISONGKO and MUSASHI (2022), four feature mapping circuits were utilized (PauliFeatureMap, RawFeatureVector, ZFeatureMap, and ZZFeatureMap). In terms of performance, the order from best to worst was RawFeatureVector, ZFeatureMap, ZZFeatureMap, and PauliFeatureMap. However, in this thesis, a comparison with minimal changes in the number of qubits is desired. Therefore, RawFeatureVector cannot be used since this circuit acts as parameterized initialization for state vectors with "feature_dimension" dimensions, thus requiring "log2(feature_dimension)" qubits (QISKIT, 2020). In other words, the number of qubits is not directly proportional to the number of inputs provided. To make a fairer comparison in terms of the number of qubits, ZFeatureMap and ZZFeatureMap will be used. It is worth mentioning that, in the same investigation, Angle Encoding and IQP embedding are also utilized, but only for hybrid circuits. Additionally, there is a desire to compare the hybrid circuit against the parameterized quantum circuit; therefore, the use of ZFeatureMap and ZZFeatureMap will be maintained.

- ZFeatureMap: This circuit consists of a Hadamard gate for each qubit and then the parameter is multiplied and added within a U1 gate, as shown in Figure 10; the resulting circuit contains no interactions between features of the encoded data, and therefore no entanglement (QISKIT, 2020).

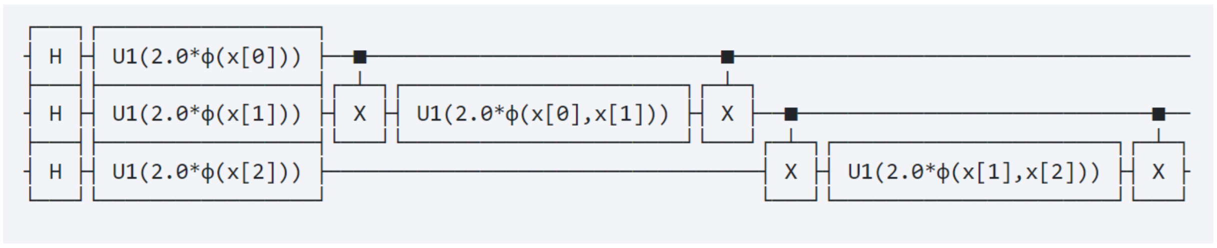
Figure 10 - ZFeatureMap with 3 qubits.



Source: (QISKIT, 2020)

- ZZFeatureMap: This circuit starts the same as a ZFeatureMap, but then CNOT gates are linked either in a Circular, Linear, or Inversely Linear manner, followed by U1 gates containing the function $\Phi(x, y) = (\pi - x)(\pi - y)$, where x and y are related to the qubits of each input data, as shown in Figure 11. The ZZFeatureMap allows interactions in the data to be encoded in the feature map according to the connectivity graph and the classical data map. ZZFeatureMap with default parameters is equivalent to the feature map described in Figure 11 and can take additional inputs (QISKIT, 2020).

Figure 11 - ZZFeatureMap with 3 qubits



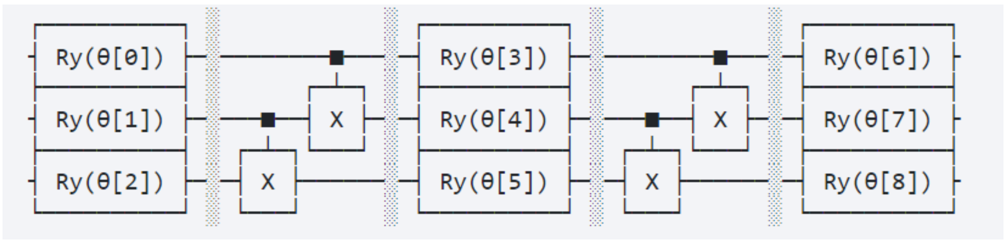
Source: (QISKIT, 2020)

2.4.2.2 Data processing circuits

In this type of circuits, free parameters are used that vary when training the quantum circuit. In this thesis we use two PQC the Real Amplitudes and Quantum Convolutional Circuit.

The Real Amplitudes circuit is a heuristic trial wave function used as Ansatz in chemistry applications or classification circuits in machine learning. The circuit consists of alternating layers of Y rotations and CX entanglements (QISKIT, 2020).

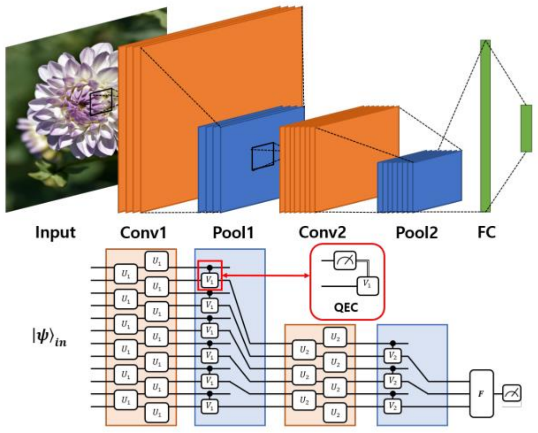
Figure 12 - Real Amplitudes Circuit with 2 repetitions and 3 qubits.



Source: (QISKIT, 2020)

Quantum Convolutional Circuit is a type of quantum circuit that is inspired by the convolutional layers in classical deep neural networks and is used to process input data in signal or image classification problems, as shown in Figure 13.

Figure 13 - Quantum Convolutional Circuits Concept



Source: (OH, 2020)

The concept proceeds as follows (OH, 2020):

1) The convolution circuit finds the hidden state by applying multiple qubit gates between adjacent qubits.
2) The pooling circuit reduces the size of the quantum system by observing the fraction of qubits or applying 2-qubit gates such as CNOT gates.
3) Repeat the convolution circuit and pooling circuit defined in 1)-2).
4) When the size of the system is sufficiently small, the fully connected circuit predicts the classification results.

2.5 Literature Review

Quantum computing promises to be the next disruptive technology, with numerous possible applications and implications for organizations and markets. A recently published report by McKinsey estimates the global market value of quantum computing to be at USD 1 trillion by 2035, mainly in the financial, chemical, pharmaceutical, and automotive sectors (RIETSCHE, DREMEL, *et al.*, 2022).

Many industrial systems demand equipment with high levels of reliability. Over the years, companies and academia have been developing mathematical methods and advancing engineering techniques to assist in the maintenance of active and reliable systems (MENDES, LINS, *et al.*, 2022). Reliability engineering studies are often conducted to minimize the probability of failure in complex systems, for instance, the redundancy allocation problem, The focus is to assign several parallel components to achieve the best possible overall system reliability within budget constraints (MENDES, LINS, *et al.*, 2022).

Despite the current limitations of hardware, quantum computing has been explored to improve the speed and efficiency of ML models. This master thesis focuses on the application of Quantum Machine Learning to diagnose rolling bearings which are essential components in rotating machinery, based on vibration signals (SOUTO, MENDES, *et al.*, 2023).

In the investigation conducted by SURYOTRISONGKO and MUSASHI (2022), it shows how different 4 Feature Maps and 4 different parameterized quantum circuits are used for different datasets, but these configurations have not been tested for damage prognosis.

In the investigation conducted by SAN MARTÍN SILVA and LÓPEZ DROGUETT (2022), they use a Hybrid quantum circuit to perform the classification of damage states for the MFPT dataset, so, it is a study dedicated to PHM. They do not use entanglement in their circuits and only focus on studying a single circuit. It utilizes Angle Encoding, and the quantum circuit only maintains three parameters to train.

It is crucial to emphasize the importance of being able to investigate and analyze quantum circuits in the field of PHM, as addressed in previous studies conducted by SOUTO and MENDES (2023) and SAN MARTÍN SILVA and LÓPEZ DROGUETT (2022). These studies provide a solid framework for understanding how quantum circuits can play a fundamental role in PHM. Furthermore, it is relevant to closely examine the effectiveness of quantum circuits both with the support of neural networks and in their standalone implementation, as investigated in the work carried out by SURYOTRISONGKO and MUSASHI (2022), however, it is important to note that doesn’t focus on a PHM dataset. Therefore, there is a pressing need to conduct a comparative study in the context of PHM, which allows for analyzing and comparing the performance of PQC with and without the assistance of quantum circuits.

This study, presented in this thesis aims to fill this knowledge gap and provide a more comprehensive and in-depth understanding of the potential applications of quantum circuits in the field of PHM.

3 METHODOLOGY

This study follows a quantitative methodology, where real datasets and models are used for analysis and comparison. The quantitative approach allows for obtaining objective and measurable results through the application of specific metrics, which are used to evaluate the performance of the models, as well as to make comparisons between them. Through this process, the aim is to identify patterns, relationships, and trends in the data, and to determine which models are more effective in terms of accuracy and efficiency. The results obtained from this quantitative study are essential for generating knowledge and supporting robust conclusions in the research area under investigation.

3.1 Datasets

In this thesis, a set of famous bearing data in the reliability and maintenance area is used. A bearing consists of 4 parts: inner ring, outer ring, a rolling element (balls, rollers, or cones, depending on the shape of the bearing), and a cage that is responsible for keeping the rolling bodies separated.

The 3 sets of data work with vibration, measured with an accelerometer. What is special about these 3 sets of data is that there is information about bearing with 2 types of damage: damage to the inner ring and damage to the outer ring. And information from a bearing working without damage.

Standard vibration-based metrics like mean, variance, RMS, kurtosis, as well as higher order statistics, are frequently employed for machinery diagnostics (Souto, Mendes, Lins, Moura, & López, 2023). This thesis employs 5 features that are directly processed from the raw vibration signal data, namely peak-to-peak, average, variance, maximum amplitude, and RMS.

A feature extraction is related to dimensionality reduction; it involves reducing the resources required to describe a large data set. When performing analysis of complex data, one of the major problems stems from the number of variables involved. Analysis with many variables generally requires a considerable amount of memory and computation power and may even cause a classification algorithm to overfit training samples and generalize poorly to new samples

(QISKIT, 2020). When the input data to an algorithm is too large to be processed and is suspected to be redundant (for example, the same measurement is provided in both pounds and kilograms), then it can be transformed into a reduced set of features, named a feature vector. The process of determining a subset of the initial features is called feature selection. The selected features are expected to contain the relevant information from the input data, so the desired task can be performed using the reduced representation instead of the complete initial data (QISKIT, 2020).

In summary, the treatment of each set of data was performed similarly. In fact, the data is downloaded directly from their official sources, the vibration data is obtained and classified according to its damage, after each vibration data a dimensionality reduction, with the mean, variance, maximum amplitude, root mean square (RMS) and peak-to-peak value of each signal is obtained.

A repository was created on github (https://github.com/diegoaichele/MasterThesisDiegoA), which automates this process, allowing you to choose the length of each signal to be analyzed and the overlap that the signals may have with each other.

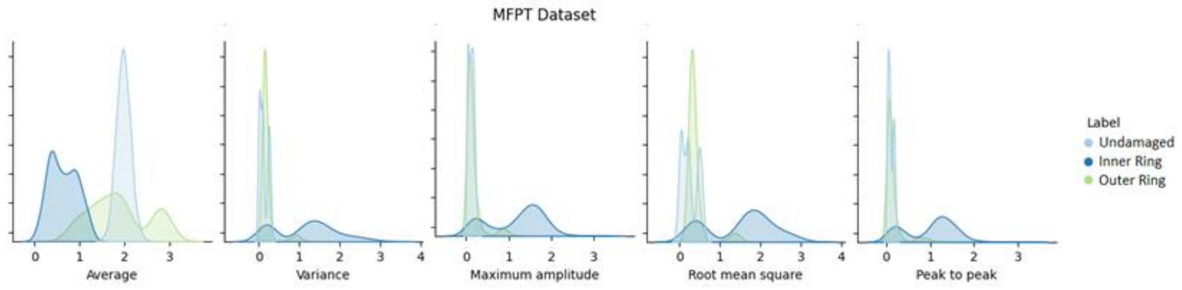
3.1.1 MFPT

The MFPT database has 20 .mat files, of which 3 are undamaged bearings, 10 are bearings with damage to the outer ring and 7 are bearings with damage to the inner ring (Bechhoefer, 2018).

The data is plotted visually, for each corresponding dimension, as shown in Figure 30 in Appendix A. For this case study we join them and consider three classes: Undamaged (U), inner ring fault (IR) and outer ring fault (OR).

In the following graph in Figure 14, when we analyze the density distribution of each extracted feature, we can observe how the average of a time series can provide more information than the others in this dataset, for example, it can be stated that the Inner Ring is easily comparable as all its average values are around 1.5 or below, while for Undamaged, they are found to be above 1.5. An undamaged bearing exhibits tightly clustered values for each feature, suggesting that its vibrations are stable. Conversely, for a bearing with inner ring damage, its data is scattered across all the features.

Figure 14 - Summary 5 Feature Plot of MFPT



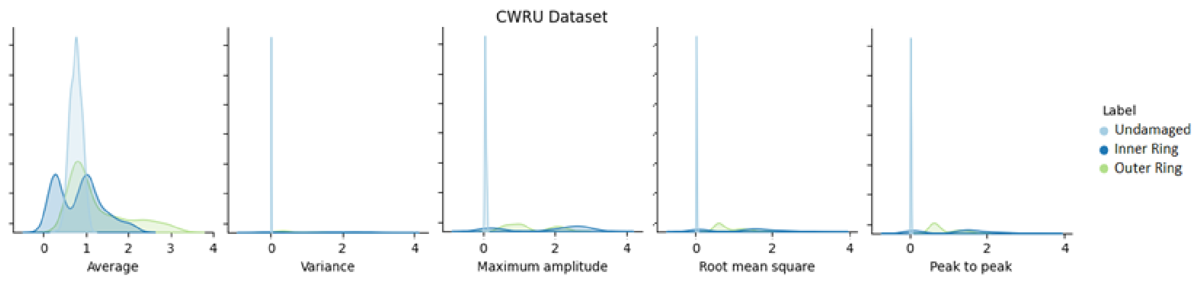
3.1.2 CWRU

Vibration signals were collected using a 16-channel digital audio tape (DAT).

Then, we extract 5 features. The data is plotted visually, for each corresponding dimension, as shown in Figure 31 in Appendix A. For this case study we join them and consider three classes: Undamaged (U), inner ring fault (IR) and outer ring fault (OR).

In the following graph in Figure 15, when we analyze the density distribution of each extracted feature, we can observe how the average of a time series can provide more information than the others in this dataset. An undamaged bearing exhibits tightly clustered values for each feature, suggesting that its vibrations are stable. Conversely, for a bearing with inner ring damage and outer ring damage, its data is scattered across all the features.

Figure 15 - Summary 5 Feature Plot of CWRU



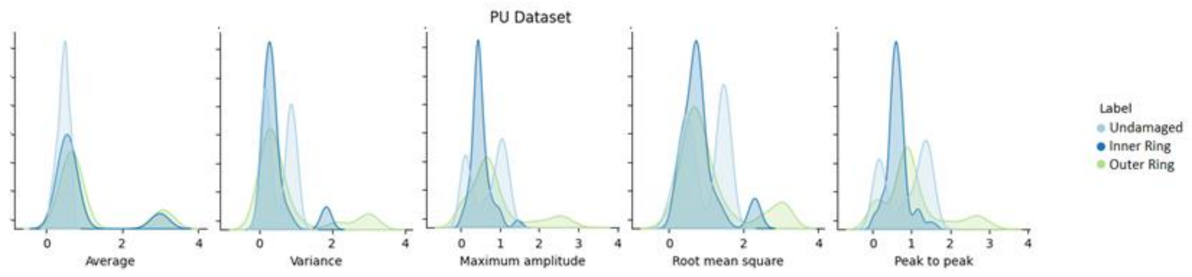
3.1.3 PU

This experiment use 32 different bearing damages in ball bearings of type were performed: Undamaged (healthy) bearings (6 experiments), artificially damaged bearings (12 experiment) and bearings with real damages caused by accelerated lifetime tests (14 experiment).

Then, we extract 5 features as mentioned before. The data is plotted visually, for each corresponding dimension, as shown in Figure 32 (Appendix); for this case study, we join them and consider three classes: Undamaged (U), inner ring fault (IR) and outer ring fault (OR).

In the following graph in Figure 16, when we analyze the density distribution of each extracted feature. For all bearing types the data is scattered across all the features. It can be difficult to provide an easy-to-compare description.

Figure 16 - Summary 5 Feature Plot of PU



3.2 Ansatz Circuits

After transforming the vibration data of each bearing database into important features such as average, variance, maximum amplitude, RMS, and peak to peak, we transformed the 'classical' data into quantum data and, next, we must input the latter through parameterizable circuits that can vary their values to learn and classify.

3.2.1 Transform Data Circuits

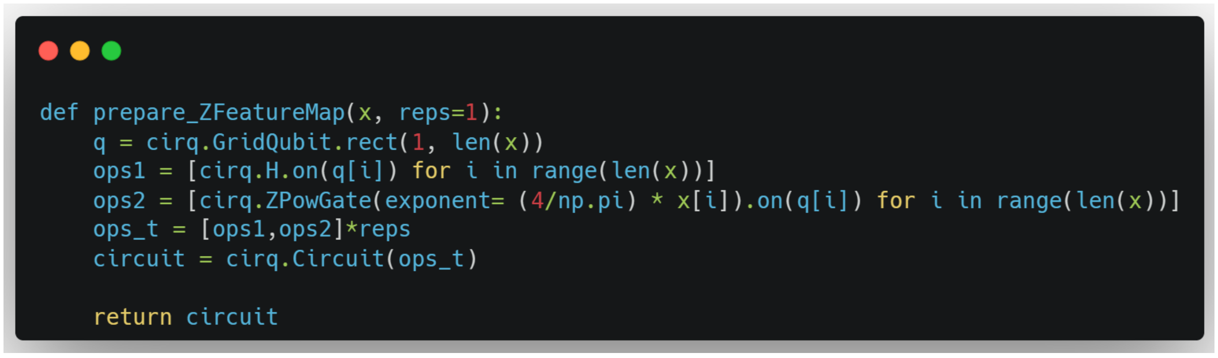
Two methods are used to transform the data, as specified in the Theoretical Background chapter, namely ZFeatureMap and ZZFeatureMap. In this thesis, the ZfeatureMap and ZZfeaturemap were utilized due to their prevalent usage in the existing literature and these were the ones that, with the same number of qubits for comparison, achieved better performance. (SURYOTRISONGKO and MUSASHI, 2022). However, it is important to note that these feature maps had not been extensively tested in the context of maintenance. Hence, they were chosen to explore their applicability in other types of models. By using these feature maps, the work aimed to contribute to the literature by evaluating their effectiveness and potentially revealing new insights for maintenance-related tasks. The codes to write these circuits in Cirq are respectively shown in Figure 17 and Figure 18.

In the function shown in Figure 17, since there is no entanglement, only the Repetition option is available, which specifies how many times the input will be repeated in the quantum circuit.

In the function shown in Figure 18, you can input the data inputs and then select the type of entanglement you want, such as Linear or Full. This determines the order in which each qubit is connected and with whom it is entangled.

In the utilized circuits, a repetition of 3 was used since no significant improvement was observed by increasing the repetition.

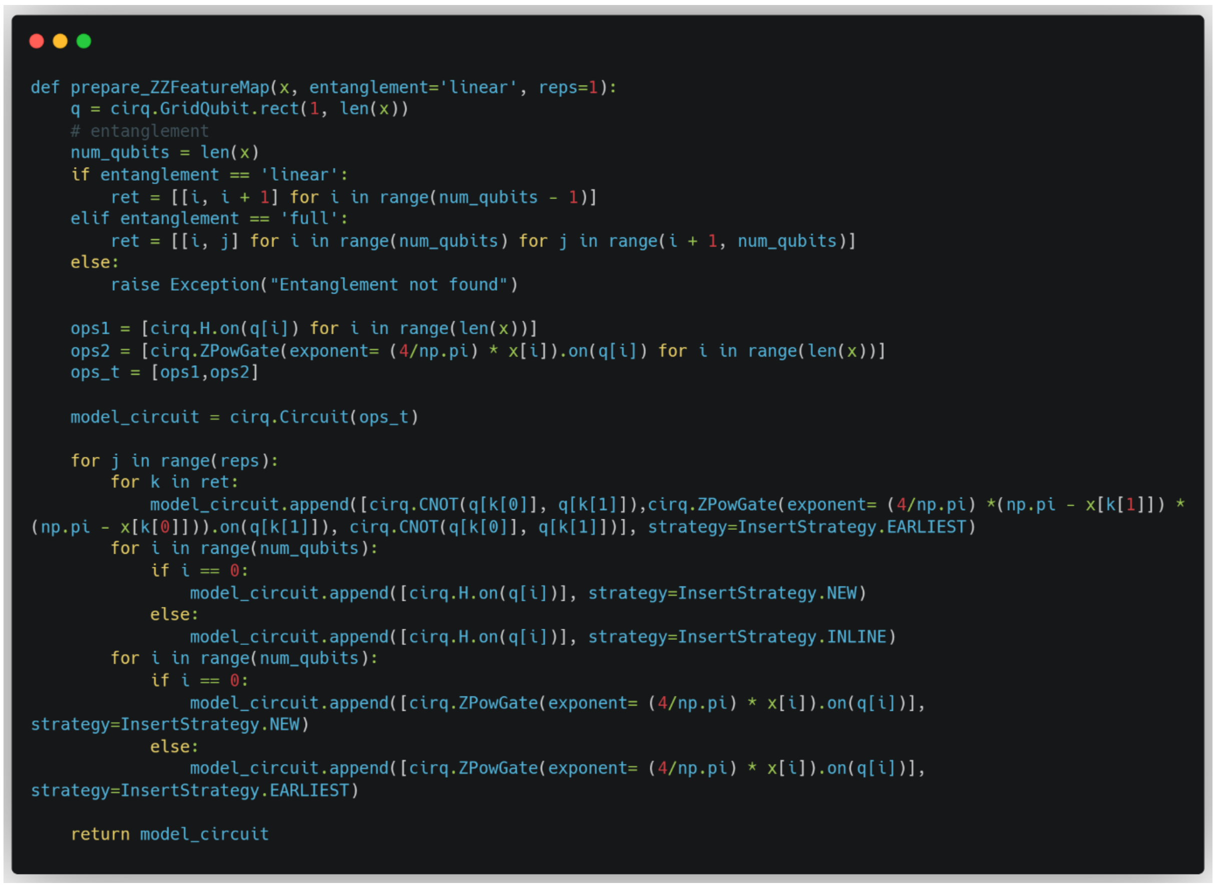
Figure 17 - Code of ZFeatureMap



```
def prepare_ZFeatureMap(x, reps=1):
    q = cirq.GridQubit.rect(1, len(x))
    ops1 = [cirq.H.on(q[i]) for i in range(len(x))]
    ops2 = [cirq.ZPowGate(exponent= (4/np.pi) * x[i]).on(q[i]) for i in range(len(x))]
    ops_t = [ops1,ops2]*reps
    circuit = cirq.Circuit(ops_t)

    return circuit
```

Source: The author (2023)

Figure 18 - Code of ZZFeatureMap

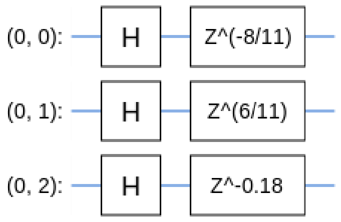


```
def prepare_ZZFeatureMap(x, entanglement='linear', reps=1):
    q = cirq.GridQubit.rect(1, len(x))
    # entanglement
    num_qubits = len(x)
    if entanglement == 'linear':
        ret = [[i, i + 1] for i in range(num_qubits - 1)]
    elif entanglement == 'full':
        ret = [[i, j] for i in range(num_qubits) for j in range(i + 1, num_qubits)]
    else:
        raise Exception("Entanglement not found")

    ops1 = [cirq.H.on(q[i]) for i in range(len(x))]
    ops2 = [cirq.ZPowGate(exponent= (4/np.pi) * x[i]).on(q[i]) for i in range(len(x))]
    ops_t = [ops1,ops2]

    model_circuit = cirq.Circuit(ops_t)

    for j in range(reps):
        for k in ret:
            model_circuit.append([cirq.CNOT(q[k[0]], q[k[1]]),cirq.ZPowGate(exponent= (4/np.pi) *(np.pi - x[k[1]]) *
(np.pi - x[k[0]])).on(q[k[1]]), cirq.CNOT(q[k[0]], q[k[1]])], strategy=InsertStrategy.EARLIEST)
        for i in range(num_qubits):
            if i == 0:
                model_circuit.append([cirq.H.on(q[i])], strategy=InsertStrategy.NEW)
            else:
                model_circuit.append([cirq.H.on(q[i])], strategy=InsertStrategy.INLINE)
        for i in range(num_qubits):
            if i == 0:
                model_circuit.append([cirq.ZPowGate(exponent= (4/np.pi) * x[i]).on(q[i])],
strategy=InsertStrategy.NEW)
            else:
                model_circuit.append([cirq.ZPowGate(exponent= (4/np.pi) * x[i]).on(q[i])],
strategy=InsertStrategy.EARLIEST)

    return model_circuit
```

Source: The author (2023)

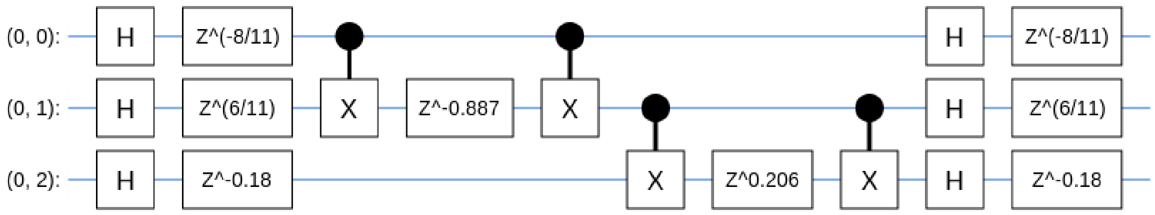
If we input the same input values, array with values [1,2,3], in both functions, with one repetition parameter, the circuits shown in Figure 19 and Figure 20 are generated, respectively.

Figure 19 - ZFeatureMap with values.



Source: The author (2023)

Figure 20- ZZFeatureMap with values.

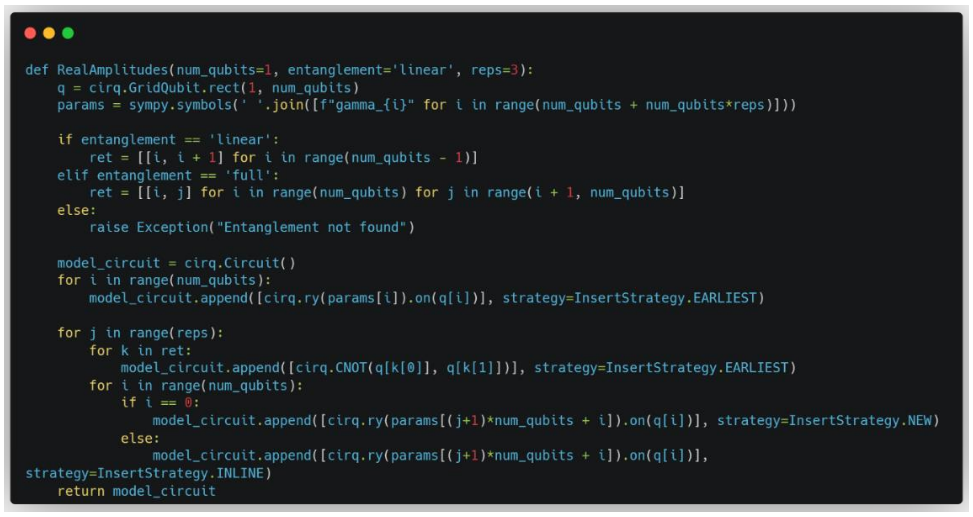


Source: The author (2023)

3.2.2 Parametrized Quantum Circuits

Two parameterized quantum circuits are proposed, also specified in Theoretical Background, which are Real Amplitudes and Quantum Convolutional Network. The codes to write these circuits in Cirq are respectively shown in Figure 21 and Figure 22.

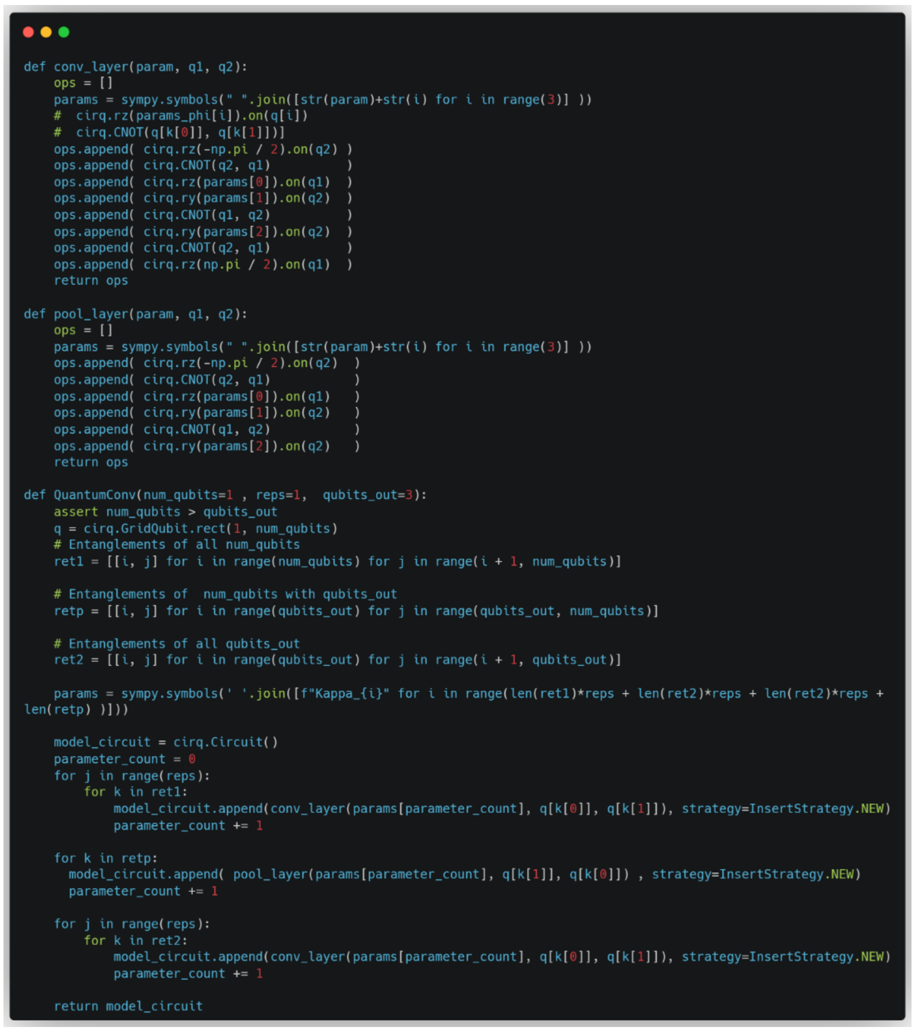
Figure 21 - Code of Real Amplitudes



```
def RealAmplitudes(num_qubits=1, entanglement='linear', reps=3):
    q = cirq.GridQubit.rect(1, num_qubits)
    params = sympy.symbols(' '.join([f"gamma_{i}" for i in range(num_qubits + num_qubits*reps)]))

    if entanglement == 'linear':
        ret = [[i, i + 1] for i in range(num_qubits - 1)]
    elif entanglement == 'full':
        ret = [[i, j] for i in range(num_qubits) for j in range(i + 1, num_qubits)]
    else:
        raise Exception("Entanglement not found")

    model_circuit = cirq.Circuit()
    for i in range(num_qubits):
        model_circuit.append([cirq.ry(params[i]).on(q[i])], strategy=InsertStrategy.EARLIEST)

    for j in range(reps):
        for k in ret:
            model_circuit.append([cirq.CNOT(q[k[0]], q[k[1]])], strategy=InsertStrategy.EARLIEST)
        for i in range(num_qubits):
            if i == 0:
                model_circuit.append([cirq.ry(params[(j+1)*num_qubits + i]).on(q[i])], strategy=InsertStrategy.NEW)
            else:
                model_circuit.append([cirq.ry(params[(j+1)*num_qubits + i]).on(q[i])],
strategy=InsertStrategy.INLINE)
    return model_circuit
```

Source: The author (2023)

Figure 22 - Code of Quantum Convolutional Neural Network

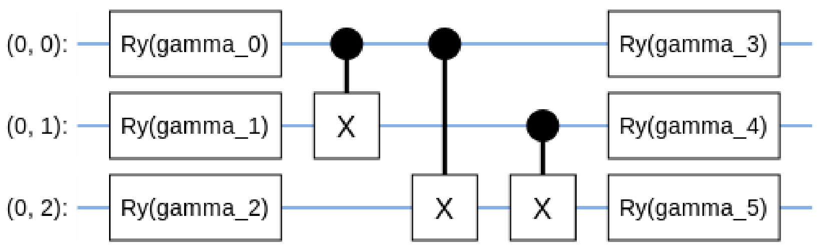


```python
def conv_layer(param, q1, q2):
    ops = []
    params = sympy.symbols(" ".join([str(param)+str(i) for i in range(3)] ))
    #  cirq.rz(params_phi[i]).on(q[i])
    #  cirq.CNOT(q[k[0]], q[k[1]])]
    ops.append( cirq.rz(-np.pi / 2).on(q2) )
    ops.append( cirq.CNOT(q2, q1)          )
    ops.append( cirq.rz(params[0]).on(q1)  )
    ops.append( cirq.ry(params[1]).on(q2)  )
    ops.append( cirq.CNOT(q1, q2)          )
    ops.append( cirq.ry(params[2]).on(q2)  )
    ops.append( cirq.CNOT(q2, q1)          )
    ops.append( cirq.rz(np.pi / 2).on(q1)  )
    return ops

def pool_layer(param, q1, q2):
    ops = []
    params = sympy.symbols(" ".join([str(param)+str(i) for i in range(3)] ))
    ops.append( cirq.rz(-np.pi / 2).on(q2)  )
    ops.append( cirq.CNOT(q2, q1)           )
    ops.append( cirq.rz(params[0]).on(q1)   )
    ops.append( cirq.ry(params[1]).on(q2)   )
    ops.append( cirq.CNOT(q1, q2)           )
    ops.append( cirq.ry(params[2]).on(q2)   )
    return ops

def QuantumConv(num_qubits=1 , reps=1,  qubits_out=3):
    assert num_qubits > qubits_out
    q = cirq.GridQubit.rect(1, num_qubits)
    # Entanglements of all num_qubits
    ret1 = [[i, j] for i in range(num_qubits) for j in range(i + 1, num_qubits)]

    # Entanglements of  num_qubits with qubits_out
    retp = [[i, j] for i in range(qubits_out) for j in range(qubits_out, num_qubits)]

    # Entanglements of all qubits_out
    ret2 = [[i, j] for i in range(qubits_out) for j in range(i + 1, qubits_out)]

    params = sympy.symbols(' '.join([f"Kappa_{i}" for i in range(len(ret1)*reps + len(ret2)*reps + len(ret2)*reps +
len(retp) )]))

    model_circuit = cirq.Circuit()
    parameter_count = 0
    for j in range(reps):
        for k in ret1:
            model_circuit.append(conv_layer(params[parameter_count], q[k[0]], q[k[1]]), strategy=InsertStrategy.NEW)
            parameter_count += 1

    for k in retp:
      model_circuit.append( pool_layer(params[parameter_count], q[k[1]], q[k[0]]) , strategy=InsertStrategy.NEW)
      parameter_count += 1

    for j in range(reps):
        for k in ret2:
            model_circuit.append(conv_layer(params[parameter_count], q[k[0]], q[k[1]]), strategy=InsertStrategy.NEW)
            parameter_count += 1

    return model_circuit
```

Source: The author (2023)

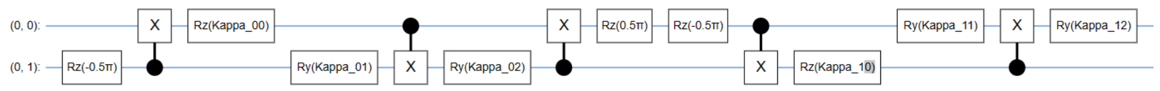
If we use both functions, with one repetition parameter, the circuits shown in Figure 21and Figure 24 are generated, respectively.

Figure 23 – Real Amplitudes circuits in Cirq



Source: The author (2023)

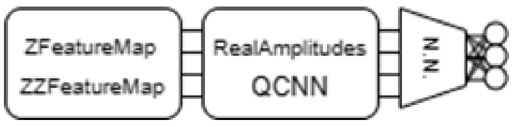
Figure 24 - Quantum Convolutional Neural Network circuits in Cirq



Source: The author (2023)

Finally, for the hybrid circuits, a layer of a neural network is attached to the output of the parameterized quantum circuits, as shown as Figure 25. The neural network used consists of one hidden layer with 15 neurons and ReLU activation function. The output layer has 3 neurons with SoftMax activation function.

Figure 25 - Diagram of hybrid circuits.



Source: The author (2023)

3.2.3 Training an only quantum circuit

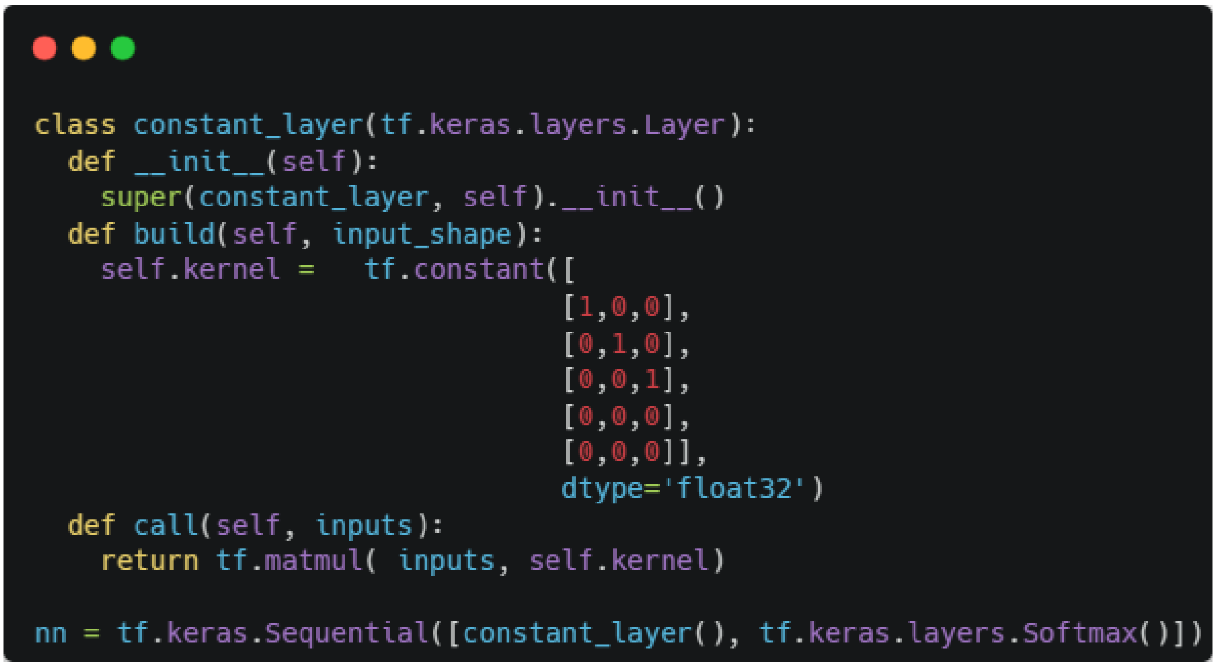
When working with TensorFlow Quantum (TFQ) to train a quantum circuit, a technique called hybrid quantum learning can be utilized. In this approach, a quantum circuit is combined with a classical neural network to perform classification or regression tasks using quantum data. We will explain how to introduce a quantum circuit into TensorFlow Quantum (TFQ) using only an untrained quantum circuit.

To train the quantum circuit in TFQ, a customized neural network acting as a quantum classifier or regressor must be constructed. This neural network is configured to have only constant non-trainable parameters at its output. This means that parameter values will not be adjusted during the training process and will remain fixed.

The customized neural network is designed according to the specific requirements of the problem and the number of available labels or classes. In the mentioned case, where there are 3 labels and 5 qubits, a matrix can be built that takes the first 3 qubits of the quantum circuit as shown in Figure 26. This matrix is used as the output of the neural network, with the non-trainable constant parameters defining how the qubits relate to the labels.

Once the structure of the quantum neural network has been defined, the training process can proceed. During training, the parameters of the quantum circuits will be adjusted, while the non-trainable constant parameters of the neural network will remain unchanged. This allows for training exclusively the quantum circuit. To perform the necessary calculations at the output of the quantum circuit, a SoftMax activation function is applied, which takes the output values and normalizes them to obtain a probability distribution representing the probability of belonging to each of the available labels. To sum up, when training a quantum circuit in TFQ, a customized neural network with non-trainable constant parameters is constructed to define the relationship between the qubits and the labels.

Figure 26 - Constant Layer



```
class constant_layer(tf.keras.layers.Layer):
  def __init__(self):
    super(constant_layer, self).__init__()
  def build(self, input_shape):
    self.kernel =   tf.constant([
                                [1,0,0],
                                [0,1,0],
                                [0,0,1],
                                [0,0,0],
                                [0,0,0]],
                                dtype='float32')
  def call(self, inputs):
    return tf.matmul( inputs, self.kernel)

nn = tf.keras.Sequential([constant_layer(), tf.keras.layers.Softmax()])
```

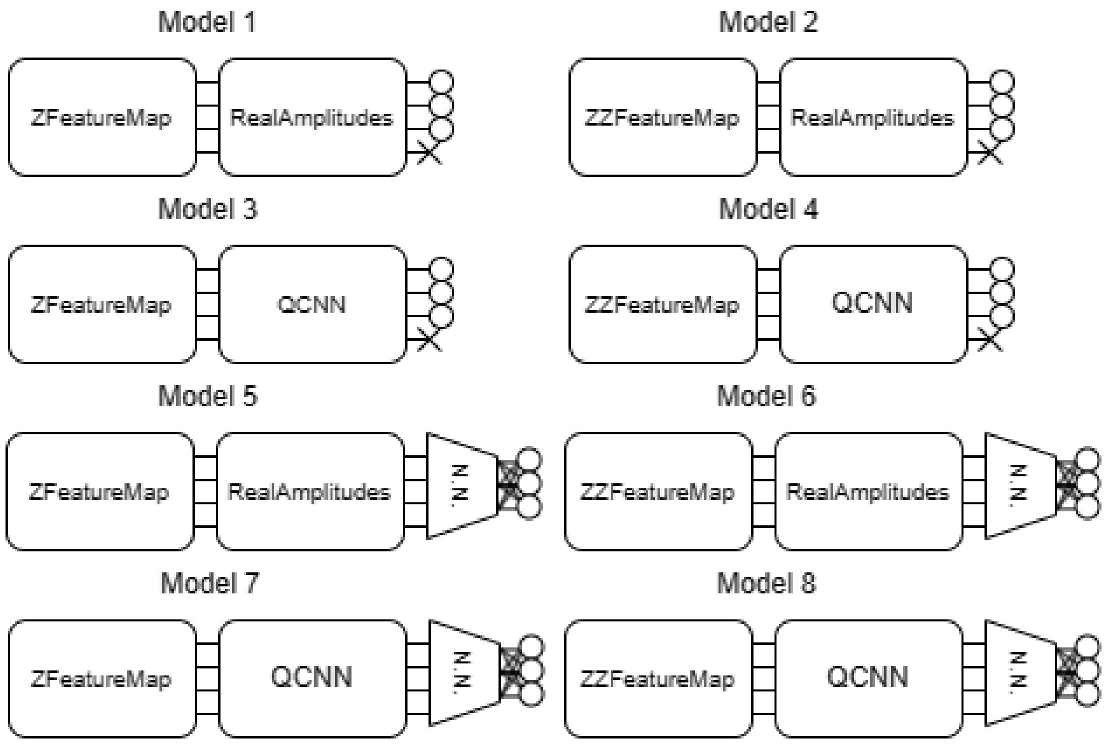
Source: The author (2023)

3.2.4 Proposed models and training

For the proposed models, a circuit to transform classical data to quantum data, a parameterized circuit, and to connect to neural network selected. In summary, the initial set of four models comprises quantum circuits operating independently of neural networks. Subsequently, we replicate the same set of models, incorporating neural networks after the circuits. For every model, we conduct three iterations, followed by the calculation of their respective metrics. This enables us to make meaningful comparisons between the models.

The above-mentioned information is visualized in Figure 27, which presents a schematic demonstrating the utilization of the first three wires of the quantum circuits for prediction in the initial four models. Conversely, in the last four models, a neural network is employed, leveraging the output from all the circuits to make predictions. Adam is used for training in the sessions.

Figure 27 - Schematic diagram of the utilized models



Source: The author (2023)

4 RESULTS

As previously mentioned, 8 models were used for the analysis. However, due to time constraints and the computational resources available, it was only feasible to run each model 3 times. Despite the limited number of runs, it was observed that the results exhibited a consistent trend, suggesting a degree of stability. While running the models multiple times would have provided more statistical robustness, the 3 runs conducted were sufficient to identify and assess the general performance patterns of each model. It is important to note that a larger number of runs would have provided a more comprehensive analysis, but within the given constraints, the 3 runs still yielded meaningful insights and reliable indications of model performance. The Loss and Accuracy plots, as well as the Confusion Matrix of the models, are displayed in APPENDIX B and the results of all the used models will be shown in APPENDIX C. Now, a summary of all the models used in this thesis will be presented.

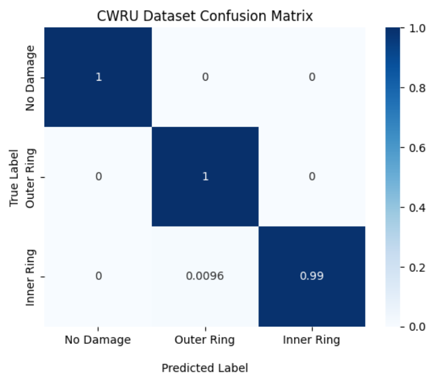
Table 1- Summary of results of models

Models	MFPT				CWRU				PU			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Model 1	73.97	73.04	75.00	74.27	83.22	82.47	86.47	82.94	66.38	63.92	71.76	68.05
Model 2	79.53	79.17	79.35	79.82	83.75	83.12	86.53	83.46	78.42	78.47	78.61	79.08
Model 3	77.58	77.17	78.38	77.88	85.31	84.56	89.51	85.07	69.81	68.24	74.83	71.51
Model 4	86.45	86.69	87.37	86.58	98.22	98.22	98.34	98.20	82.22	82.28	82.62	82.80
Model 5	92.20	92.22	92.42	92.21	99.68	99.68	99.68	99.68	91.94	91.97	91.97	92.09
Model 6	**92.78**	**92.74**	**92.81**	**92.87**	99.79	99.78	99.78	99.78	91.20	91.38	91.44	91.44
Model 7	92.49	92.49	92.60	92.50	99.79	99.78	99.78	99.78	**93.88**	**93.99**	**93.96**	**94.05**
Model 8	91.52	91.55	91.75	91.59	**100.0**	**100.0**	**100.0**	**100.0**	90.64	90.70	90.98	90.90

As a result, the best way to enter the data of the main features is through a ZZFeatureMap circuit, which explains that one qubit must be dedicated to each feature extracted from the data. The use of ZZFeatureMap gives us a better precision of approximately 6% compared to the ZFeatureMap (Compared Models without using neural networks). Regarding selecting parameterized circuits, it can be observed that quantum convolutional circuits yield better results, which can be attributed to their larger number of parameters available for training. Nevertheless, hybrid models deliver better precision than their counterparts that does not use neural networks, which can say that neural networks are still a powerful tool to use to classify data, even after they have been processed in quantum circuits. Hybrid models give us a 10% improvement in precision.

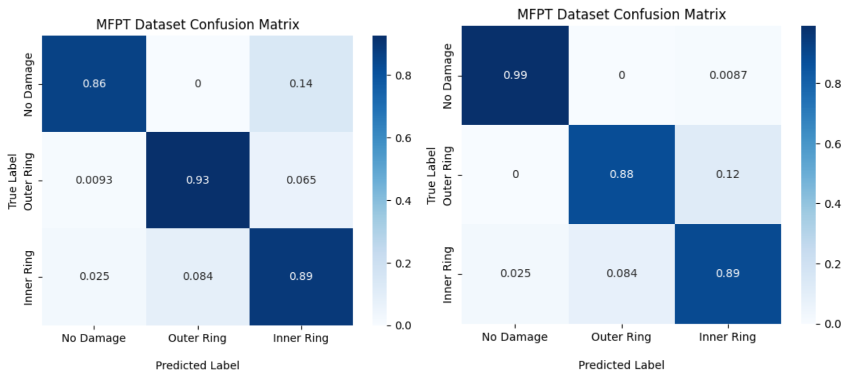
It's important to remember that the percentage results shown are based on a test dataset, so the small errors observed in CWRU are errors that stem from a limited amount of data used for testing as is shown in Figure 28.

Figure 28 - Confusion Matrix Model 6 of CWRU



As can be seen in several MFPT results, as shown in the Figure 29, it's easy to classify the undamaged bearings, as observed in the methodology and the graphs. This indicates that in both hybrid and non-hybrid models, it's possible to distinguish the undamaged category.

Figure 29 - Confusion Matrix Model 4 and 6 of MFPT



In the bibliography, Zhao also used the same datasets, from which we will extract only the best results of the MLP. This allows us to observe that by using neural networks alone, we achieve better performance than a hybrid model. This comparison may be due to the differences in parameters used in the models, but it can be said that currently, better results can be achieved without high quantum complexity.

Table 2 - Summary results MLP models of Zhao (ZHAO, LI, *et al.*, 2020).

Models	MFPT		CWRU		PU	
	Mean (%)	Max (%)	Mean (%)	Max (%)	Mean (%)	Max (%)
A 2	96.23	96.5	100	100	98.53	99.23
B 2	94.52	94.95	100	100	97.97	98.46
C 2	95.88	97.28	100	100	98.22	98.92

A factor that cannot be omitted in the comparison of models is the quantity of parameters demanded by neural networks. Even with small neural networks (3 layers), there can be 906 trainable parameters, of which a quantum network only utilizes 408 in the QCNN. The contrast in trainable parameters observed in the Table 3 could potentially explain the variation in accuracy results shown earlier.

Table 3 - Number of parameters per model.

Models	#Parameters	
	Parameters in quantum circuits	Parameters in neural nets
Model 1	55	0
Model 2	55	0
Model 3	408	0
Model 4	408	0
Model 5	55	903
Model 6	55	903
Model 7	408	903
Model 8	408	903

5 CONCLUSIONS

The three bearing databases were used for this thesis and were utilized satisfactorily. Eight models were created, to be compared both in terms of their data processing methods and the parametrization of their data for training and learning. Neural networks are also used to see how much they contribute to the precision of each model.

It can be concluded that neural network models achieve accuracy results above 90%, while accuracy is around 65% without the use of neural networks. However, with these results, a comparison can only be made for the quantum circuit to determine which suggests that the parameterized quantum circuits used in this thesis, such as the Real Amplitudes circuits in Qiskit and the quantum convolutional circuit. are delivering stable results. However, as mentioned earlier, it is important to ensure that the testing dataset also achieves similar accuracy rates to avoid overfitting or underfitting issues.

If only parameterized circuits are used for classification, there is still time to move away from depending solely on neural networks. Research is needed to avoid underfitting results, as is the case with PU datasets. Moreover, advances in regularization techniques, such as dropout in neural networks, will need to be developed in the same way for quantum circuits, as it is often observed that the training dataset and testing dataset tend to diverge and do not converge to a similar value.

We must consider that this work is run on classical computers, so the calculations performed by the models are analytical and simulate a quantum computer. Therefore, the results obtained are as clean as possible since they do not experience the "quantum noise" that is currently delivered by the hardware. The execution times in this master thesis are not meant to be compared against a classical neural network, as the results obtained using Cirq are analytical results of the circuits, and as mentioned earlier, are not run on a quantum computer.

As future work, we can perform testing of the same models using Qiskit, which, like Cirq, allows us to run quantum circuits on real quantum machines rather than just performing simulations. Considering this, it will be necessary to delve further into addressing issues related to noise correction, as well as exploring the matrices of quantum computers. The digitization of quantum errors means it is possible to reuse certain techniques from classical coding theory in quantum error correction. However, there remains a number of complications that prevent the straightforward translation of classical codes to quantum codes (JOSCHKA, 2019). The interconnections between qubits can impact the positioning of qubits within a quantum circuit, which holds significant importance since the quantum state of a qubit can be influenced by its neighboring qubits. This can cause a phenomenon known as quantum interference, which can affect the accuracy of calculations in the circuit. Therefore, it is important to carefully design the location of qubits in the circuit to minimize interference and maximize the accuracy of results (NIELSEN and CHUANG, 2000).

The code library is available at: https://github.com/diegoaichele/MasterThesisDiegoA

REFERENCES

ASHER PERES, D. R. T. Quantum Information and Relativity Theory. **arXiv:quant-ph/0212023 [quant-ph]**, 2003.

BANCHI, L.; CROOKS, G. Measuring Analytic Gradients of General Quantum Evolution with the Stochastic Parameter Shift Rule. **Quantum Journal**, v. 5, p. 386, January 2021.

BECHHOEFER, E. A Quick Introduction to Bearing Envelope Analysis. **Society for Machinery Failure Prevention Technology (MFPT)**, 2018. Disponivel em: <https://mfpt.org/wp-content/uploads/2018/03/MFPT-Bearing-Envelope-Analysis.pdf>. Acesso em: Fevereiro 2022.

BIAO WANG, Y. L. A Hybrid Prognostics Approach for Estimating Remaining Useful Life of Rolling Element Bearings. **EEE Transactions on Reliability**, v. 36, n. 1, 2020. ISSN 401 - 412.

BRECHER, C. Bearing. **Encyclopedia of Production Engineering. Springer, Berlin, Heidelberg.**, p. 7, 2014.

CIRQ. CIRQ, 2020. Disponivel em: <https://quantumai.google/cirq>. Acesso em: Fevereiro 2022.

CONG, E. A. Quantum Convolutional Neural Networks. **Arxiv**, p. 12, 2019.

CROOKS, G. E. Gradients of parameterized quantum gates using the parameter-shift rule and gate. **Arxiv**, p. 5, 2019.

FARHI, E.; GOLDSTONE, J.; GUTMANN, S. A Quantum Approximate Optimization Algorithm. **arXiv:1411.4028 [quant-ph]**, 2014.

HU, Y.; MIAO, X. Prognostics and health management: A review from the perspectives of design, development and decision. **Reliability Engineering & System Safety**, 2022.

IBM. 3 Things to Know About IBM's Development Roadmap to Build an Open Quantum Software Ecosystem. **IBM**, 2021. Disponivel em: <https://newsroom.ibm.com/3-Things-to-Know-About-IBMs-Development-Roadmap-to-Build-an-Open-Quantum-Software-Ecosystem>. Acesso em: Fevereiro 2022.

JORDAN, S. P. Fast Quantum Algorithm for Numerical Gradient Estimation. **PHYSICAL REVIEW JOURNALS (APS)**, p. 4, 2004.

JOSCHKA, R. Quantum Error Correction: An Introductory Guide, 25 Julho 2019. 29.

KYRIIENKO , ; ELFVING,. Generalized quantum circuit differentiation rules. **Phys Rev A**, v. 104, n. 5, p. 12, November 2021.

LESSMEIER, C. KAt-DataCenter. **Chair of Design and Drive Technology, Paderborn University**, 2020. Disponivel em: <KAt-DataCenter>. Acesso em: Fevereiro 2022.

LIU, Y. A rigorous and robust quantum speed-up in supervised machine learning. **arXiv:2010.02174v2 [quant-ph]**, p. 27, 2020.

MITARAI, K. et al. Quantum Circuit Learning, p. 7, 2019.

MYRON E., H.; DERONG, L.; STANLEY H., S. Solving the N-bit parity problem using neural networks. **Neural Networks**, 11 Maio 1999. 3.

NIELSEN, M. A.; CHUANG, I. L. **Quantum Computation and Quantum Information**. 2010. ed. New York: Cambridge University Press, 2000.

OH, S. A Tutorial on Quantum Convolutional Neural. **arXiv**, 2020.

PENNYLANE. Glossary of Quantum Machine Learning. **Glossary and key concepts of Quantum Machine Learning**, 2022. Disponivel em: <https://pennylane.ai/qml/glossary.html>.

PERAL GARCÍA, D. Systematic Literature Review: Quantum Machine Learning and its applications, 11 Janeiro 2022.

QISKIT. Open-Source Quantum Development. **Qiskit**, 2020. Disponivel em: <https://qiskit.org/>. Acesso em: Fevereiro 2022.

QISKIT. Representing Qubit States. **Qiskit**, 2020. Disponivel em: <https://qiskit.org/textbook/ch-states/representing-qubit-states.html>. Acesso em: Fevereiro 2022.

QISKIT. Hybrid quantum-classical Neural Networks with PyTorch and Qiskit. **Qiskit TextBook**, 2021. Disponivel em: <https://qiskit.org/textbook/ch-machine-learning/machine-learning-qiskit-pytorch.html>. Acesso em: FEVEREIRO 2022.

RIETSCHE, R. et al. QUANTUM COMPUTING. **Electron Markets**, p. 13, August 2022.

SAN MARTÍN, G.; LÓPEZ, E. Quantum Machine Learning for Health State Diagnosis and, 2021.

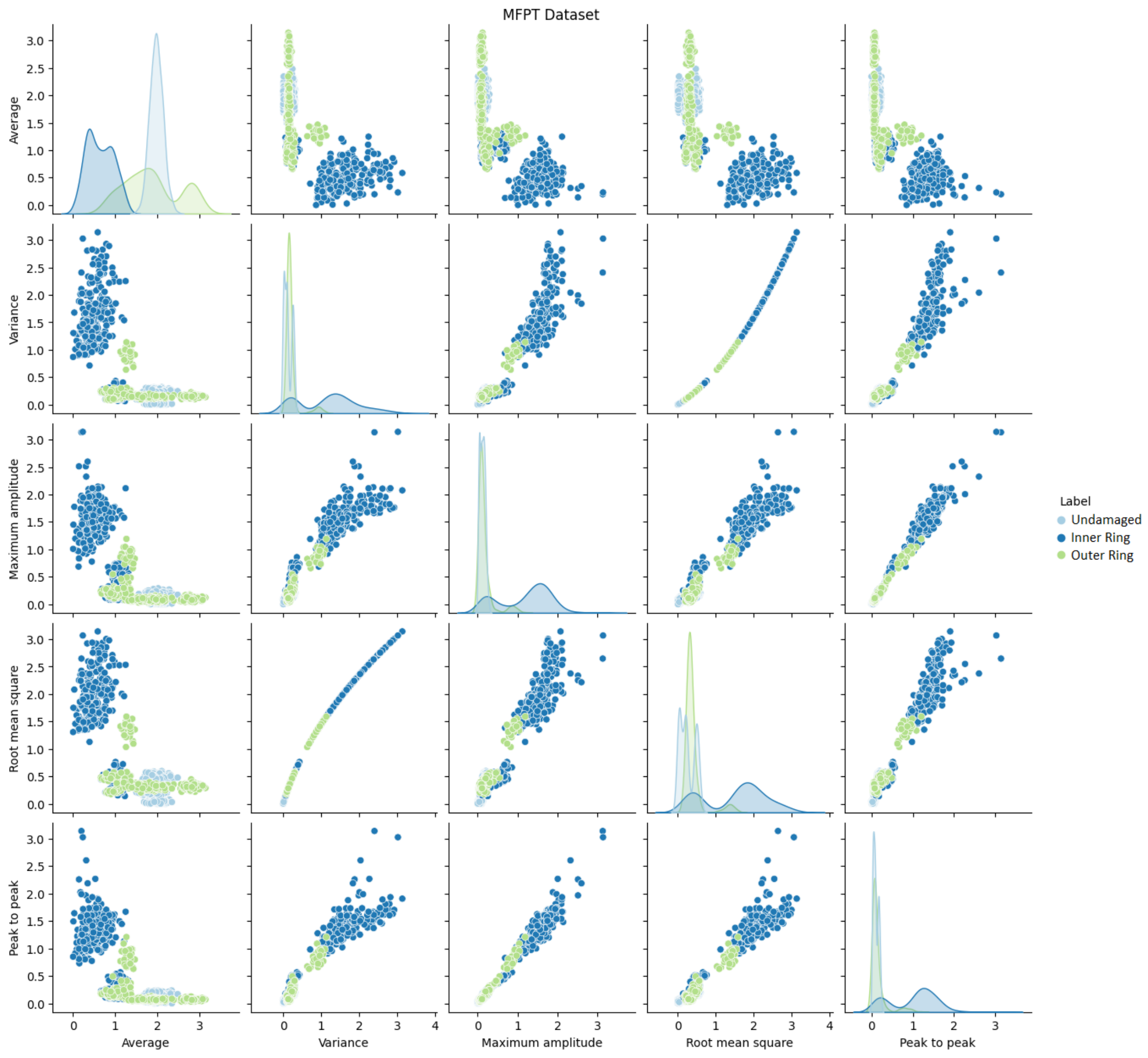
SMITH, W. A.; RANDALL, R. B. Rolling element bearing diagnostics using the Case Western Reserve University data: A benchmark study. **Mechanical Systems and Signal Processing**, Sydnei, v. 64 - 65, p. 100 - 131, Decembro 2015.

SOUTO, C. et al. Prognostics and health management via quantum machine. **IEEE**, p. 25132 - 25151, 2023.

SURYOTRISONGKO, H.; MUSASHI,. Hybrid Quantum Deep Learning and Variational Quantum Classifier-Based. **International Journal of Intelligent Engineering and Systems**, v. 15, n. 3, p. 215-224, June 2022.

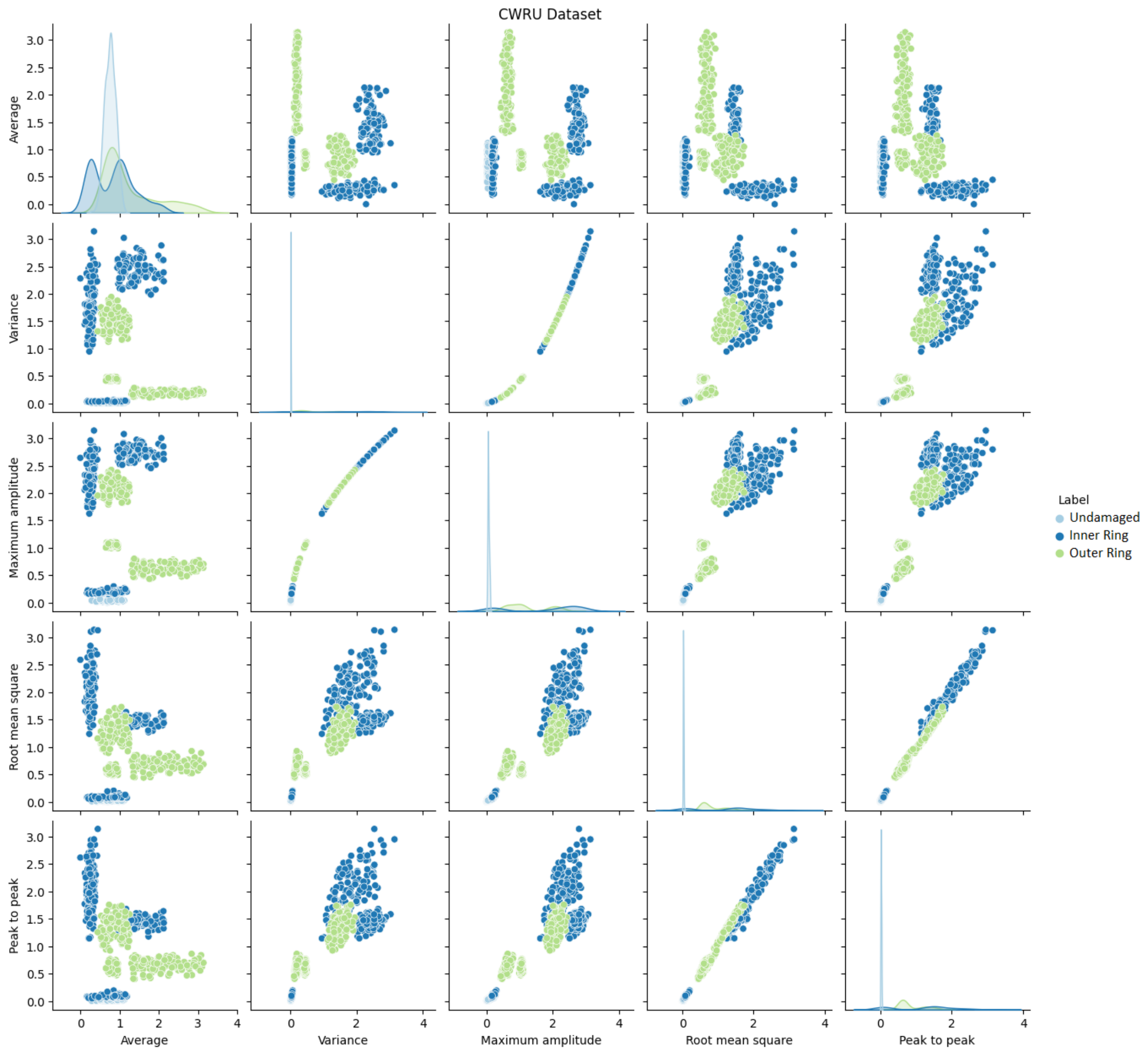
APPENDIX A – 5 FEATURE PLOT OF THE DATASETS

Figure 30 - MFPT Dataset



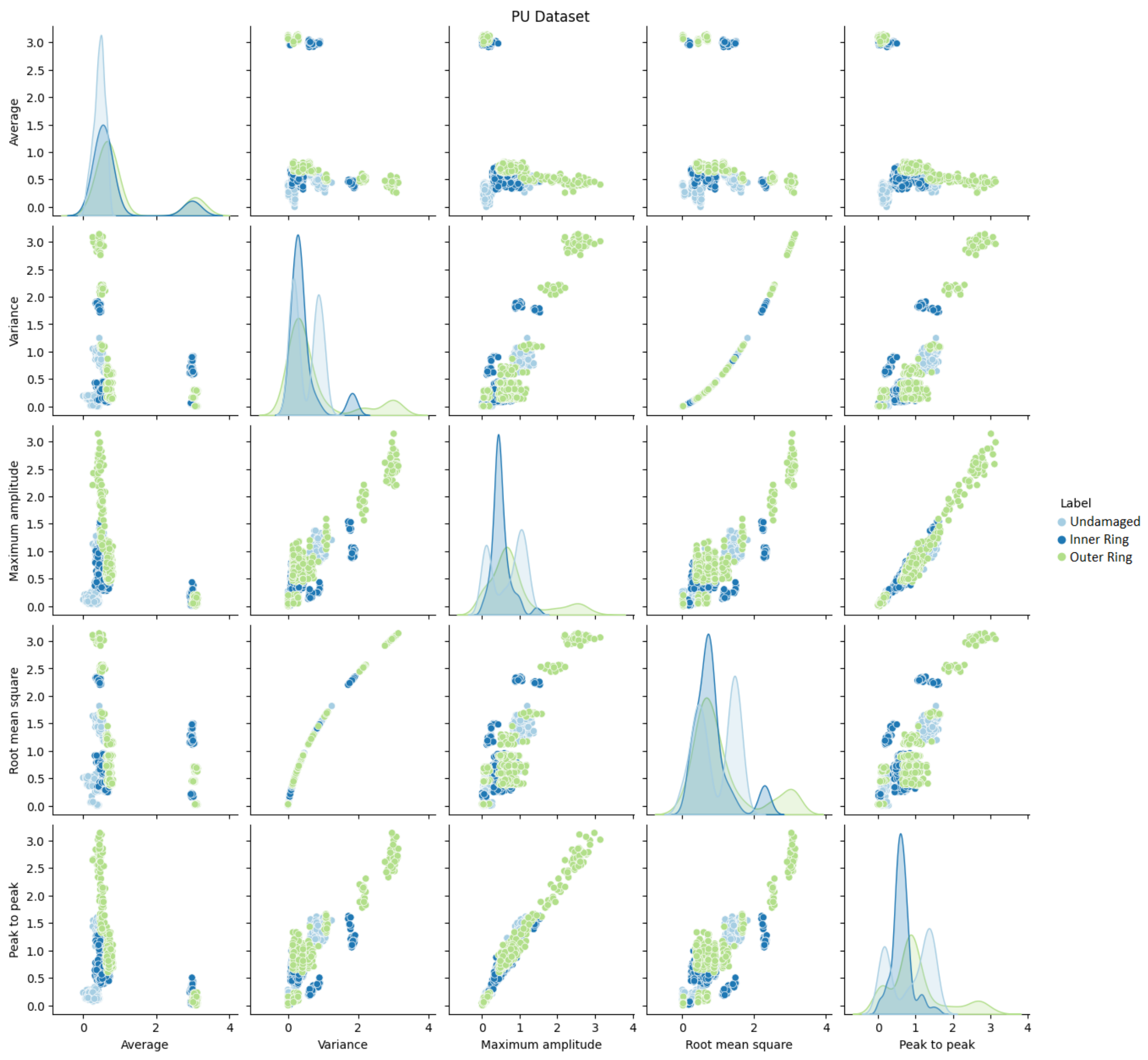
Source: The author (2023)

Figure 31 - CWRU Dataset



Source: The author (2023)

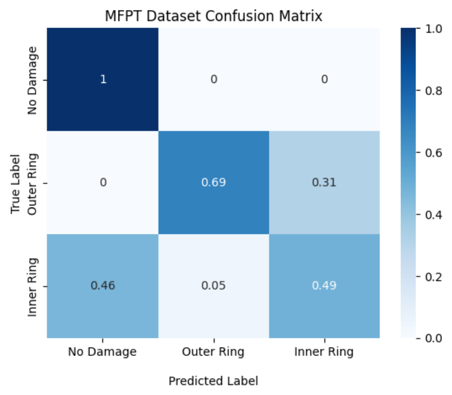
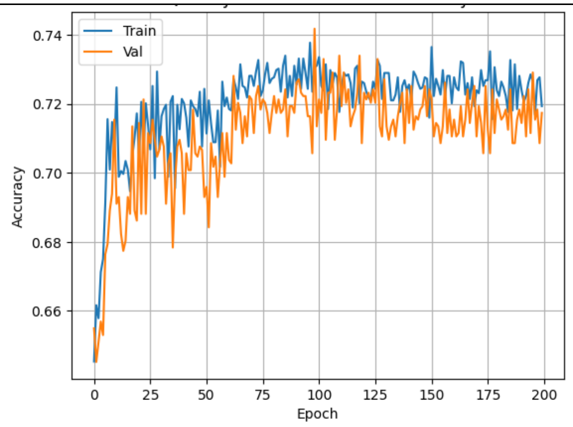
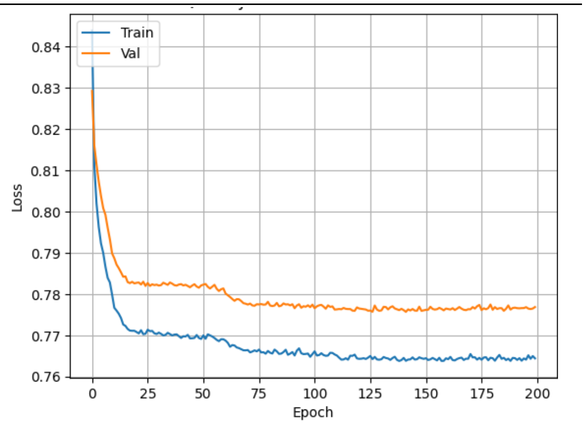
Figure 32 - PU Dataset

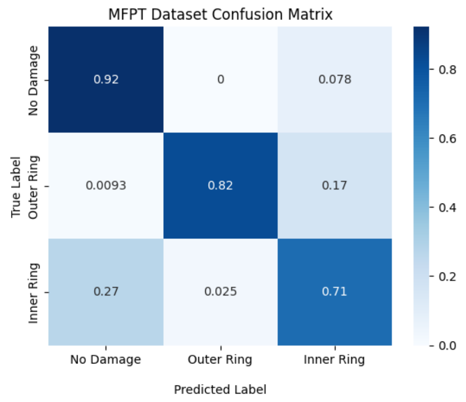
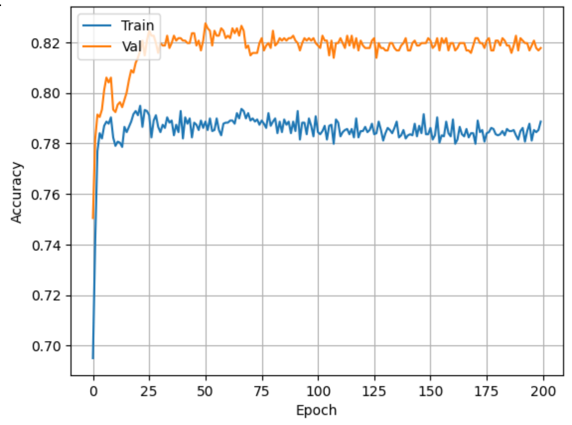
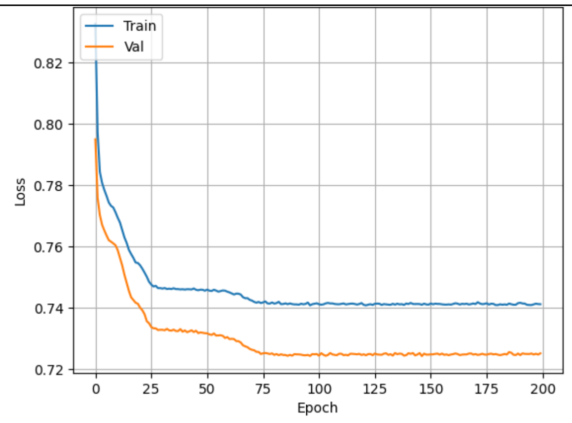
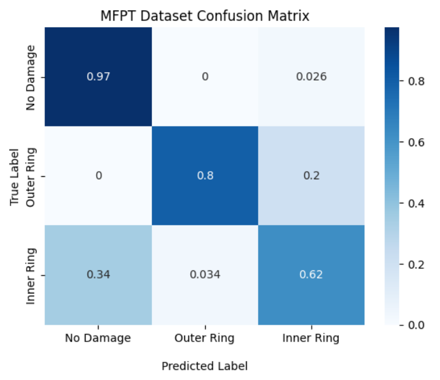
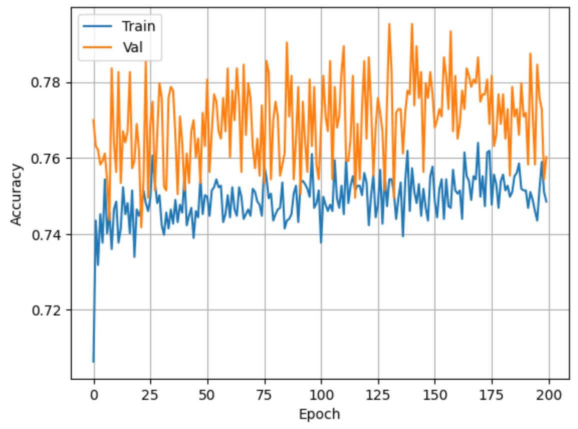
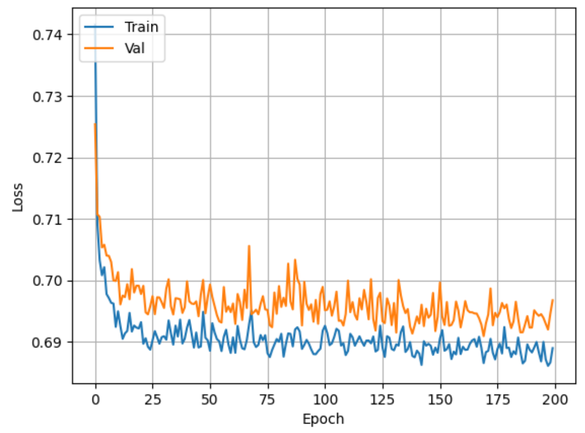


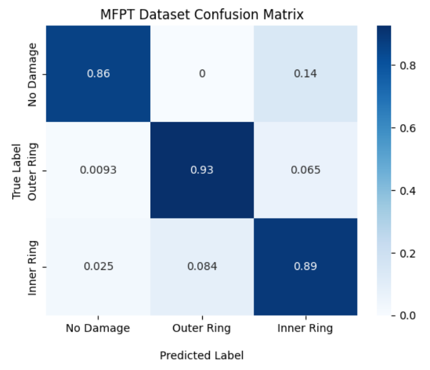
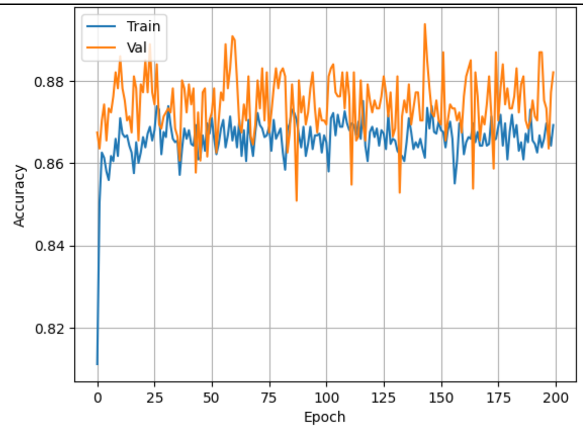
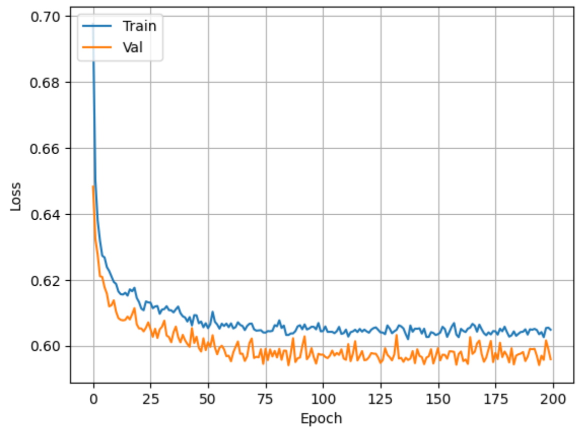
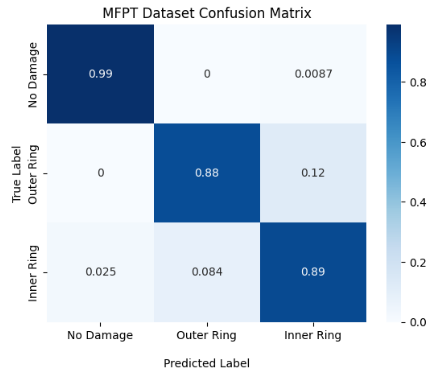
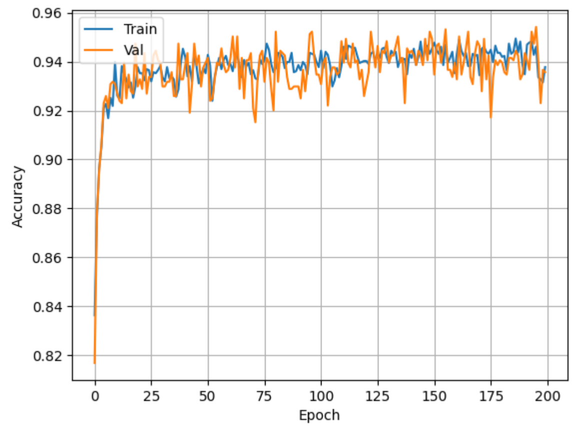
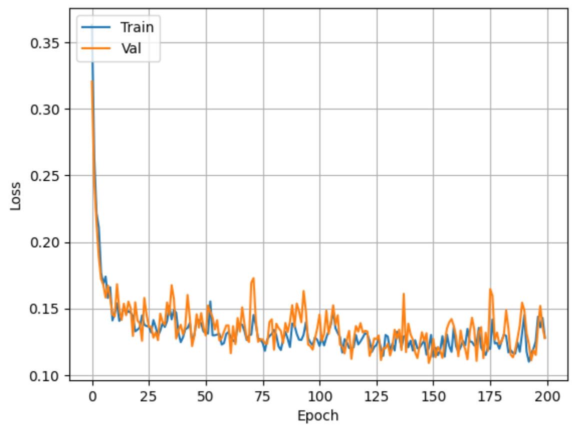
Source: The author (2023)

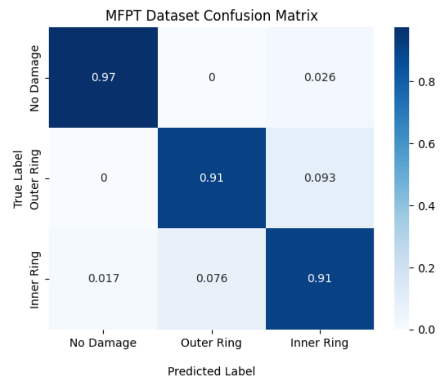
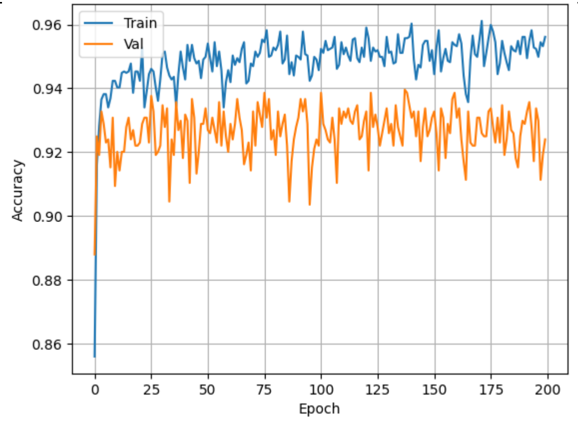
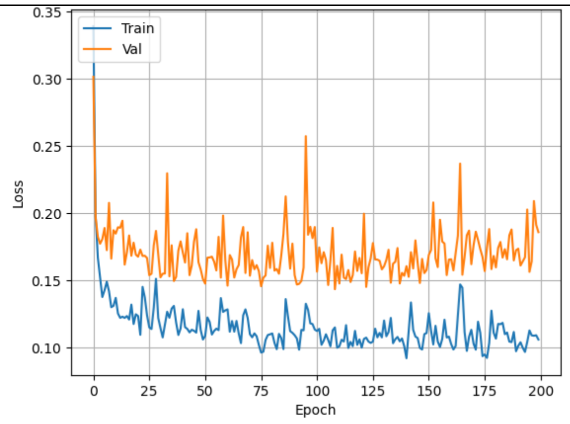
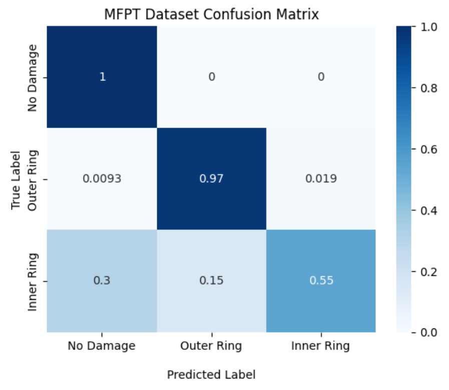
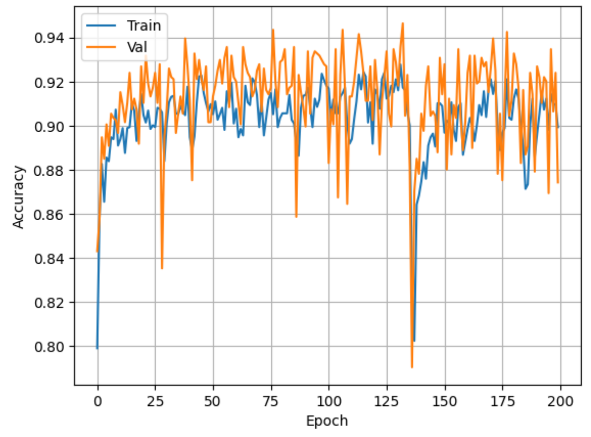
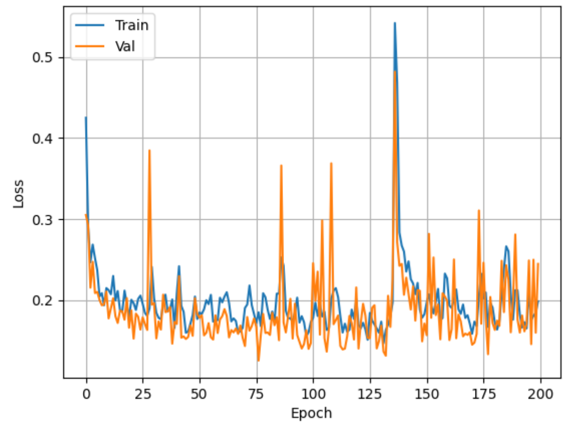
APPENDIX B – CONFUSION MATRIX, ACCURACY AND LOSS OF TRAINED MODELS

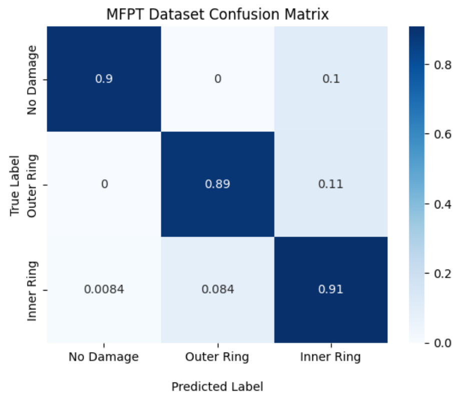
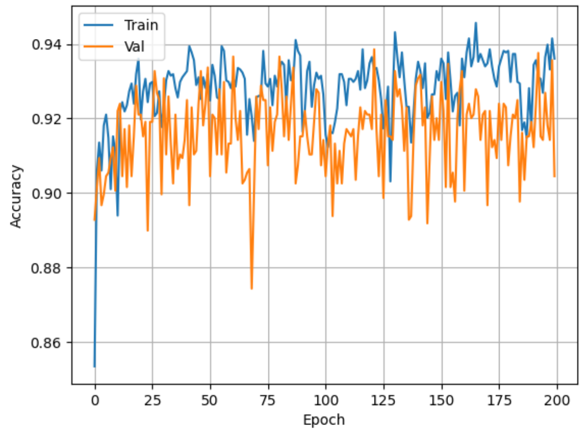
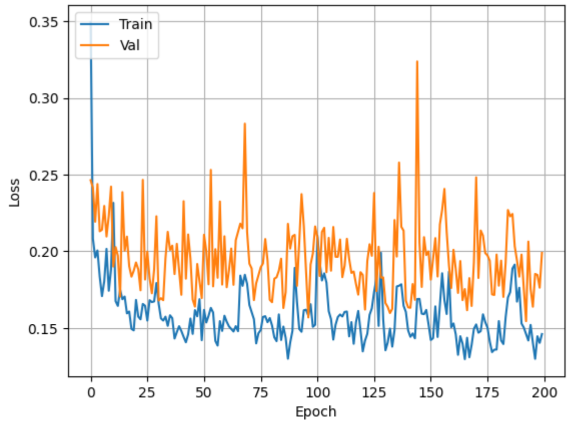
MFPT Results

Model	Confusion Matrix	Accuracy	Loss
Model 1			

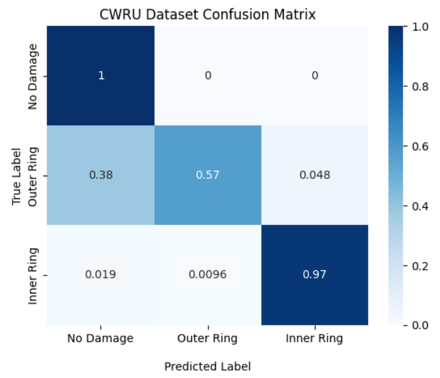
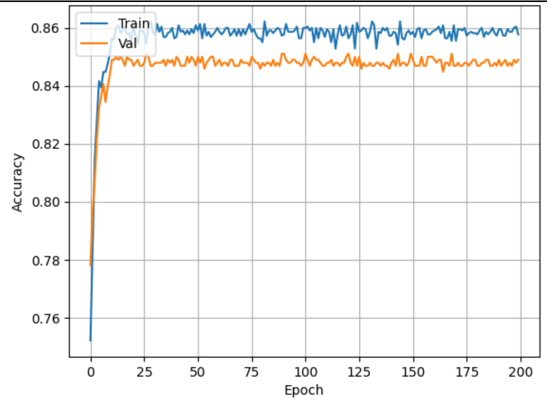
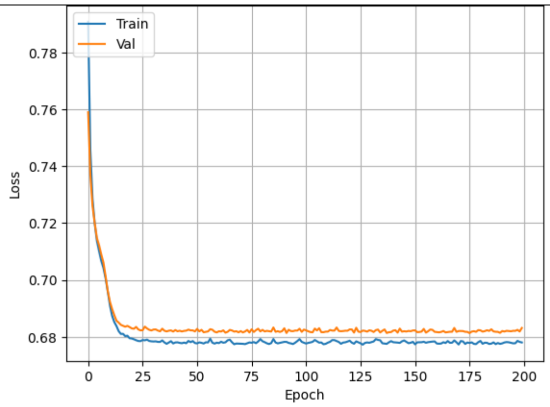
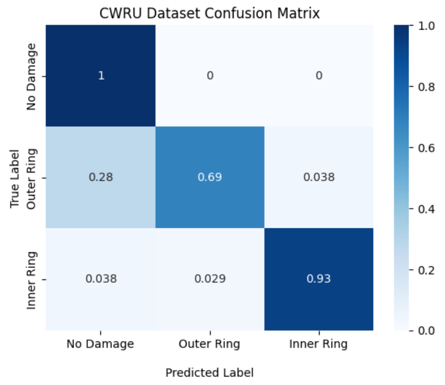
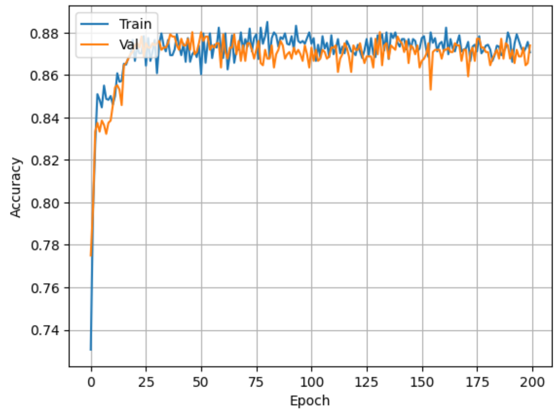
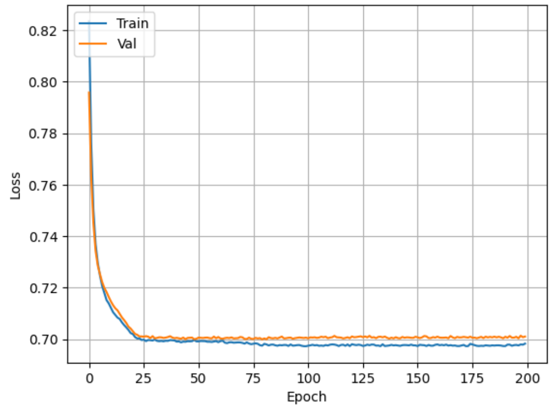
Model	**Confusion Matrix**	**Accuracy**	**Loss**
Model 2			
Model 3			

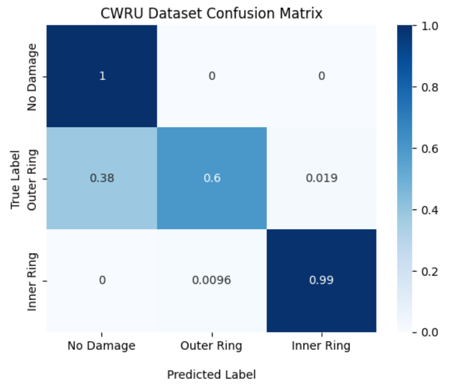
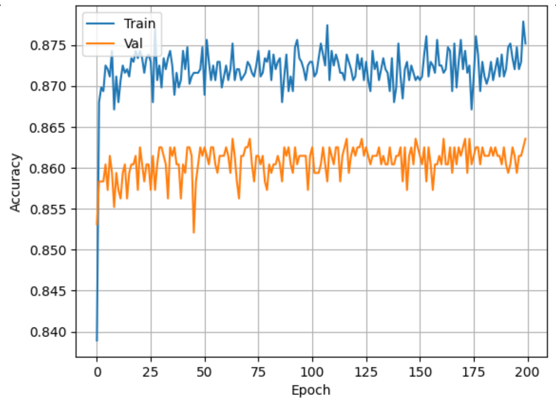
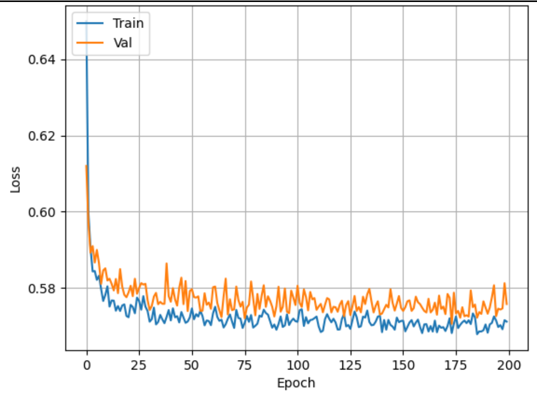
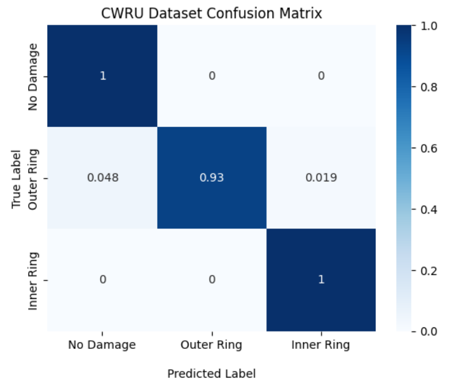
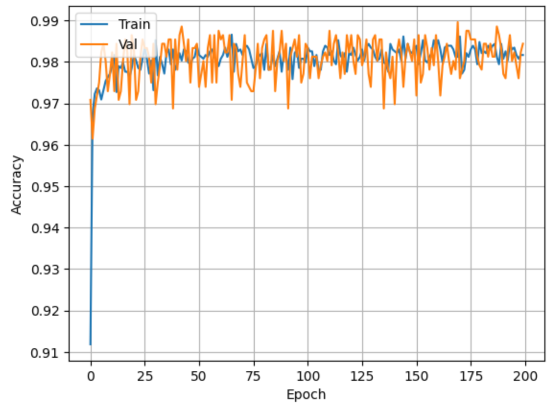
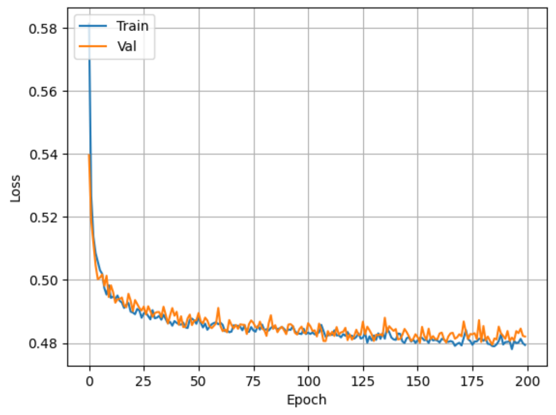
Model	Confusion Matrix	Accuracy	Loss
Model 4			
Model 5			

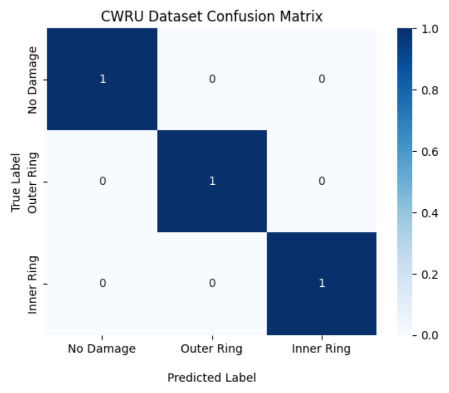
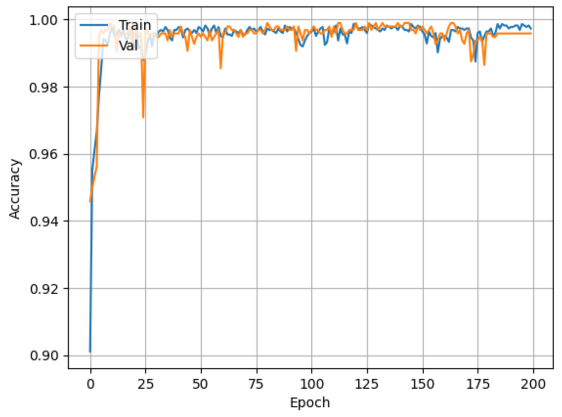
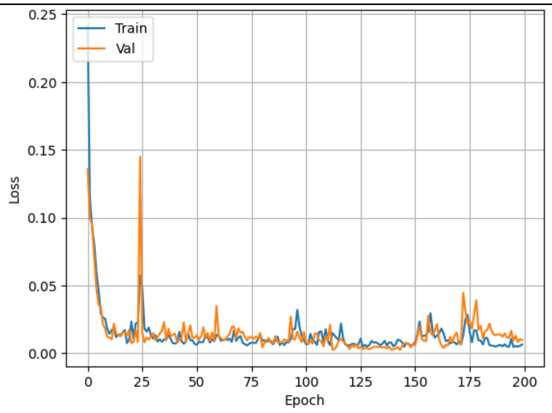
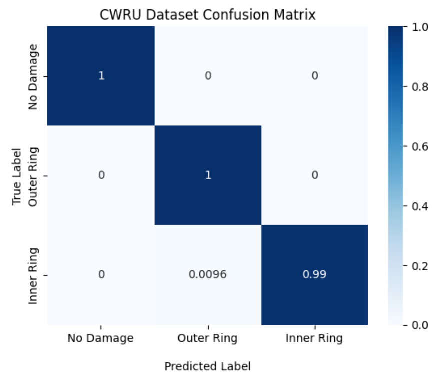
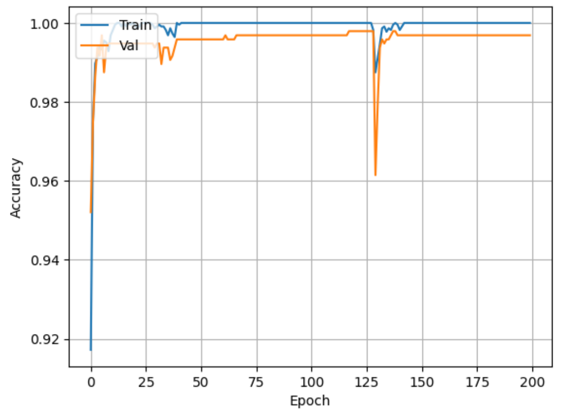
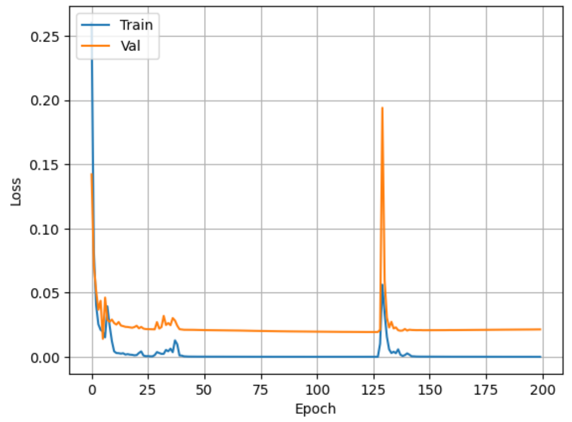
Model	**Confusion Matrix**	**Accuracy**	**Loss**
Model 6			
Model 7			

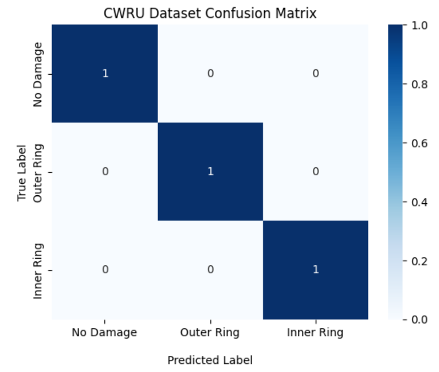
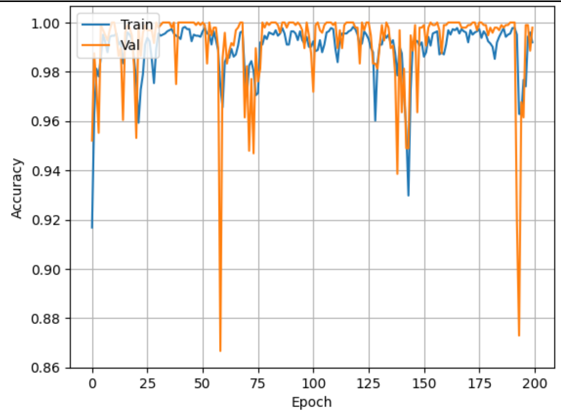
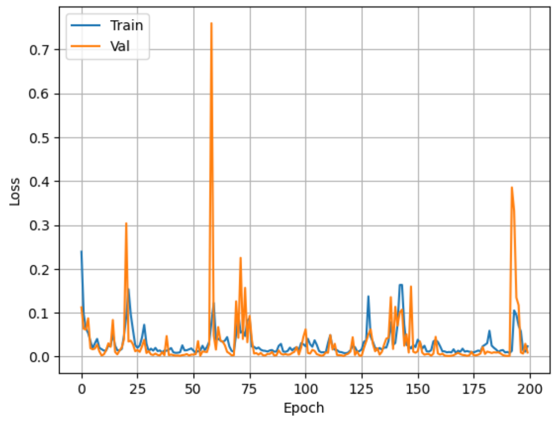
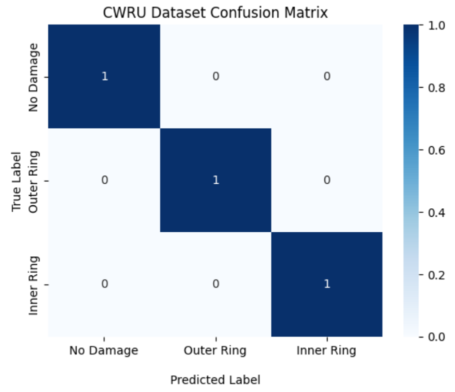
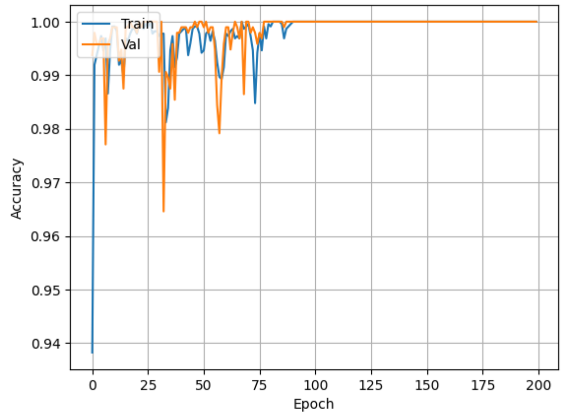
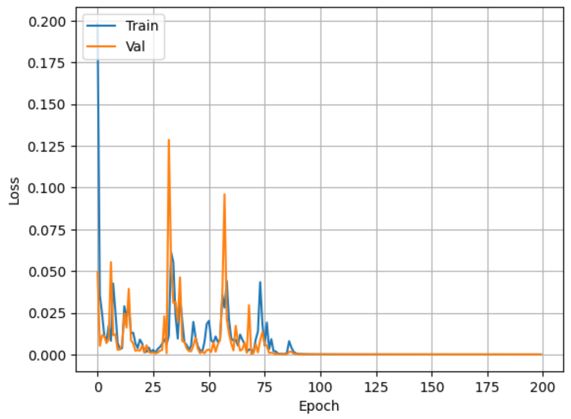
Model	Confusion Matrix	Accuracy	Loss
Model 8			

CWRU Results

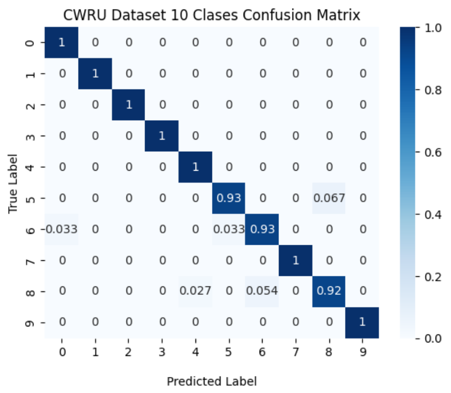
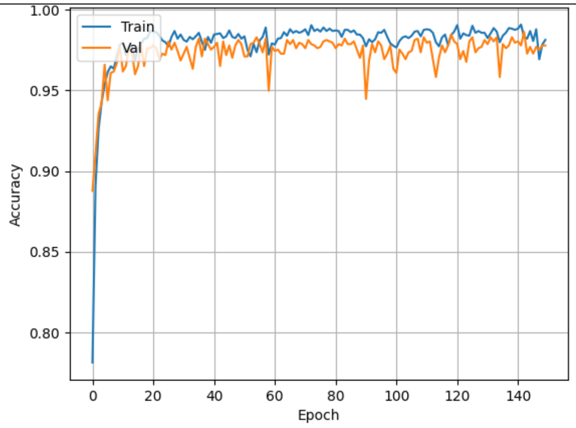
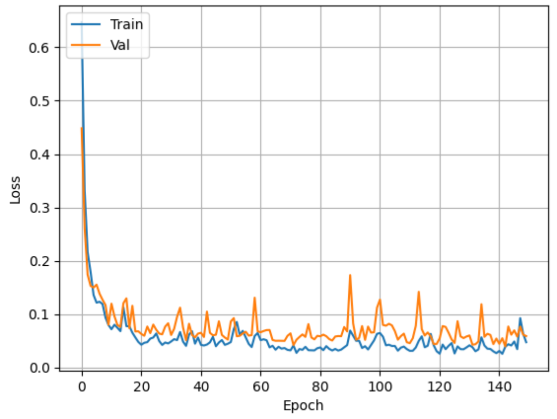
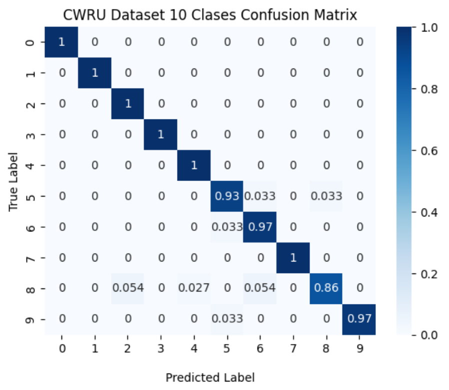
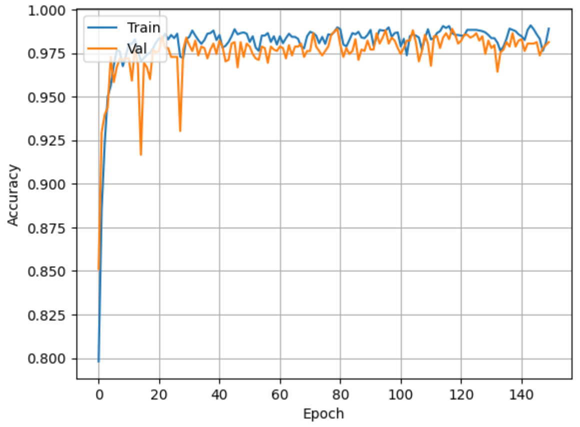
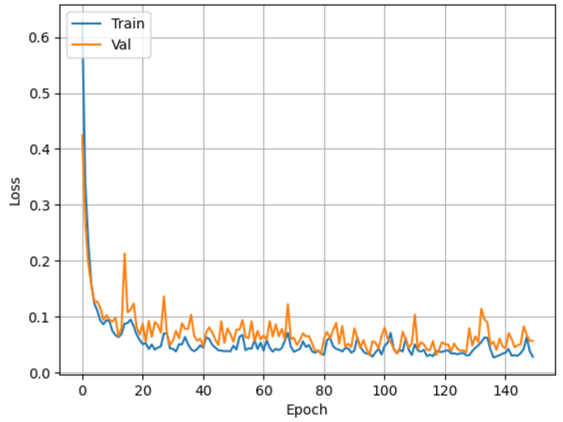
Model	Confusion Matrix	Accuracy	Loss
Model 1			
Model 2			

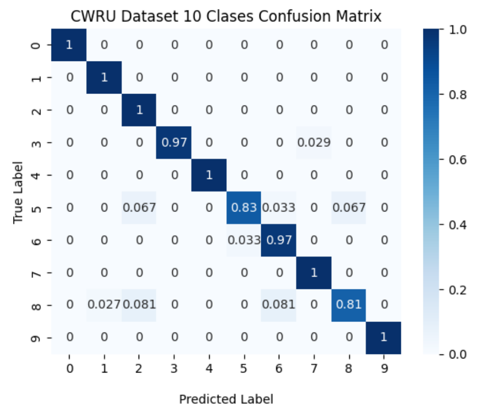
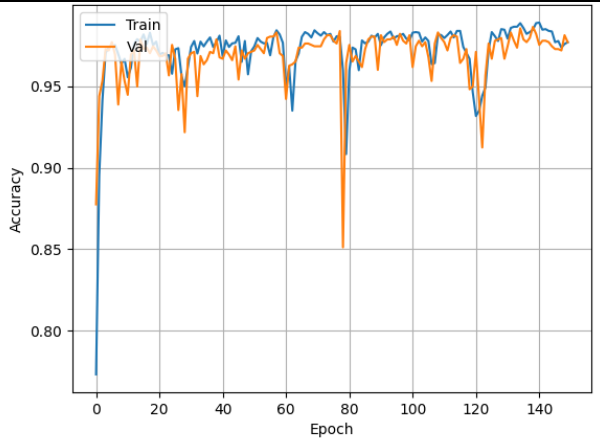
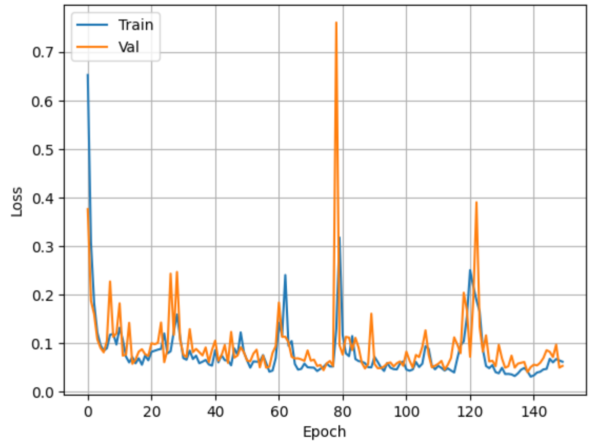
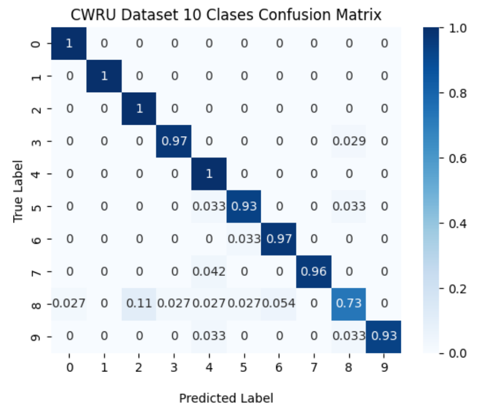
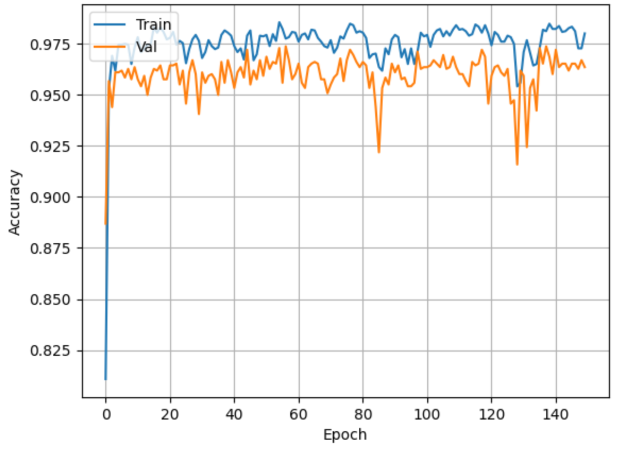
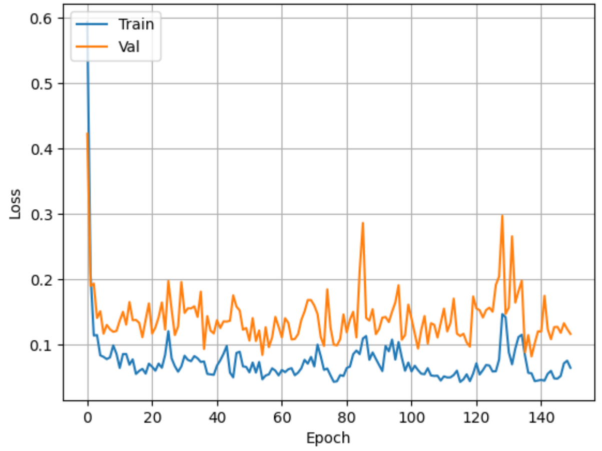
Model	**Confusion Matrix**	**Accuracy**	**Loss**
Model 3			
Model 4			

Model	**Confusion Matrix**	**Accuracy**	**Loss**
Model 5			
Model 6			

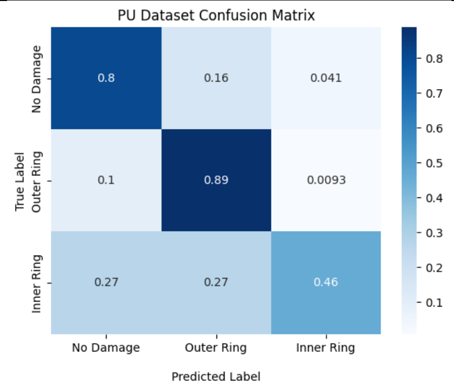
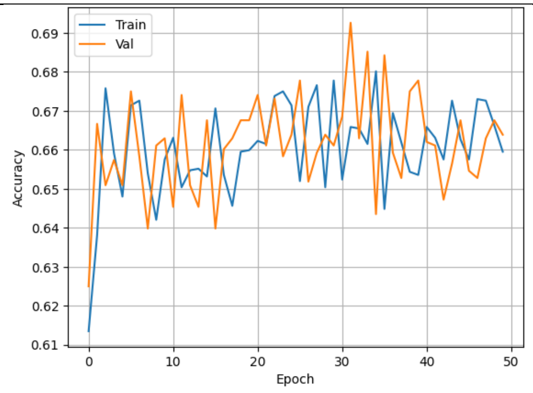
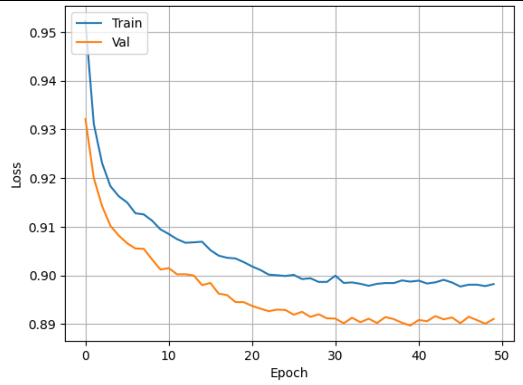
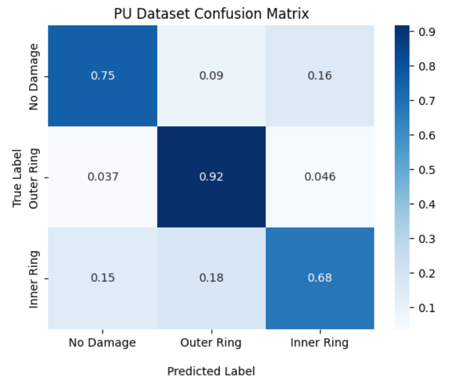
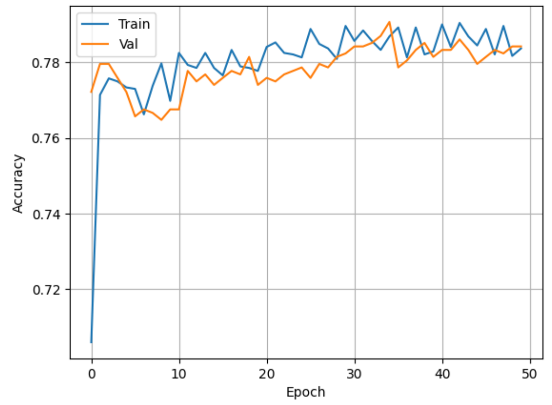
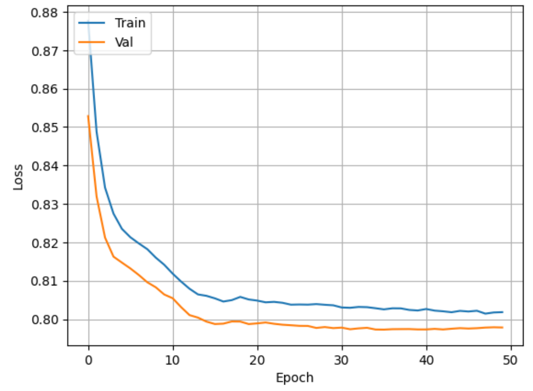
Model	Confusion Matrix	Accuracy	Loss
Model 7			
Model 8			

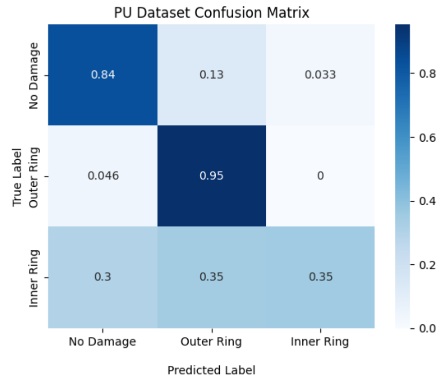
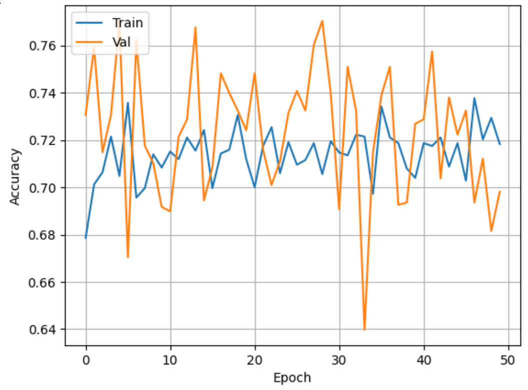
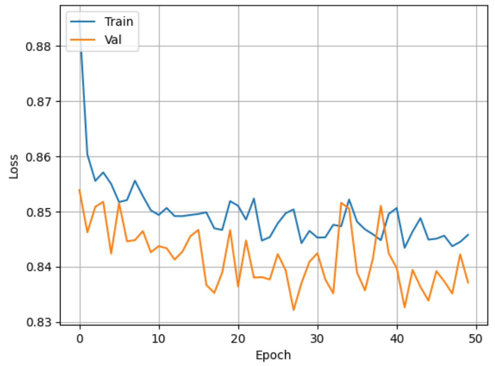
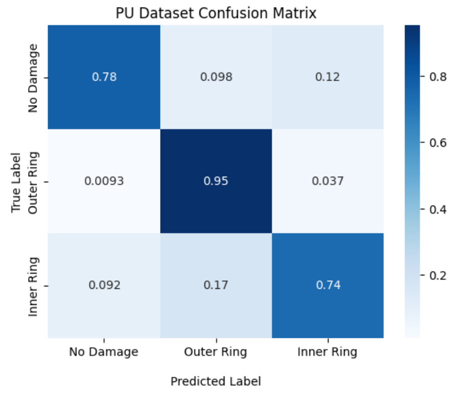
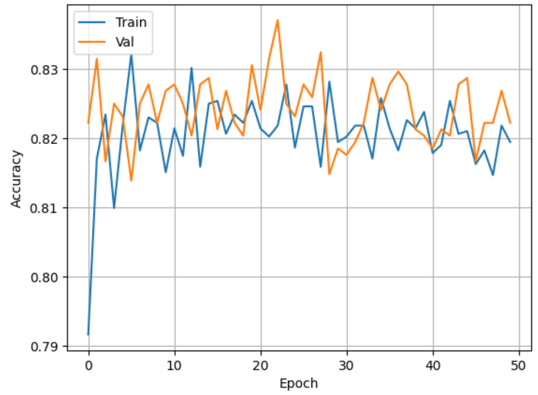
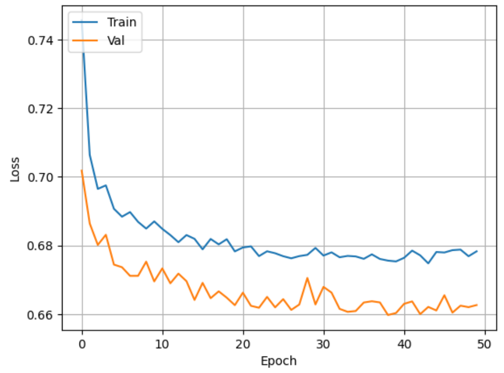
CWRU 10 Class Results

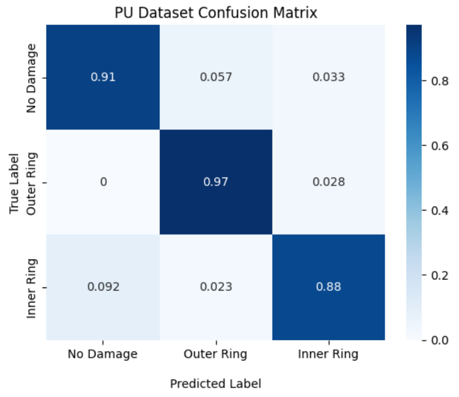
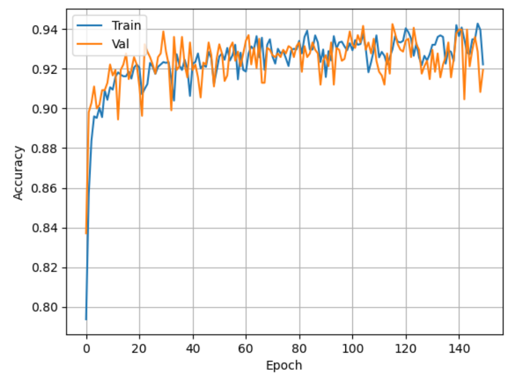
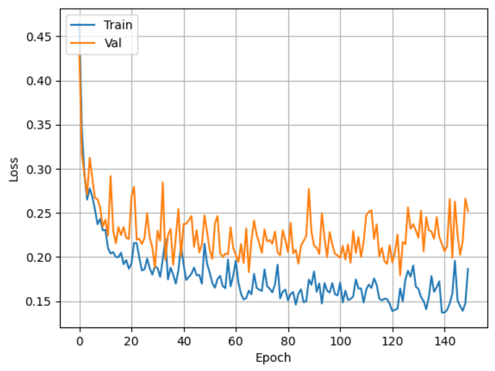
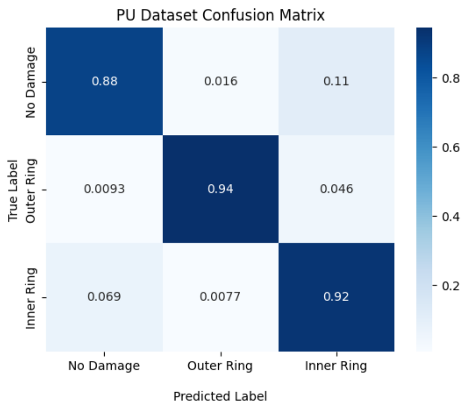
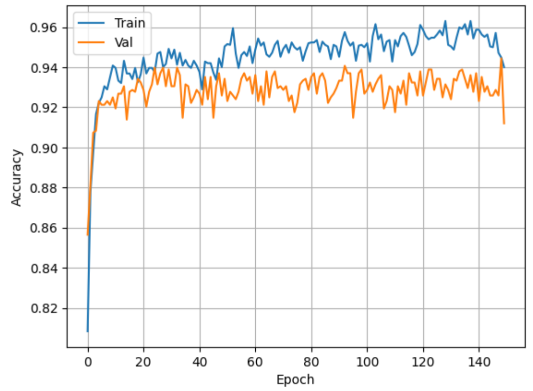
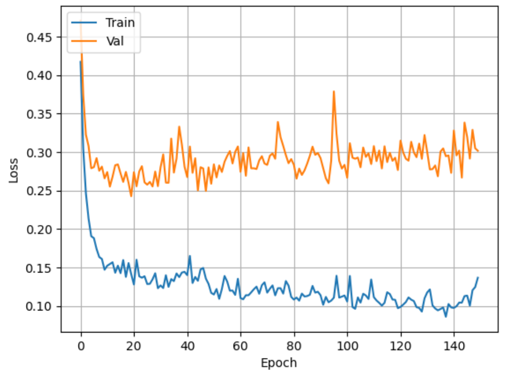
Model	Confusion Matrix	Accuracy	Loss
Model 5			
Model 6			

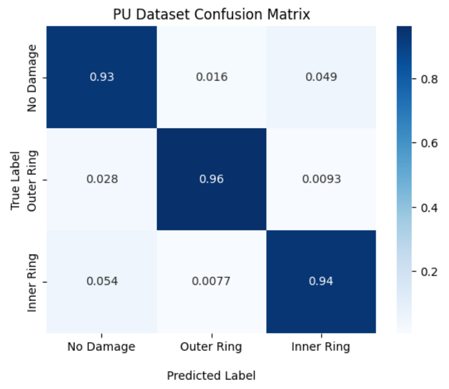
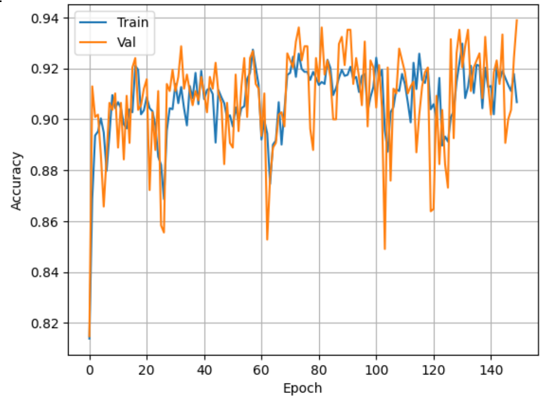
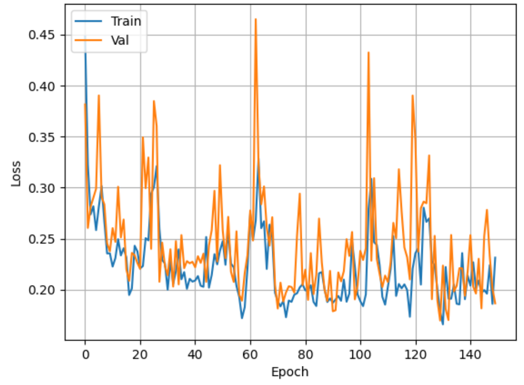
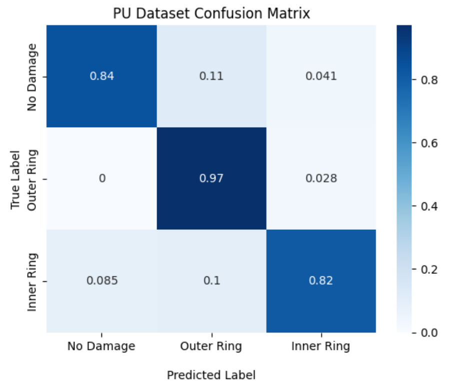
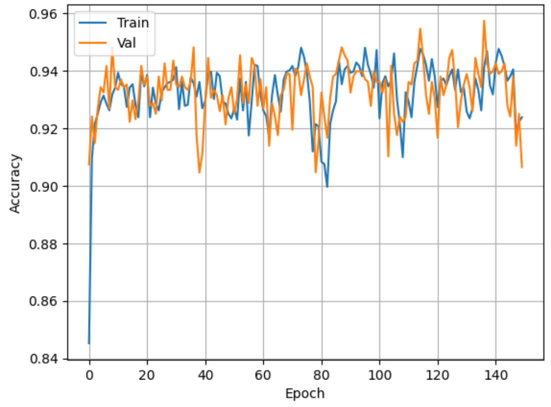
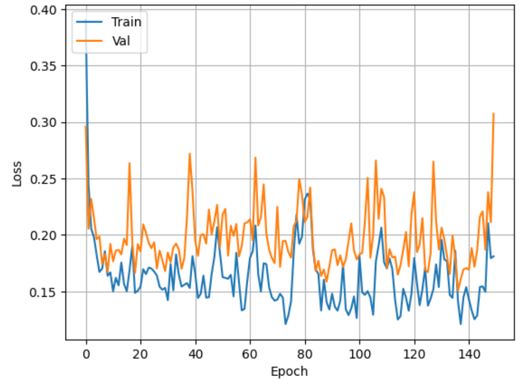
Model	Confusion Matrix	Accuracy	Loss
Model 7			
Model 8			

PU Results

Model	Confusion Matrix	Accuracy	Loss
Model 1			
Model 2			

Model	Confusion Matrix	Accuracy	Loss
Model 3			
Model 4			

Model	Confusion Matrix	Accuracy	Loss
Model 5			
Model 6			

Model	Confusion Matrix	Accuracy	Loss
Model 7			
Model 8			

APPENDIX C – RESULTS OF TRAINED MODELS

Model 1

In model 1, a ZFeatureMap input is used followed by RealAmplitude without using neural networks.

Table 4 - Results of Model 1

	MFPT				**CWRU**				**PU**			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	73.09	71.99	74.25	73.45	84.06	83.34	87.44	83.79	64,16	60.21	70.58	66.16
Run 2	75.73	75.08	76.55	75.94	83.43	82.67	86.77	83.15	64.44	61.79	70.38	66.20
Run 3	73.09	72.04	74.20	73.43	82.18	81.39	85.19	81.87	70.55	69.77	74.32	71.79
Mean	73.97	73.04	75.00	74.27	83.22	82.47	86.47	82.94	66.38	63.92	71.76	68.05

Source: The author (2023)

In general, the CWRU provides superior performance compared to the other two categories (MFPT and PU). It exhibits the highest average accuracy (83.22%), F1 score (82.47%), precision (86.47%), and recall (82.94%). This indicates that the model or approach used for the CWRU is effective in accurately classifying samples and has a higher generalization capability compared to the other categories. According to the figures observed in the annexes, the CWRU database appears to be more differentiable compared to the rest of the databases. This suggests that the CWRU dataset may be considered easier for classification purposes.

On the other hand, the MFPT category shows intermediate performance, with an average accuracy of 73.97%, F1 score of 73.04%, precision of 75.00%, and recall of 74.27%. While it does not reach the level of performance observed for the CWRU, it still outperforms the PU overall.

The PU exhibits the lowest performance among the three categories, with an average accuracy of 66.38%, F1 score of 63.92%, precision of 71.76%, and recall of 68.05%. These results suggest that the model or approach used for PU struggles to accurately classify samples and generalize to new instances.

Model 2

In model 2, a ZZFeatureMap input is used followed by RealAmplitude without using neural networks.

Table 5 - Results of Model 2

	MFPT				**CWRU**				**PU**			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	79.52	79.11	79.22	79.83	83.75	83.13	86.44	83.46	80.00	80.07	80.25	80.59
Run 2	79.82	79.61	79.54	80.08	83.75	83.09	86.69	83.46	77.77	77.84	77.90	78.40
Run 3	79.23	78.78	78.93	79.55	83.75	83.13	86.44	83.46	77.50	77.50	77.66	78.25
Mean	79.53	79.17	79.35	79.82	83.75	83.12	86.53	83.46	78.42	78.47	78.61	79.08

Source: The author (2023)

In terms of overall performance, the CWRU consistently demonstrates the highest results compared to the MFPT and PU. It has the highest average accuracy (83.75%), F1 score (83.12%), precision (86.53%), and recall (83.46%).

The MFPT category shows relatively strong performance, with an average accuracy of 79.53%, F1 score of 79.17%, precision of 79.35%, and recall of 79.82%. Although slightly lower than the CWRU category, it outperforms the PU category across all metrics.

On the other hand, the PU category exhibits the lowest performance among the three categories, with an average accuracy of 78.42%, F1 score of 78.47%, precision of 78.61%, and recall of 79.08%. While still competitive, it falls behind the CWRU and MFPT categories in terms of overall performance.

Model 3

In model 3, a ZFeatureMap input is used followed by Quantum Convolutional Network without using neural networks.

Table 6 - Results of model 3

	MFPT				CWRU				PU			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	78.07	77.43	78.69	78.30	85.62	84.87	90.06	85.39	65.83	64.35	71.89	67.66
Run 2	77.48	76.79	78.18	77.74	85.00	84.26	88.96	84.75	73.88	72.71	77.71	75.42
Run 3	77.19	77.29	78.26	77.31	85.31	84.56	89.50	85.07	69.72	67.65	74.88	71.45
Mean	77.58	77.17	78.38	77.88	85.31	84.56	89.51	85.07	69.81	68.24	74.83	71.51

Source: The author (2023)

In terms of overall performance, the CWRU category consistently demonstrates the highest results compared to the MFPT and PU categories. It has the highest average accuracy (85.31%), F1 score (84.56%), precision (89.51%), and recall (85.07%).

The MFPT category shows relatively strong performance, with an average accuracy of 77.58%, F1 score of 77.17%, precision of 78.38%, and recall of 77.88%. Although lower than the CWRU category, it outperforms the PU category across all metrics.

On the other hand, the PU category exhibits the lowest performance among the three categories, with an average accuracy of 69.81%, F1 score of 68.24%, precision of 74.83%, and recall of 71.51%. While still competitive, it falls behind the CWRU and MFPT categories in terms of overall performance.

Model 4

In model 4, a ZZFeatureMap input is used followed by Quantum Convolutional Network without using neural networks. The CWRU category consistently demonstrates the highest performance compared to the MFPT and PU categories. It has the highest average accuracy (98.22%), F1 score (98.22%), precision (98.34%), and recall (98.20%).

Table 7 - Results of Model 4

MFPT				CWRU				PU			
Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)

Run 1	85.96	86.23	87.14	86.08	98.12	98.12	98.24	98.09	83.05	83.05	83.29	83.76
Run 2	86.84	87.09	87.63	86.94	97.50	97.50	97.70	97.46	81.94	82.11	82.37	82.28
Run 3	86.54	86.76	87.33	86.72	99.06	99.05	99.09	99.04	81.66	81.68	82.20	82.36
Mean	86.45	86.69	87.37	86.58	98.22	98.22	98.34	98.20	82.22	82.28	82.62	82.80

Source: The author (2023)

The MFPT category shows strong performance, with an average accuracy of 86.45%, F1 score of 86.69%, precision of 87.37%, and recall of 86.58%. While lower than the CWRU category, it still outperforms the PU category across all metrics.

The PU category exhibits the lowest performance among the three categories, with an average accuracy of 82.22%, F1 score of 82.28%, precision of 82.62%, and recall of 82.80%. Although competitive, it falls behind the CWRU and MFPT categories in terms of overall performance.

Model 5

In model 5, a ZFeatureMap input is used followed by RealAmplitude using neural networks.

Table 8 - Results of Model 5

	MFPT				**CWRU**				**PU**			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	91.81	91.87	92.07	92.00	99.06	99.05	99.04	99.06	90.83	90.82	90.80	91.09
Run 2	91.81	91.81	92.29	91.69	100.0	100.0	100.0	100.0	93.05	93.07	93.07	92.95
Run 3	92.98	92.90	92.90	92.93	100.0	100.0	100.0	100.0	91.94	92.01	92.01	92.22
Mean	92.20	92.22	92.42	92.21	99.68	99.68	99.68	99.68	91.94	91.97	91.97	92.09

Source: The author (2023)

The CWRU category consistently demonstrates the highest performance compared to the MFPT and PU categories. It has the highest average accuracy (99.68%), F1 score (99.68%), precision (99.68%), and recall (99.68%).

The MFPT category shows strong performance, with an average accuracy of 92.20%, F1 score of 92.22%, precision of 92.42%, and recall of 92.21%. While lower than the CWRU category, it still outperforms the PU category across all metrics.

The PU category exhibits the lowest performance among the three categories, with an average accuracy of 91.94%, F1 score of 91.97%, precision of 91.97%, and recall of 92.09%. Although competitive, it falls behind the CWRU and MFPT categories in terms of overall performance.

Model 6

In model 6, a ZZFeatureMap input is used followed by RealAmplitude using neural networks.

Table 9 - Results of Model 6

	MFPT				**CWRU**				**PU**			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	92.39	92.31	92.57	92.57	99.68	99.68	99.68	99.68	91.38	91.54	91.67	91.48
Run 2	92.98	92.97	92.95	93.02	100.0	100.0	100.0	100.0	90.83	90.97	91.11	91.35
Run 3	92.98	92.93	92.90	93.02	99.68	99.68	99.68	99.68	91.38	91.62	91.85	91.48
Mean	92.78	92.74	92.81	92.87	99.79	99.78	99.78	99.78	91.20	91.38	91.44	91.44

Source: The author (2023)

The CWRU category consistently demonstrates the highest performance compared to the MFPT and PU categories. It has the highest average accuracy (99.79%), F1 score (99.78%), precision (99.78%), and recall (99.78%).

The MFPT category shows strong performance, with an average accuracy of 92.78%, F1 score of 92.74%, precision of 92.81%, and recall of 92.87%. While lower than the CWRU category, it still outperforms the PU category across all metrics.

The PU category exhibits the lowest performance among the three categories, with an average accuracy of 91.20%, F1 score of 91.38%, precision of 91.44%, and recall of 91.44%. Although competitive, it falls behind the CWRU and MFPT categories in terms of overall performance.

Model 7

In model 7, a ZFeatureMap input is used followed by Quantum Convolutional Network using neural networks. The CWRU category consistently demonstrates the highest performance across all metrics compared to the MFPT and PU categories. It has the highest average accuracy (99.79%), F1 score (99.78%), precision (99.78%), and recall (99.78%).

Table 10 - Results of Model 7

	MFPT				CWRU				PU			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	91.81	91.75	91.72	91.83	99.68	99.68	99.68	99.68	94.16	94.22	94.16	94.32
Run 2	92.39	92.85	92.80	92.33	99.68	99.68	99.68	99.68	93.05	93.20	93.15	93.31
Run 3	93.27	93.23	93.28	93.35	100.0	100.0	100.0	100.0	94.44	94.54	94.56	94.52
	92.49	92.49	92.60	92.50	99.79	99.78	99.78	99.78	93.88	93.99	93.96	94.05

Source: The author (2023)

The MFPT category shows competitive performance, with an average accuracy of 92.49%, F1 score of 92.49%, precision of 92.60%, and recall of 92.50%. Although lower than the CWRU category, it outperforms the PU category across all metrics.

The PU category exhibits the lowest performance among the three categories, with an average accuracy of 93.88%, F1 score of 93.99%, precision of 93.96%, and recall of 94.05%. While still strong, it falls behind the CWRU category in terms of overall performance.

Model 8

In model 8, a ZZFeatureMap input is used followed by Quantum Convolutional Network using neural networks.

Table 11 - Results of Model 8

	MFPT				CWRU				PU			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	90.35	90.43	90.83	90.58	100.0	100.0	100.0	100.0	90.55	90.65	91.01	90.77
Run 2	91.52	91.52	91.69	91.43	100.0	100.0	100.0	100.0	94.16	94.22	94.30	94.20
Run 3	92.69	92.69	92.72	92.76	100.0	100.0	100.0	100.0	87.22	87.22	87.62	87.72
Mean	91.52	91.55	91.75	91.59	100.0	100.0	100.0	100.0	90.64	90.70	90.98	90.90

Source: The author (2023)

The CWRU category consistently demonstrates perfect performance across all metrics compared to the MFPT and PU categories. It has an average accuracy, F1 score, precision, and recall of 100.0%.

The MFPT category shows strong performance, with an average accuracy of 91.52%, F1 score of 91.55%, precision of 91.75%, and recall of 91.59%. It performs well but falls slightly behind the CWRU category in terms of overall performance.

The PU category exhibits lower performance compared to the other categories, with an average accuracy of 90.64%, F1 score of 90.70%, precision of 90.98%, and recall of 90.90%. While still reasonably good, it lags behind the CWRU and MFPT categories.

Results of CWRU 10 Classes

In the following method, we employ the cream-of-the-crop models that exhibited superior performance over the ones previously showcased, in order to carry out a head-to-head comparison with (SOUTO et al., 2023).

Table 12 - Results of CWRU with 10 Classes on best models performance

	CWRU 10 Classes											
	Model 5				**Model 7**				**Model 8**			
	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
Run 1	97.70	97.27	97.29	97.37	97.70	97.21	97.21	97.44	93.36	92.46	92.73	92.84
Run 2	97.44	96.92	97.08	96.91	94.89	94.08	94.70	94.29	97.93	97.19	97.00	97.72
Run 3	97.70	97.26	97.16	97.44	96.93	96.31	96.30	96.50	96.93	96.45	96.61	96.52
Mean	97.61	97.15	97.17	97.24	96.51	95.87	96.07	96.80	95.83	95.35	95.45	95.69

Source: The author (2023)

Among the three models (Model 5, Model 7, Model 8), Model 5 demonstrates the highest overall performance. It achieves an average accuracy of 97.61%, an F1 score of 97.15%, a precision of 97.17%, and a recall of 97.24%. Model 7 exhibits slightly lower performance, with an average accuracy of 96.51%, an F1 score of 95.87%, a precision of 96.07%, and a recall of 96.80%. Model 8 performs the poorest among the three models, with an average accuracy of

95.83%, an F1 score of 95.35%, a precision of 95.45%, and a recall of 95.69%. In conclusion, based on the provided results, Model 5 surpasses Model 7 and Model 8 in terms of accuracy, F1 score, precision, and recall.

In (SOUTO et al., 2023), different configurations of machine learning circuits were compared on the CWRU 10 Classes dataset with 5 features. According to the provided results, no significant differences in performance were observed among the configurations with 1, 5, and 10 layers of circuits. Overall, all configurations achieved comparable results, with an average accuracy around 96% and F1-score and recall also close to 96%.

Table 13 – Best models selection CWRU in Souto research

	CWRU 10 Classes with 5 features											
	Ry, Rx, Rz				**VQE: Ry, Rx, Rz+Cz**				**VQE: Ry, Rx, Rz+iSwap**			
#Circuit Layer	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)	Acc. (%)	F1 (%)	Pres. (%)	Recall (%)
1	98.08	98.08	98.23	98.08	98.08	98.02	98.21	98.08	96.55	96.54	96.65	96.55
5	96.93	96.84	97.38	96.93	98.08	98.08	98.26	98.08	96.55	96.49	96.61	96.55
10	96.17	96.08	96.49	96.17	96.17	96.20	96.70	96.17	96.17	96.13	96.65	96.17

Source: Adaptation of “CWRU: ML and QML Accuracy, Precision, Recall and F1-Score results.” (SOUTO, et al., 2023)

In conclusion of both works, based on these results, it is true that one cannot confidently claim that one model is better than another in terms of performance. It is possible that both models are already in their optimal states, and further comparisons could be made by considering the running time. However, it should be noted that this thesis does not include such a comparison of running time.