Quantum Explainable Artificial Intelligence

George Bowden

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Master of Science, Quantum Technologies
of
University College London.

Department of Physics and Astronomy
University College London

August 25, 2022

I, George Bowden, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Classical machine learning algorithms such as Deep Neural Networks are difficult to interpret for a human due to the complex nature of the models. An Explainable AI (XAI) produces details or reasons to make its functioning clear or easy to understand. XAI methods such as Local Interpretable Model-Agnostic Explanations (LIME) have the advantage of being ”model agnostic”, which allows them to be used with an arbitrary ”black-box” ML tools. Quantum machine learning techniques have potential for improving upon classical AI techniques in a variety of fields, particularly in solving certain NP problems, however QML is as opaque as any black-box model due to its use of quantum mechanical properties in computations. Therefore it will also be necessary to be able to create interpretable explanations for the outputs of quantum machine learning models. In this project, we set out the state of the art in XAI and suggest possible techniques for use as Quantum XAI methods and demonstrate the usefulness of such methods on quantum computers. Our novel contribution is the application of the LIME algorithm to a QNN modeling the magnetisations of the Ising chain in 1-d.

Acknowledgements

Acknowledge all the things!

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Overview

Over the last two decades, many responsibilities have been given to AI and machine learning algorithms. To ensure that the output of these algorithms is trustworthy, an array of techniques to remove opacity from the decision-making process of machine learning models have been and continue to be developed. Explainable AI methods were first developed for Deep Neural networks [1] which, due to their large number of parameters, can be difficult for a human to interpret. There are a variety of methods available for classical supervised and unsupervised machine learning methods which can benefit from better explainability and a simpler way to explain the models outputs. Quantum machine learning is similarly opaque to humans due to the nature of its computation. Explainable AI methods will be important to help justify the use of quantum machine learning predictions when better and more noise resistant quantum computers are developed and are considered a promising avenue for improvement on classical data modelling techniques such as neural networks [2]. Quantum machine learning is in its infancy but has the advantage that much of the theory for classical machine learning can be altered and implemented with reasonable ease. Previous works have achieved accurate binary, and L-class classification using variational quantum algorithms on various data sets such as IRIS[3, 2] and handwritten digits [4]. In particular, quantum convolutional circuits have proven very successful [3]. However, there has thus far been little analysis of how trustworthy these results are, a consideration that will become more relevant as the use cases of QML increase and the available hardware improves. This project aims to suggest alternative XAI approaches that may provide better explanations for a quantum classifier.

In section two, I will discuss why we need Explainable AI, what the different families of XAI are and give examples of the types of methods that may be generalisable to Quantum Machine Learning.

Section three will be concerned with a discussion of specific techniques and examples of existing methods of various different types. Following this in section four, I will introduce Quantum machine learning's key concepts and the required context as to the current state of quantum machine learning. I will finish in section four with a discussion of the current state of Quantum-XAI, what possible next steps may be, and a brief presentation of my results so far.

1.1 Artificial Intelligence (AI)

1.1.1 Classical Machine Learning

Machine learning is an area of Mathematics and computer science where patterns are learned from data to obtain information and make predictions about the inputs [5]. Machine learning can be split up into three areas; supervised learning, unsupervised learning and reinforcement learning. In supervised learning, the input data will be labelled. The algorithm's task is to infer the relationship between inputs and outputs, usually by altering its parameters based on some prescription. Unsupervised learning describes problems where the data is not labelled, and the task of the machine is to find patterns in the data without being given any examples. A prevalent example of this is clustering. Finally, reinforcement learning refers to the process of rewarding and/or punishing a computer based on its performance in a "game". We reward the computer for a move that brings it closer to achieving the objective and punish it for mistakes. Reinforcement learning is a technique that works well when we need a computer to play a game with well-defined rules and outcomes.

The complexity of the patterns we expect ML algorithms to learn can be staggering, as demonstrated by the company DeepMind in their celebrated algorithms; AlphaGo [6], AlphaZero [7], AlphaFold [8] and the most impressive in my opinion AlphaStar [9]. AlphaStar is an AI that has achieved grandmaster rank in the Blizzard game StarCraft2. It is impressive in part because it achieved its results while being handicapped to the same actions per minute as a typical human grandmaster[1].

Despite the huge success that classical machine learning has had, many techniques such as DNNs, which are discussed above, suffer from issues involving extremely long training times and the requirement for huge volumes of data and limitations depending on the structure of the data. Specifically, DNNs are

[1] Unlike chess where the AI can look inhumanly far into the future.

better at analysing Euclidean data rather than non-Euclidean data. For this reason, novel techniques such as QML may be considered. Examples of algorithms where quantum computers may be advantageous include many graph-based routines such as quadratic unconstrained binary optimisations [10].

1.1.2 Classification

L-class classification is designed to infer the class label of an unseen data point $\underline{x} \in \mathbb{C}^N$ given a labelled dataset

$$D = \{(\underline{x}_1, y_1), \ldots, (\underline{x}_m, y_m)\} \subset \mathbb{C}^N \times \{0, 1, \ldots, L-1\}$$

A classification problem can be solved using the cost function

$$c(\theta) = \sum_{i=1}^{m} c\left(y_i, f\left(\underline{x}_i, \underline{\theta}\right)\right)$$

When f is the machine learning model defined by the set of parameters $\underline{\theta}$ and data $\underline{x}_i$ $c(a,b)$ is a discrimination function between a and b. for example least squares

$$c(a,b) = \|\underline{a} - \underline{b}\|^2$$

After training the class label for the unseen data point is $\tilde{x}$ is determined by $f(\tilde{x}, \tilde{\theta})$,

$$\tilde{\theta} := \text{argmin}_\theta C(\theta)$$

1.2 Explainable Artificial Intelligence (XAI)

The need for trustworthy AI models has never been as clear as it is today. AI systems help doctors, drive cars, run industrial processes, choose our music, and do countless other tasks that shape modern society. The importance of the work expected of AI is only likely to increase as time goes on, and as such, we need to be able to rely on the outputs of the AI systems and understand why they make the choices they do. Professionals are far more likely to use an AI if they understand the justifications given by the model. Explainable AI (XAI) is the name given to methods that attempt to identify causal relationships between AI model predictions and their input [11]. XAI methods were initially created to explain the opaque predictions of Deep neural networks (DNNs), which are almost entirely impossible to understand for any reasonable human due to the enormous amount of parameters that DNNs typically depend upon.

1.2.1 Interpretability and Explainability

Interpretability is the degree to which a human can understand the cause of a decision and predict a model's result [11]. Having models that are simple for a human to interpret are important because they connect humans to the problems they are responsible for. The goal of XAI is to provide valid explanations, have high fidelity and be simple enough to understand. Interpretability and fidelity are often mutually exclusive since a simple explanation may need to simplify a complex model. For example, a neural network that is simple enough for a human to read would not have much capacity for modelling a complex problem [12]. Conversely, a model with a large parameter set and complex algorithm could be highly accurate but make no sense to the average human. Indeed models need to be interpretable to the individuals who use them daily, individuals such as doctors who may not have a deep knowledge of machine learning. Thus XAI must provide simple, human-readable explanations that have high fidelity and can therefore be verified by a human with subject matter knowledge. Explanations can also be used to spot biases in models by expressing contradictions that the model has picked up in human-readable terms [1]. For example, if a system discriminates between different ethnicities, it would be helpful to detect such failures at the prediction level. Furthermore, XAI can help define the boundaries of the usefulness of an AI by demonstrating when its predictions are not justified. It is equally as important to know when not to trust a model as it is to know when one can trust it. We can consider interpretability locally and globally depending on the model. A local explanation is an explanation that well approximates a model in an arbitrary neighbourhood of a data point but not necessarily over the whole feature space. On the other hand, a global explanation is one that is faithful to the model over the entire feature space. If we try to explain a model globally, it may be that the explanation is too complicated. We may instead choose to represent the model locally. This is sensible because it allows us to consider explanation over a particular region with specific features constant. This may have the effect of making the explanation more straightforward to interpret for an equivalently complex explanation. More detail as to the state of the art of XAI can be found in [13].

1.3 Explainable Artificial Intelligence Methods Overview

I will now spend some time reviewing important cases of specific XAI methods that illustrate common concepts. Many XAI methods are motivated by deep neural networks; they are perhaps the most versatile form of machine learning because their architecture can be altered for specific problems such as image

recognition, data classification or regression (to list but a couple). The issue, however, is that outputs are set by an enormous number of parameters, which results in the model being a black box that takes an input, performs an incredibly complicated evaluation and outputs a solution [14]. Their complex nature means that humans are cut out completely, so if an error does occur, it is hard for us to know where it may happen or what may cause it. In some situations, this could be quite harmful. For example, incorrectly classifying a medical scan could result in lower-quality patient care but may occur due to a very convoluted issue in the model's feature space. On the other hand, there are so-called "glass box" methods that are highly interpretable without much additional manipulation [11]. For example, decision trees are, by definition, explainable since each criterion used by the model is a ready-made explanation.

1.3.1 Visualisation Methods

Visualisation methods relate the output of a model to an image by determining which parts of the data point contribute most to the probability of a particular class or prediction. [1]. Visualisation methods are best for image data when we would like the model to highlight the areas of interest to the model, for example, regions that the model is most sensitive to or areas with the highest activation for a particular prediction. Such methods determine feature relevance using gradient descent or by modifying the input to the model and noting the resulting difference in the prediction to determine the importance of the altered region. For example consider figure 1 from [12]. This particular explanation was created using LIME, where the predictions of a model were used to train a DNN. The pixels which corresponded to the particular classification were found by checking for super-pixels with a positive contribution to each class. This can be easily checked by zeroing out certain super-pixels.

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 1.1: Visual explanation for various labels taken from [12]. Each image shows the explanation for a certain label. For example the far right image is the set of pixels that are given as the explanation for the label "Labrador".

One such example is Deconvolution developed by Matthew D. Zeiler and Rob Fergus [1]. Initially, the idea was to discover higher-level features in image data in a convolutional network (which reduces

dimension at each convolutional layer). If each convolutional step is given by

$$A^l, s^l = maxpool(f(A^{l-1} * K^l) + b^l)$$

where A^l is the l^{th} layer, s^l is an index vector used to store the indices of the highest activations on each image region, K^l is a learned convolutional filter, b^l is a bias and $f()$ is an activation function. Then Deconvolution works in reverse by recreating the previous layer from the vector s^l and A^{l-1}. Note that max-pooling is normally a non-invertible operation, but using the index vector s, this is no longer the case. So each preceding layer is given by

$$A^{l-1} = unpool(f((A^l - b^l) * K^{l^T}), s^l))$$

Repeating this process back to the original pixel layer will recreate the most important features and hence give one an idea of where the prediction came from. This is just one example of a visual approach to image data, but many others work using backpropagation, or some version of gradient descent or perturbation [1].

1.3.2 Model Distillation

A second approach is called model distillation and refers to explanations applied post hoc to the model. Insights from the DNN are distilled into more useful representations for the user. This is done by training a second model on the output of the first, where the second model is more straightforward for a human to interpret. [11] Not all distillations are interpretable. For example, in [15] a distilled DNN is trained from an ensemble of methods to develop more accurate predictions rather than make them explainable. The level of interpretability is determined by the choice of the explanation model. One example of model distillation is Locally Interpretable Model-Agnostic Explanations (LIME) [12], which uses the output of the predictive model trained by the developer to assess the appropriateness of local models denoted by g. The cost function used to select this model is

$$\mathscr{L}(f, g, \pi_x) = \sum_{z,z'} \pi_x(z)(f(z) - g(z'))^2$$

Where $g(z)$ is the local model around the point z, $f(z)$ is the output of an arbitrary classifier we want to explain acting on the point z, and $\pi_x(z)$ is a weighting that depends on the distance between Z and

z' where z' is called an interpretable representation of the vector z. The nature of z' will depend on the choice of explainable model.

The algorithm that LIME uses is as follows.

1. Sample around a chosen point that we may want to explain.

2. Classify each of the sampled points using the predictor we wish to explain.

3. Calculate the distance between the point we wish to explain and all the sample points according to some chosen metric π_x.

4. Consider a family of linear models[2] and use $\mathscr{L}(f,g,\pi_x)$ to evaluate the cost of each function. The contribution is weighted by the distance of the explained point to the sample point to ensure local explainability is prioritised.

5. One can add a term to $\mathscr{L}(f,g,\pi_x)$ to minimise the number of features in the local explainer. For a tabular problem this will be a regularisation term given by $\Omega(g)$.

In step 5, we regularise the complexity of an explanation by associating a cost to the model that increases with the complexity. Complexity could be the number of parameters used or the number of features. this is done using the cost function

$$\mathscr{L}(f,g,\pi_x) = \sum_{z,z'} \pi_x(z)(f(z) - g(z'))^2 + \Omega(g)$$

Where $\Omega(g)$ is the complexity of g. For the latter sections $\Omega(g)$ can be taken to be a regularisation term that alters the number of features used to make the prediction. Intuitively if the model is less complex, it is easier for a human to interpret. LIME is a model agnostic explanation which means that it can be applied to any black-box model post hoc. The issue with LIME comes from using a trained model to select explanations; if LIME is trained on a region with limited training samples, then it will likely produce useless explanations. In later sections we use LIME to analyses both classical and quantum dataset using a local linear lasso regression as the explainable model g, the interpretable feature z' will vary depending on the exact problem and will be discussed at the time and the Gaussian distance kernel is used for π_x.

[2]Linear models are generally easier to interpret and are used by the source [12], but depending on the complexity and non-linear model is also a valid choice of explanation

1.3.3 Intrinsic Methods

So far, all the methods discussed have been post hoc, applied to a model after it is trained to justify the models' behaviour. Intrinsic explanations are produced by the model along with the output. A simple example is that of a decision tree where the branches are intrinsic to the model but also constitute explanations. This would be preferable to post hoc methods since they would be able to train correct parameters and correct explanations. The idea is that an explanation is fed back to the model in training time, evaluated, and the result is used to alter the model by shifting its decision landscape or its predictions [13]. Intrinsic methods are slightly more difficult to talk about since they are most commonly used to generate explanations for text data [11]. There are two types of intrinsic XAI techniques. The first is called "joint training". In Joint training, the explanation for the model is trained along with the model itself. For example, if the task is to optimise an objective function, then the explanation could be added to the objective function throughout the training process. This approach is commonly used in translation problems as in [16] and in natural language processing due to the fact that it is possible to generate natural language explanations such as a description or synonymous sentence. These explanations are not always consistent, as discussed in [17]. The second type of intrinsic method is called "attention mechanisms". Attention mechanisms are used to help a neural network keep track of important features in order to improve model performance. Because the tracked parameters are supposedly the most important, they provide an intrinsic explanation as to why the model has made the prediction it has. In a neural network, the attention mechanisms can record the relative importance (using the activation of each layer with respect to the feature) at each layer of the neural network. By doing this, we can see which features the neural network finds most important and use that to aid downstream[3] processing [18].

1.3.4 Counterfactual Explanation

A counterfactual is a conditional statement that demonstrates the effect of a change of face, for example: "If kangaroos had no tails, they would topple over". A definition of counterfactual explanations given by [19] is that a counterfactual explanation of a prediction describes the smallest change to the feature values that changes the prediction to a different output. The idea here is simple: Does the prediction change if we provide different facts? Suppose that we are given a data point with its predictive output. We can alter one (or more) of the features and see what the change in classification is [20]. This constitutes an explanation since we can quiz the model with different contexts and make sure that its predictions change

[3] in subsequent layers

in the way one might expect. Suppose Alice is rejected for a loan due to a lack of income. It might be the case that she needs to increase her salary by £10k. While an increase of £50k may do the job, it is easier to seek the minimal change required. Counterfactual explanations are meant to find such minimal increases. In [21] counterfactual explanations are given as a minimisation problem which minimises the distance between the data point x and the counterfactual data point x' characterised by

$$\mathscr{L} = \text{argmin}_{x'} d(x, x')$$

subject to

$$f\left(x'\right) = y'$$

where the latter is the requirement that the classification class is unchanged by the change to the counterfactual explanation x'. Both conditions can be expressed as

$$\mathscr{L} = \arg\min_{x'} \max_{\lambda} \lambda \left(f\left(x'\right) - y'\right)^2 + d\left(x, x'\right)$$

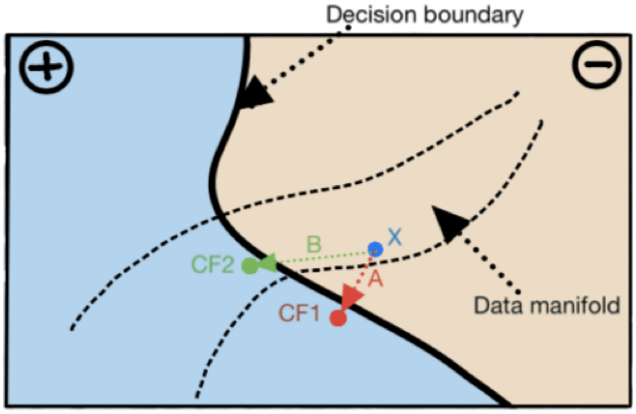

Figure 1.2: Two possible paths for a data point (shown in blue), originally classified in the negative class, to cross the decision boundary. The endpoints of both the paths (shown in red and green) are valid counterfactuals for the original point. Note that the red path is the shortest, whereas the green path adheres closely to the manifold of the training data but is longer [20].

Counterfactual explanations also need to be ”actionable” [20], variables such as race, gender, height etc that cannot be changed should not be considered. Thus we should only draw data points from the set $\mathscr{A}$ which contains vector space components where immutable components are not changed, now giving

us the cost function

$$\mathscr{L} = \arg\min_{x' \in \mathscr{A}} \max_{\lambda} \lambda \left(f\left(x'\right) - y' \right)^2 + d\left(x, x'\right)$$

Further to this, there are several more issues to consider such as:

- Sparsity - The similarity between two data points should be minimised by changing as few features as possible. This can be done by adding the penalty term $g(x - x')$ where g is the $L0/L1$ norm.

- Data manifold closeness - It is undesirable for the explanation data point to be totally unlike any before seen data since the model may not be properly trained for such a data point, or it may lead to an unrealistic scenario. To account for this, we include the penalty term $l(\mathscr{X}, x')$ which is the minimal norm between the explanation data point x' and all data points in the training data set $\mathscr{X}$.

- Causality - There are various other effects that can be described as causal. If a feature changes, one should also consider its dependencies since in general features are not independent of each other. How exactly this is done will depend on the individual case.

Considering all the effects described above, we are left with the objective function [20]

$$\arg\min_{x' \in \mathscr{A}} \max_{\lambda} \lambda \left(f\left(x'\right) - y' \right)^2 + d\left(x, x'\right) + g\left(x' - x\right) + l\left(x'; \mathscr{X}\right)$$

Counterfactual explanations could be used for applicants seeking loans, jobs or other services and opportunities to help them understand what they need to do to change the outcome or why the given outcome was chosen.

1.3.5 Model Agnostic Explanation

Some XAI methods for explanations using properties of the models' methods, such as Deconvolution, rely on the use of Convolutional neural networks for classification. Model agnostic explainers use the predictions from the model instead of the model properties to form interpretable explanations. Such explanations are universal but necessarily after the fact explanations, limiting their usefulness since, unlike intrinsic explanations, they are not generated with the predictions and hence are unvalidated.

1.4 Quantum Machine Learning

1.4.1 Quantum Computing and Quantum Machine Learning

A qubit is a bit of quantum information expressed as an arbitrary superposition of the orthogonal computational basis states.

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \in \mathscr{H}$$

such that

$$|\alpha|^2 + |\beta|^2 = 1$$

Where $|\psi\rangle$ represents a single qubit and H is a Hilbert space. The idea of a quantum computer was first suggested by Richard Feynman in his paper [22] where he argues that the best way to simulate a probabilistic system such as a quantum system is by using a naively probabilistic computer, such as a quantum computer. The notion of a quantum computer has developed since Feynman's paper, and we now have well-defined computer architectures and working NISQ [23] devices on which computations can be performed. Computations on NISQ devices are very vulnerable to noise and currently have a large probability of an error occurring. Current quantum computers [4] are formed by single and double qubit unitary operations, which are combined to form a quantum circuit. Quantum algorithms, and in

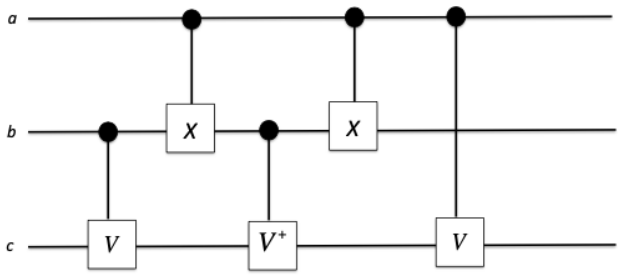

Figure 1.3: A quantum circuit diagram. Each horizontal line represents a single qubit, and each vertical line represents a controlled operation. Operations are performed left to right.

our case, neural networks, are designs using quantum circuit diagrams where qubits are represented by lines and unitary operations by boxes. The specific ansatz for each problem will vary dramatically. In quantum machine learning, there are several choices suggested, such as a convolutional ansatz [3] which have many similarities to classical convolutional networks as surveyed in [24]. The demonstrated

[4] not including quantum annealers

ansatz in [2] consists of a set of two-qubit unitary operations in the first layer followed by a pooling layer analogous to traditional convolutional neural networks. However, in CNNs, the pooling typically includes an entangling operation such as a controlled unitary [3]. Although there are similarities between the CNN and quantum convolutional neural networks QCNN there are also differences. CNNs can select specific patterns from the data using kernels and filters that are sensitive to particular patterns. The analogues QCNN, however, cannot. They get their name from the fact that sequential layers reduce in dimension by a factor of one half, projecting the feature space onto successively smaller dimensions each time. Another interesting example of a computational circuit ansatz is given in [25] where the authors embed the label of a classification problem as a qubit in the quantum circuit. Quantum machine learning is an area of research concerned with the development and production of machine learning techniques that can leverage the unique quirks of quantum mechanics to provide a computational or algorithmic advantage. [5] The features that distinguish quantum computing from regular algorithms and machine learning techniques stem from the nature of the basic unit of information, the qubit [5]. States in quantum mechanics live in a Hilbert space formed by the graded algebra generated by the tensor product of n qubits.

$$\mathscr{H} = \bigoplus_{i=0}^{\infty} \bigotimes_{j=0}^{i} V_j$$

Where V is a linear space with a field or skew ring structure, qubits can also be entangled. By entanglement, we mean that given a system of qubits, it is possible to encode correlation between various subsystems of the qubits. These correlations can solve problems that have proved untenable for classical algorithms. An example of this is prime factorisation done by Shor's algorithm [26]. Qubits are transformed using 2^n dimensional unitary operators where n is the number of qubits, for example, the Pauli rotation gates, which are all single-qubit rotation gates.

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Each gate encodes rotations of π about the x, y, z axes, respectively, which are single-qubit gates. Higher-dimensional unitary gates (in the same way as states) can be expressed as tensor products of lower-dimensional local qubits operations. In quantum computing, algorithms are executed by a circuit, which

[5] or qudit/qutrit for higher-level systems

itself can be expressed as an n qubit unitary gate. Quantum machine learning and quantum algorithms try to leverage the above properties independently of classical computation or as quantum-classical ensembles [27] to perform information processing tasks. Due to the probabilistic nature of quantum mechanics, experiments are repeated multiple times to find the actual result, which is a measurement of some set of qubits on a chosen basis.

1.4.2 Entanglement

Entanglement is on of the most exciting properties of Quantum Mechanics and described a probabilistic correlation between spatially separated qubits. An entangled state is one that cannot be written as tensor product of pure states,

$$|\psi\rangle \neq |\psi_A\rangle \otimes |\psi_B\rangle$$

Examples of entangled states are the Bell states,

$$\begin{aligned}
\left|\Phi^+\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B\right) \\
\left|\Phi^-\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle_A \otimes |0\rangle_B - |1\rangle_A \otimes |1\rangle_B\right) \\
\left|\Psi^+\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle_A \otimes |1\rangle_B + |1\rangle_A \otimes |0\rangle_B\right) \\
\left|\Psi^-\right\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle_A \otimes |1\rangle_B - |1\rangle_A \otimes |0\rangle_B\right)
\end{aligned}$$

These bell states an maximally entangled states. Each state can be generated by a simple circuit such as the one given in figure 3.

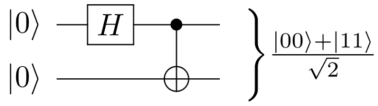

Figure 1.4: The quantum circuit required to produce the $|\Phi^+\rangle$ Bell state.

In general states written as probability distributions over product states are not entangled, such as the state

$$|\psi\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle = \left(\frac{|1\rangle + |0\rangle}{\sqrt{2}}\right)^{\otimes 2} = |+\rangle|+\rangle$$

is not entangled. However if we consider the state

$$|\phi\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle)$$

is an entangled state since it cannot be factorised. Using an entangling gate it is possible to entangle two qubits that were previously unentangled. Suppose we apply a controlled-Z gate to $|\psi\rangle$, we would get

$$cz|\psi\rangle = cz\frac{1}{2}(|00\rangle + |01\rangle) + (|10\rangle + |11\rangle)$$

$$= \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle - |11\rangle) = \frac{1}{2}(|0\rangle|+\rangle + |1\rangle|-\rangle) \text{ is an entangled state.}$$

Hence, $cz|\psi\rangle = |\phi\rangle)$ is entangled, thus cz is an entangling gate. Entanglement can be used to obtain a variety of non-trivial results in quantum information such as quantum non-locality which refers to correlations in space-like separated qubits and super dense coding which allows for dense information encoding. In quantum computing entangling gates are used to share information between qubits and is responsible for the non-trivial information exchange. There are a number of entangling gates on n qubits such as toffoli, CNOT and the controlled rotation gates.

1.4.3 Quantum Advantage

Quantum computers can theoretically eclipse the effectiveness of classical computers in specific problems. For example, factorisation in Shor's algorithm [26]. However, the case for quantum supremacy over an extensive library of problems is not trivial. Quantum supremacy can be declared when a quantum computer is able to surpass a classical algorithm either in speed or the or the quality of the solutions [6], for example complete a task in a time superior to that of a classical algorithm. The first example of quantum supremacy was by Google in 2019 [28] but many more are awaiting suitable hardware with higher numbers of fault tolerant logical qubits [7]. One of the first demonstrated algorithms was Grover's algorithm [30] which is a quantum search algorithm, and then others followed, such as Deutsch and Deutsch-Joza [31]. Due to the infancy of the research area and the weakness of available hardware, there is considerable effort in hybrid techniques that would use quantum computers for the parts they can do well and supplement them with classical computers. Such algorithms include quantum approximation optimisation algorithm (QAOA), which is a general-purpose algorithm for combinatorial optimisation problems [32], and variational quantum eigensolvers (VQEs), both of which have broad literature.

[6] Such as a true global minimum for an objective function or a higher probability of finding a solution etc.

[7] However, it was on a highly specific problem defined in order to demonstrate that quantum computers can outperform classical computers in at least some problems [29].

1.4.4 Noisy-Intermediate Scale Quantum Devices

At present, we are limited to Noisy-Intermediate Scale Quantum computers (NISQ) which are not fault-tolerant and have low coherence times [23], which is a huge issue to overcome for practical quantum computation that requires either a large number of qubits (large circuit width) or a large number of gates (large circuit depth). This is because a large number of qubits increases the probability of an error, and a deep circuit increases computation time giving more time for an error to occur. Often the result of this is that the qubits will not remain coherent for long enough to complete an algorithm. These devices are not adequate for many theoretical uses of quantum computers due to the high error rate and unreliability. These computers generally only allow for nearest neighbour connectivity, limiting the number of available computational qubits that can be used without quantum teleportation. This also creates another opportunity for noisy quantum channels to introduce more qubit errors. It is possible to use swap gates to improve the connectivity issue, but these operations are expensive to perform and could add to coherence. There are various architectures of quantum computers available without referring to the form of the qubit used. Quantum computing typically refers to gate based quantum computing which is the standard form implemented by Google and IBM, among others [28, 33]. The second type is a Quantum annealer [34] which is a slightly less flexible form of quantum computing device that is adiabatic. It cannot function as an arbitrary unitary evolution but is very useful for optimising solutions to problems by searching over a space and finding a minimum.

1.4.5 Quantum Optimisation

Two things can be meant by quantum optimisation. The first is using a quantum algorithm to find an optimal solution for an optimisation problem. The second meaning is a method to obtain the parameters for a Quantum Neural network or parameters quantum circuit as in [27]. There are currently either classical or classical-quantum hybrid methods available. One of the reasons for this is that one cannot feed a qubit back to interact with itself as would be possible for a classical situation. Due to the challenges outlined above and the fact that VQAs [8] are such good candidates for use on near term quantum computers, we would like to have a better way to train these circuits. Suggested methods include Quantum Approximate Optimisation Algorithm, Methods to optimise the quantum circuit parameters rather than explicitly solving the problem[9] are also important. As mentioned, the parameter space is exponentially difficult to train and

[8] A family of functions which include QNNs
[9] Like QAOA

is further complicated by the existence of the barren plateau problem, [35] suggests a method of gradient computation on a quantum circuit that requires $poly - log(d)$ queries where d is the dimension of the Hilbert space. [36] gives a detailed account of modern methods as well as suggestions for various oracles that may be used to query the quantum circuit using ancillary qubits.

1.4.6 Variational Quantum Algorithms (VQA)

So far, I have introduced quantum computers and stated that we wish to search of a computational advantage using such technology. Due to the early phases of quantum computation which we are currently experiencing it can be difficult to find task at which a quantum computer demonstrates a concrete advantage. The noisy nature of near term quantum computers hamstrings efforts to produce robust quantum advantage in many situations due to the high error rate and large overhead required to correct errors. So, what can we do? Variational quantum algorithms or VQAs are a method that brings us a step closer to useful quantum computers for everyday tasks [37] [10]. VQAs encompass a range of different functions, such as finding the ground state of a quantum system or being used as a quantum classifier. The general idea is that a circuit with some ansatz is chosen and parameterised by a set of rotation angles that define the unitary operators to be used. The parameters are optimised with respect to an objective function. For example, when a VQA is used to find the ground state of a Hamiltonian, we refer to it as a Variational Quantum Eigensolver. In this case, the objective function is a Hamiltonian. Classical optimisation techniques can be used to train a VQA, making them ideal as a quantum analogue to machine learning. For any VQA to be useful, it needs to satisfy two conditions. First, we require that the chosen ansatz has sufficient expressive power to approximate the subject unitary model accurately. Secondly, it must be trainable in a reasonable time [11]. This is of course assuming that the error rate is low enough for it to be feasible at all. The second point generally provides the problem because an expressive ansatz often has a larger parameter space to optimise over. Due to the fact that the circuit has to be computed multiple times to train its parameters, there is a certain amount of resilience to noise built into VQAs, making them ideal near term algorithms. For the same reason, it is possible to have deeper circuits in VQAs than deterministic ansatz. VQAs are often subject to barren plateaus [38], which are large regions in the feature space where the gradient is approximately flat. The result is vanishing gradients that make training

[10] For those of us who use AI every day, which is most people.

[11] Points one and two are related, expressing a larger Hilbert space fully can cause more difficulty optimising the circuit, but it is not necessarily true that the relationship works both ways. For example, it is possible to have a circuit that is trained slowly that is also not very expressive.

inefficient for large feature spaces. In [37] this phenomenon is demonstrated for an application of the max cut algorithm using classical optimisation and VQAs. The authors demonstrate that the training time of VQAs can scale exponentially with the size of the problem in a way analogous to the exponential growth of the Hilbert Space. One simple way to get around the barren plateau is a change in the choice of circuit ansatz. [3] points out that the use of convolutional ansatz can be trained more effectively than equivalent depth quantum circuits. Furthermore, convolutional ansatz has been demonstrated to be effective in many computational tasks such as classification [2].

1.4.7 Variational Quantum Eigensolver

An example of a common VQA is the Variational Quantum Eigensolver (VQE).The variational quantum eigen solver can be used to find the ground state of a Hamiltonian. In a sense it is the quantum version of the Rayleigh quotient :

$$E_0 \leq \frac{x^T H x}{x^T x} \, x \in \mathbb{R}^n$$

Instead we consider the ground energy of a Hamiltonian which is bound by

$$E_0 \leq \langle \psi | H | \psi \rangle \, , \, |\psi\rangle \in \mathscr{H}$$

where H a hermitian operator, in this case the system Hamiltonian which is the energy operator and $\mathscr{H}$ is a Hilbert space. Now consider the evolution by a unitary operator U, $UU^\dagger = 1$, the minimum VQE energy is given by

$$E_{VQE} = \min_{\underline{\theta}} \langle 0 | U^\dagger(\underline{\theta}) H U(\underline{\theta}) | 0 \rangle$$

where

$$|\psi(\underline{\theta})\rangle = U(\underline{\theta}) |0\rangle$$

and represents the energy of the state after evolution the the operator $U(\underline{\theta})$ where $\underline{\theta}$ is a vector of rotation angles that parameter is the quantum variational anzats. The operator U is an anzats that defines the search space for the algorithm. The VQE can only find solutions that are accessible to the operator U, a demonstration of why the choice of ansatz is so important. Since the Hamiltonian is a Hermitian operator

we can always write it as a sum of Pauli operators

$$H = \sum_i \omega_i P_i$$

Therefore the VQE energy can be written as

$$\begin{aligned} E_{VQE} &= \min_{\underline{\theta}} \langle 0| U^\dagger(\underline{\theta}) \sum_i \omega_i P_i U(\underline{\theta}) |0\rangle \\ &= \min_{\underline{\theta}} \sum_i \omega_i E_i(\underline{\theta}) \end{aligned}$$

where $E_i(\underline{\theta}) = \langle 0| U^\dagger(\underline{\theta}) P_i U(\underline{\theta}) |0\rangle$ and $P \in \{R_x, R_y, R_z\}$ is a Pauli rotation gate in the x, y, or z axis respectively. The parameter $\underline{\theta}$ is updated using a classical method, gradient descent. The updates are given by

$$\underline{\theta}_{i+1} = \underline{\theta}_i - \alpha \frac{\partial \langle H \rangle}{\partial \underline{\theta}_i}$$

where the derivative of the Hamiltonian expectation is written in terms of the Heisenberg picture using the parameter shift rule

$$\frac{\partial \langle H \rangle}{\partial \underline{\theta}_i} = \frac{1}{2} \left[\left\langle 0 \middle| H_H \left(\theta_{i \neq j}, \theta_j + \frac{\pi}{2} \right) \middle| 0 \right\rangle - \left\langle 0 \middle| H_H \left(\theta_i \neq j, \theta_j - \frac{\pi}{2} \right) \middle| 0 \right\rangle \right]$$

$$H_H = U^\dagger(\underline{\theta}) H U(\underline{\theta})$$

There are a few limitations of this technique:

- Deep circuits are currently difficult to execute due to exponential growth in computational time and decoherence.
- Limited by a large prefactor due to the requirement to resample from the circuit several times.
- VQEs are optimisation routines, their performance is closely related to the optimisation landscape of the particular problem.
- Barren Plateau is the name given to the flat regions that occur in the optimisation landscape which significantly hinder training. (Can be managed by choice of Ansatz)

- Unclear if error correction methods can make VQEs viable on NISQ devices due to enormous resource overhead. Logical Qubits are not efficient.

1.4.8 Data Encoding

Quantum computers are theoretically useful for simulating quantum systems due to their tensorial nature. Their computational basis forms a Hilbert space which grows exponentially with the number of qubits [citation needed], whereas bits grow linearly. The access to a larger dimensional space to transform is similar to that of a neural network but with an exponential decrease in the number of bits of information available. This, along with quantum properties such as superposition and entanglement, might provide practical advantages over classical machine learning methods. Data embedding can be done in two ways, before the computational circuit as in [3], and during the computational phase of the circuit as in [39]. In each case, there are a few well-defined methods. The simplest is amplitude embedding, where the classical data point is copied into a quantum state entry by entry, with all extra parts of the quantum state set to zero [12]. To list a few examples:

Qubit embedding is done by using a tensor product of Y rotation gates:

$$|\phi\rangle = \bigotimes_{i=1}^{N} (\cos\left(\frac{x_i}{2}\right)|0\rangle + \sin\left(\frac{x_i}{2}\right)|1\rangle) \tag{1.1}$$

and amplitude embedding by:

$$x \mapsto \frac{1}{|x|}\sum_i x_i |i\rangle \tag{1.2}$$

where $|i\rangle$ is the binary representation of the i^{th} computational basis state, and x_i is the i^{th}feature. Note that this embedding acts as the identity for quantum data represented as a state vector. There are several far more sophisticated approaches to manipulating data into a form ore compatible with the advantages of quantum computers. In [39] the authors demonstrate that a quantum model can be given as a partial Fourier series of the given data by using an appropriate encoding scheme. This is useful because quantum computers have access to rich frequency spectra and are therefore theoretically universal function approximators. The choice of the data encoding method is the connection point between the model and the data. Therefore, a good choice of encoding can have a significant impact on the expressibility of the

[12] Since the quantum state has dimension 2^n, if the classical data point doesn't also have this dimension then the quantum state may have components that are zero.

model.

1.4.9 Quantum Classification

Quantum L-class classification is designed to infer the class label of an unseen quantum state $\left|\psi_{\underline{x}}\right\rangle \in \mathscr{H}$ given a labelled data set

$$D = \{|\phi_{x_i}\rangle, y_i\}_{i=1}^{m} \subset \mathscr{H} \times \{0, 1, \cdots, L-1\}$$

The classification uses a cost function

$$c(\underline{\theta}) = \sum_{i=1}^{m} c\left(y_i - f\left(\left|\phi_{\underline{x}_i}\right\rangle\right)\right)$$

where $f(|\phi_{x_i}\rangle)$ is a POVM measurement of the form

$$f(|\phi_{x_i}\rangle) = \langle\phi|_{x_i} U^{\dagger}(\underline{\theta}) O U(\underline{\theta}) |\phi_{x_i}\rangle$$

where O is a unitary operator acting on $\mathscr{H}$. If we wish to use a quantum classifier on data arising from a classical source we are required to encode the data as described in the previous section.

1.4.10 Explainability in QML

At present, there is few pieces literature regarding the explainability of Quantum Machine Learning. However, many of the techniques used in classical machine learning have quantum analogues such as backpropagation [40, 41] or parameter importance (determined using the discussed parameter shift rule). Due to the fact that we cannot extract information from a quantum circuit without corrupting the process, we may sample from the circuit several times and use some form of quantum Monte Carlo [42, 43], e.g. variational Monte Carlo [44]. It is possible to apply model agnostic methods to quantum ML methods since QML methods produce an output that is indistinguishable from classical methods. As such, the simplest way to perform QXAI is to apply model agnostic methods. Model agnostic methods may demonstrate the difference in decision boundaries between classical and quantum algorithms, which would give a comparison between the outputs of each classifier. In my opinion, the two directions worth pursuing are quantum backpropagation methods [41] and intrinsic methods that depend explicitly on the circuit at training time in a manner similar to the label embedding in [25]. However, model agnostic approaches provide an interesting method to compare patterns of recognition of a quantum and classical model.

Chapter 2

Results and LIME Analysis

2.1 Methodology: IRIS and Penguins

This section will discuss the methods by which the classical and quantum data was prepared, classified and subsequently analysed using LIME. Following this, each dataset's classification results and explainability analysis will be given using the LIME technique developed by [12] on two classical data sets and one quantum dataset. The classical datasets are the IRIS and Penguins data sets [45, 46] which are standard choices for benchmarking the performance of classification algorithms. The quantum data set is the Ising model, one of the most theoretically rich toy models in physics, with applications in condensed matter, quantum computing and network search algorithms such as QAOA.

2.1.1 Classical Data

Two classical data sets have been considered, the IRIS and Penguins data sets [45, 46]. IRIS is a data set containing features and species labels for three species of flowers: 'Versicolor' [1], 'Virginica' and 'Setosa'. The data set has four features: 'Petal Length cm', 'Petal Width cm', 'Sepal Width cm' and 'Sepal Length cm', but the dataset can be accurately classified using the Sepal Length and Sepal Width, which we will see confirmed using LIME in the coming sections. The IRIS [45] dataset has 250 samples with four features. A test train split of 4:1 was used to ensure ample training data.

The second dataset considered was the Penguins dataset which contains information on the location and dimensions of three species of penguins. The Penguin dataset has 343 samples, each with eight features. However, the 'species' variable is used as the classification variable, and the categorical features

[1] Iris versicolor is also commonly known as the blue flag, harlequin blueflag, larger blue flag, northern blue flag, and poison flag, plus other variations of these names, and in Britain and Ireland as purple iris.

are not considered for the classification, leaving the features:'bill length mm', 'bill depth mm', 'flipper length mm', 'body mass g'. We consider trinary classification using a quantum neural network and a classical decision tree, the latter of which is of use as the benchmark since random forests are a well-understood and commonly used model. Once again, we use a test train split with a ratio of 4:1.

Examples of both data set can be found in appendix figures A.1 and A.2.

2.1.2 Classical Data Encoding

As discussed, we require that the classical data be transformed into a form suitable for a quantum computer, namely a normalised state vector. Each feature is encoded using a single qubit, which gives an easy-to-simulate 4-qubit system to optimise. Each 4-vector from both the penguins and IRIS data sets denoted $\underline{x}^i$ normalised such that all components in the data set sit in the region $[0, \frac{pi}{2}]$ is transformed by the map

$$\underline{x}^i \mapsto \bigotimes_{j=1}^{4} e^{-i\underline{x}^i_j \sigma_y} |0\rangle \,, \ \underline{x}^i_j \in [0, \frac{\pi}{2}]$$

which is the exponential form of the qubit embedding given above.

2.1.3 Classical Data Classification Circuit

The classification circuit is a four-qubit convolution ansatz inspired by [3]. The first layer is the embedding layer which encodes the classical data using the qubit embedding described above. The computational part of the circuit is given in 2.1. Each gate is parameterised by a parameter θ, which is different for

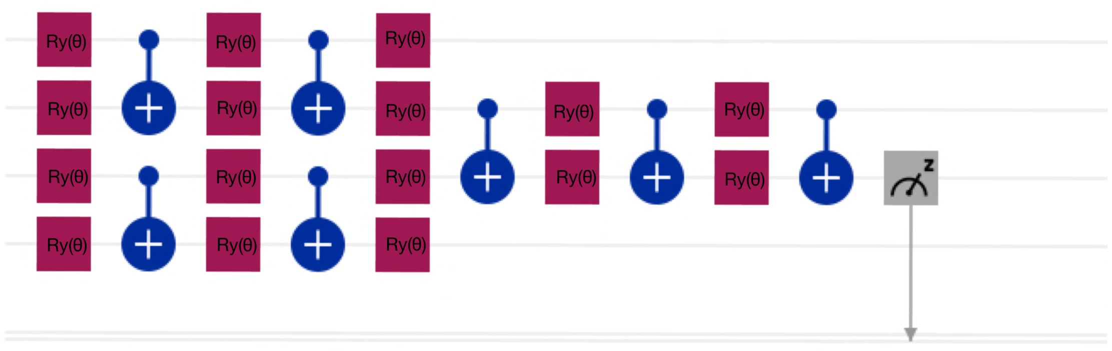

Figure 2.1: Visualisation of the chosen convolutional ansatz using the IBM package Qiskit. The circuit is able to perform trinary classification on the IRIS and Penguins datasets.

each rotation gate. Measurement $\langle 0| E^\dagger U^\dagger(\underline{\theta}) Z_i U(\underline{\theta}) E |0\rangle$ is done on the third qubit; the others are left as undisturbed wires. The quantum circuit is trained on the class labels as a function of the expectation. The

output of the classification function is given by

$$M = \frac{L}{2}(1 + \langle 0| E^{\dagger}U^{\dagger}(\underline{\theta})Z_i U(\underline{\theta})E |0\rangle) \in [0, L-1]$$

where L is the number of classes. Here $L = 3$ for the Iris and Penguins datasets. The circuit is trained such that M becomes equal to the class labels. The final classification is done by assigning the class based on the maximum probability defined below. This expectation value is used to assign the data point to class 0,1,2, the data point is assigned using a probability measure defined by the expectation value and the class labels:

$$P(y_i = 0) := \frac{(|M-1| + |M-2|)}{|M-1| + |M-2| + |M-0|}$$

$$P(y_i = 1) := \frac{(|M-0| + |M-2|)}{|M-1| + |M-2| + |M-2|}$$

$$P(y_i = 2) := \frac{(|M-1| + |M-0|)}{|M-1| + |M-2| + |M-0|}$$

Note that for classifications with more than three classes should not use this measure since it would require discontinuities in the decision landscape which is not optimal. For these classes, we can use modulo arithmetic to allow class 3 to liaise smoothly with classes 1 and 2, which would not be possible for four or more classes. The probabilities defined above will be what LIME uses to train a local explanation.

2.1.4 IRIS and Penguins Classification Results

Two classification methods have been applied. First, a classical random forest provides a benchmark against which to compare the quantum methods. Second, I used the described quantum circuit to classify the data using a variational method. The random forest classifier achieved a consistent accuracy of 98% 2.2. The quantum classifier performs similarly to the random forest classification, achieving 98% on the IRIS data set, as can be seen from 2.3. The accuracy varies slightly with the choice of initial parameters. Over ten attempts, the quantum neural network had an average accuracy of 97.6%, and the random forest had 98.1% on their randomly training sets. Note that the QNN parameters are always initialised randomly. The typical score of the Quantum classifier is consistently between 96% and 100% accurate.

On the Penguins data set, the performance of both models is similar but generally less accurate than the IRIS data set. The classical random forest technique achieves a classification accuracy of $91-93\%$ on a typical run 2.4. The quantum classifier has similarly high accuracy, achieving 91.81% on trinary

classification, comparable to the random forest method 2.5. On the Penguins data set, the quantum model was consistently slightly worse than the classical model.

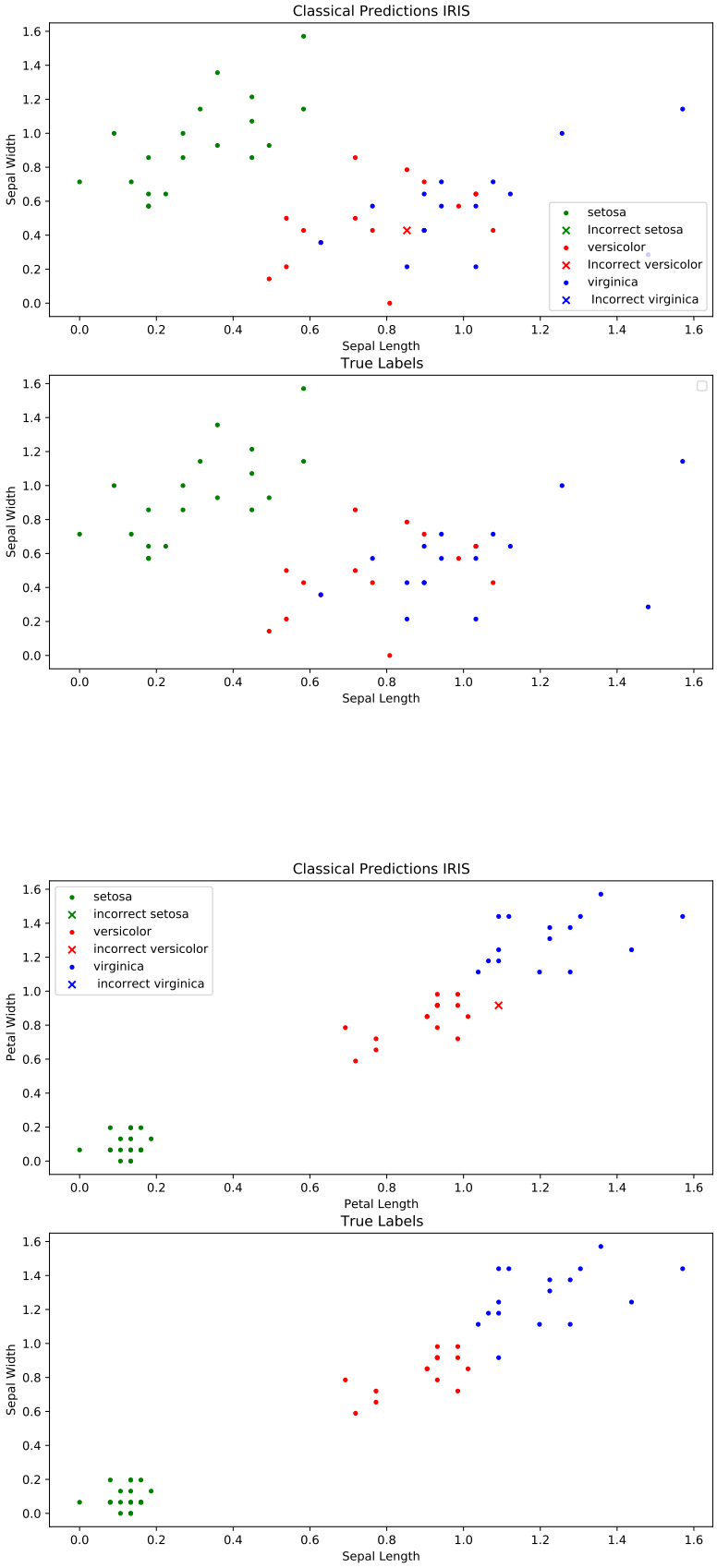

Figure 2.2: Visualisation predictions on the petal features, (left) and sepal feature (right) using a random forest classifier 98% accurate.

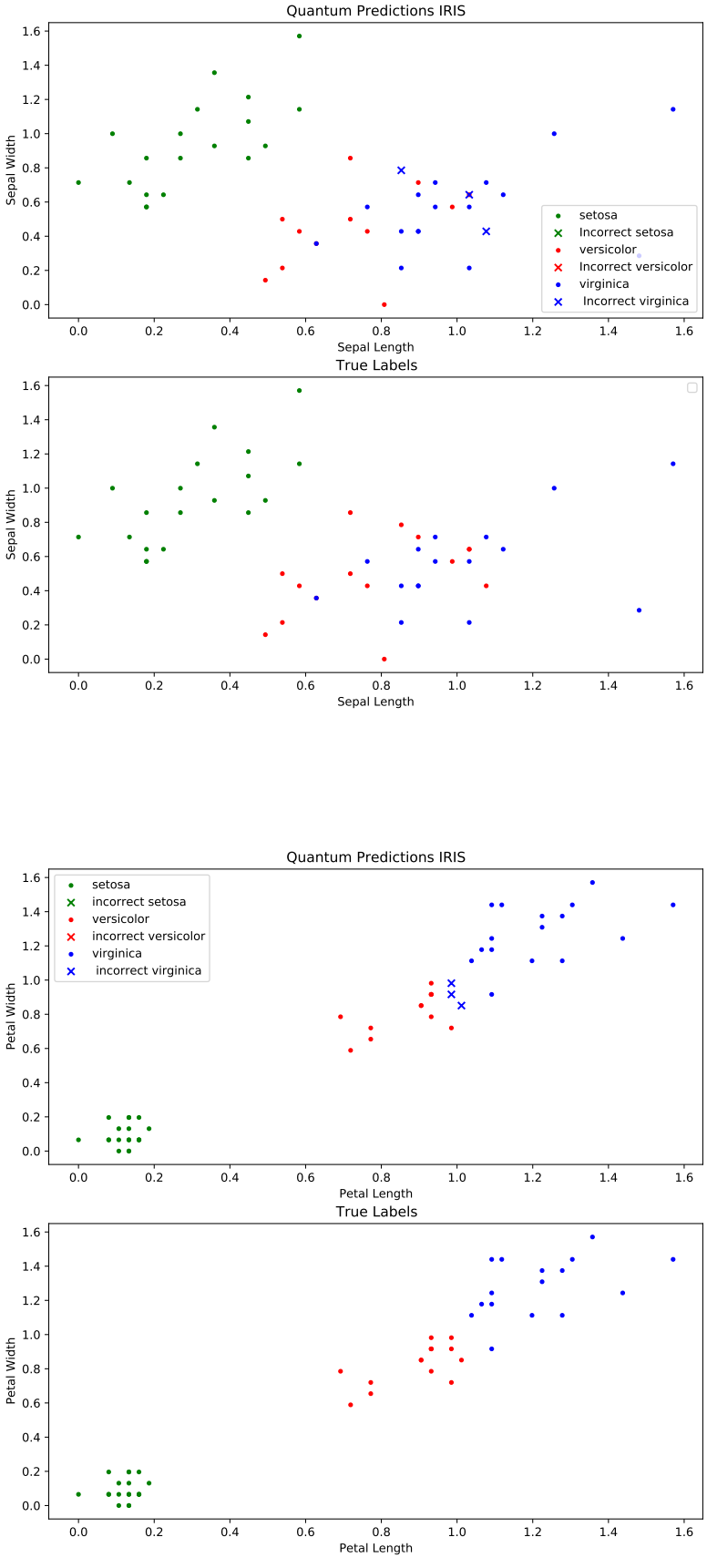

Figure 2.3: Visualisation predictions on the petal features (right) and sepal features (left), using a quantum neural network, 96% accurate.

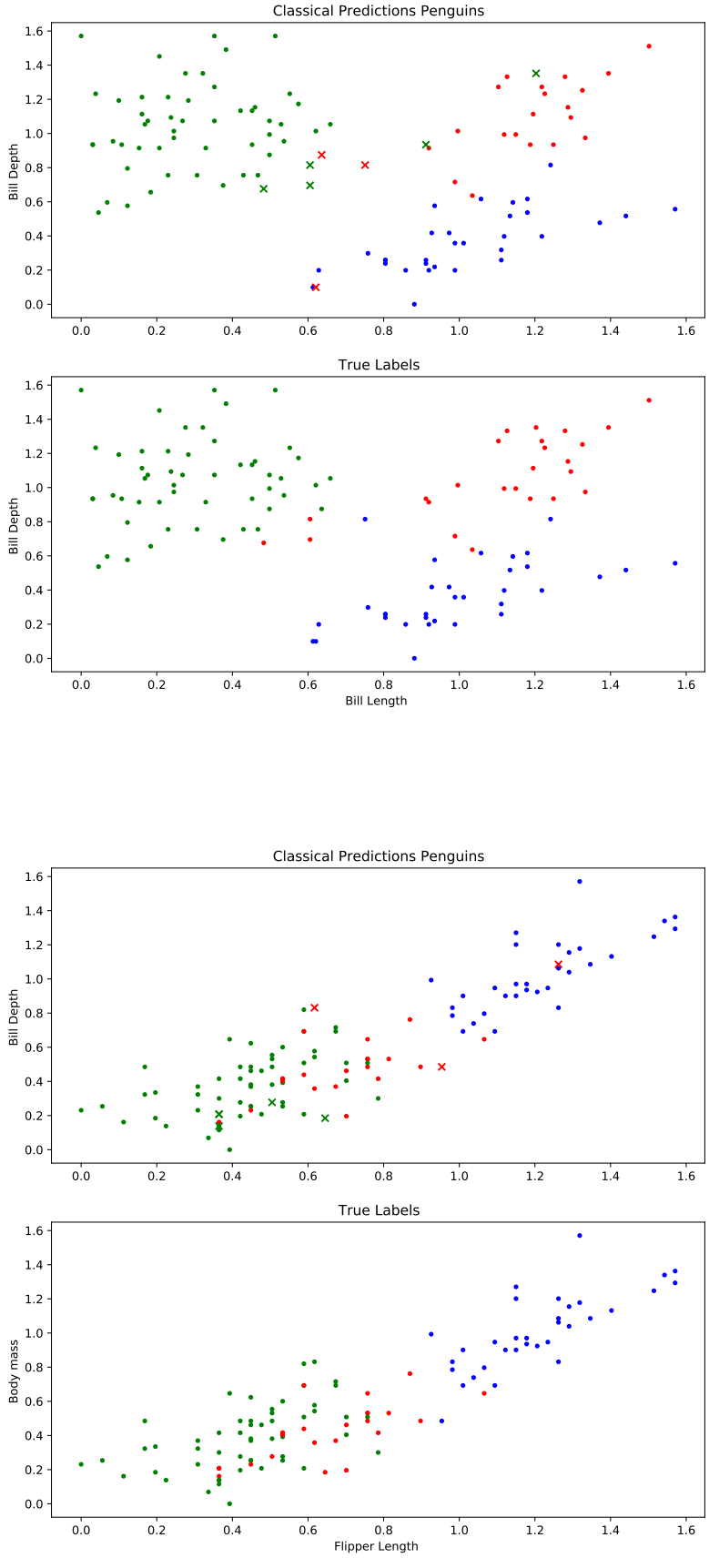

Figure 2.4: Visualisation predictions the Penguins data set using a classical random forest model, 96.36% accurate.

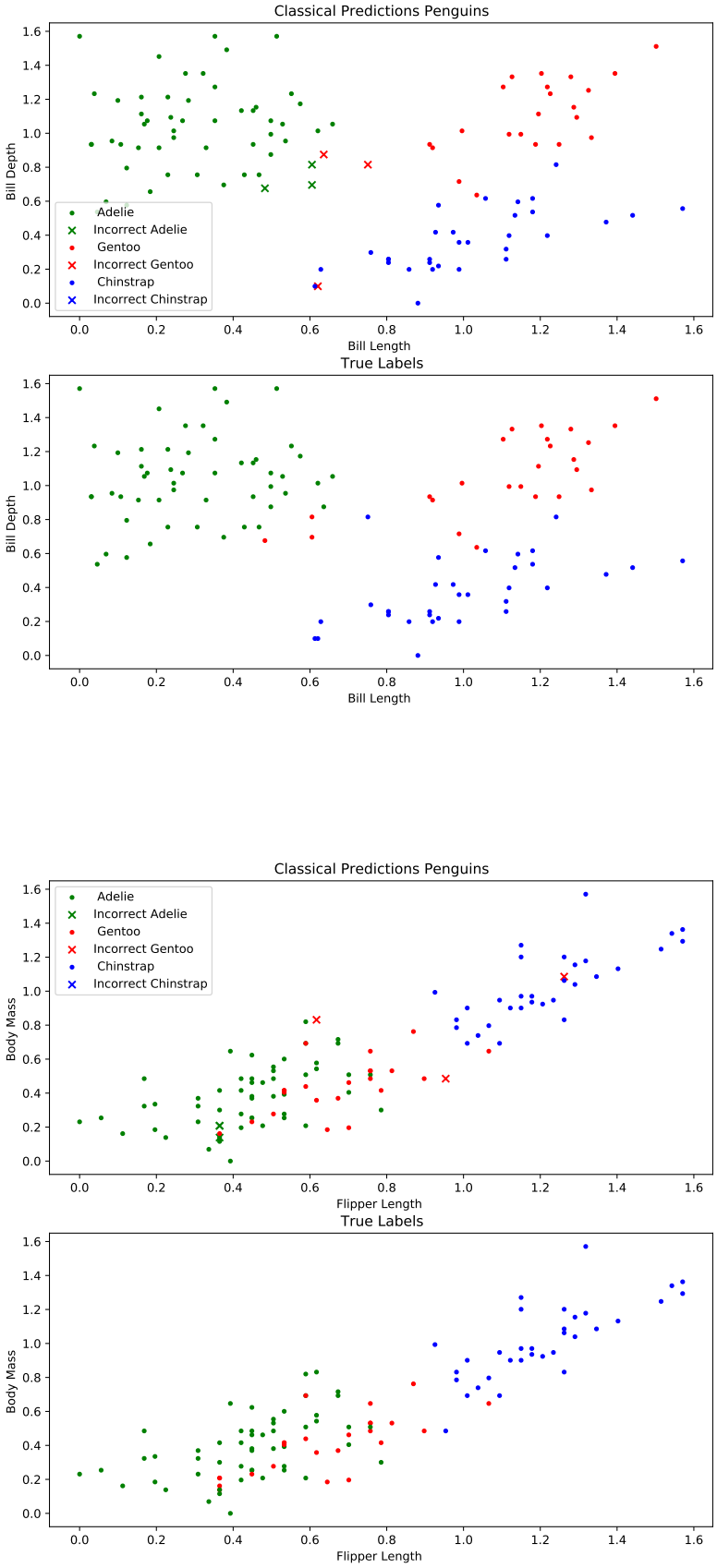

Figure 2.5: Visualisation predictions on the Penguins data set, using a quantum neural network 91.81% accurate.

2.2 Methodology: Ising Model

2.2.1 Motivation and Methods

A common technique for problems involving quantum systems is variational quantum methods [37] to find ground states, solving such problems known to be NP-hard on classical computers. Quantum computers can perform the required minimisation in polynomial time using black-box methods similar to or sometimes a QNN. We may apply explainability techniques to understand the physics that governs these presumable high complexity systems. In the next section, we will demonstrate the use of LIME in understanding the topological structure of the quantum magnetic phases of the 1-d Ising model. We find the ground states of the Hamiltonians by finding the lowest eigenvalue and compute the magnetisation of each ground state. That information is used to train a QNN to approximate the magnetisation of the ground states of unseen Hamiltonians using their ground states, the predicted magnetisation is given by

$$\langle GS|\, U^{\dagger}(\underline{\theta}) M_z U(\underline{\theta})\, |GS\rangle$$

s The ground states are acceptable for an ANN or QNN to determine the quantum phase, but this long vector of numbers is not clear for humans. LIME allows one to use a complex model such as an ANN or QNN to solve the topological landscape of the magnetisation and then provide simple explanations using variables such as the field and site-coupling strengths, which are easy to interpret for humans. Thus I will use a new set of variables for the interpretable explanation g given by $z' \in \mathbb{R}^2$ the vector containing the site coupling and magnetic field strength. The simple interpretable model is a ridge regression using the Gaussian distance kernel discussed earlier on the interpretable variable z'.

2.2.2 Ising Model

In this section we introduce the well know and analytically solved [47] Ising model on a 1-d qubit chain. The 1-d Ising model is a simple model that exhibits topological phase transitions between anti-ferromagnetic and ferromagnetic phases. The Ising model Hamiltonian is given by:

$$H_{Ising} = -h\sum_i S_i^z - J \sum_{<i,j>} S_i^z S_j^z, \tag{2.1}$$

with

$$S_i^z = 1 \otimes \cdots \otimes 1 \otimes \sigma_z \otimes 1 \otimes \cdots \otimes 1 \;\forall i \in \mathbf{Z} : i \leq N \tag{2.2}$$

where $< i, j >$ denoted all neighbouring vertices and σ_z is applied the i^{th} tensorial component, h is the magnetic field coupling strength, and J is the magnitude of the site-site interaction. I consider the case of non-periodic boundary conditions in one dimension, simplifying the Hamiltonian to

$$H_{Ising} = -h\sum_{i=1}^{N} S_i^z - J\sum_{i=1}^{N-1} S_i^z S_{i+1}^z. \tag{2.3}$$

The Ising model with a longitudinal external field has two phases. In the anti-ferromagnetic phase, the local magnetic moments will cancel, resulting in 0 overall magnetisation of the spin-chain. Conversely, in the ferromagnetic phase, there is an overall magnetic moment due to the alignment of the spin-chain qubits in the same direction due to the magnetic field.

Figure 2.6: Visualisation of the Ising spin chain. The arrow indicates the direction of spin for each qubit. In the ferromagnetic phase (right) all spins are aligned, causing the chain to have an overall magnetisation and in the anti-ferromagnetic (left) phase all of the spins are anti-aligned to their neighbours.

The overall magnetisation of the eight qubits Ising chain used in this example is given by

$$M_z = \sum_{i=1}^{N} |S_i^z|. \tag{2.4}$$

The calculated magnetisation for an eight-qubit system is given in figures 2.7 and 2.8.

2.2.3 Data Generation

To generate the data for the Ising chain, I generate Hamiltonians for various site-site couplings and external field values for my chosen number of qubits. Any choice of qubits can be made, in this work eight qubits are considered due to the difficulty in finding the ground states for larger Hamiltonians. The ground states of these Hamiltonians can be found variationally using a VQE, as discussed prior. The ground states are then labelled depending on their specific phase, which depends on the magnetisation. Since the magnetisation undergoes a discrete phase transition or the case of $n \to \infty$ classes can be assigned using the calculated magnetisation. The quantum classifier acts on the ground states and produces an expectation between 0 and 1, which is trained to match the magnetisation in the Z direction given by the

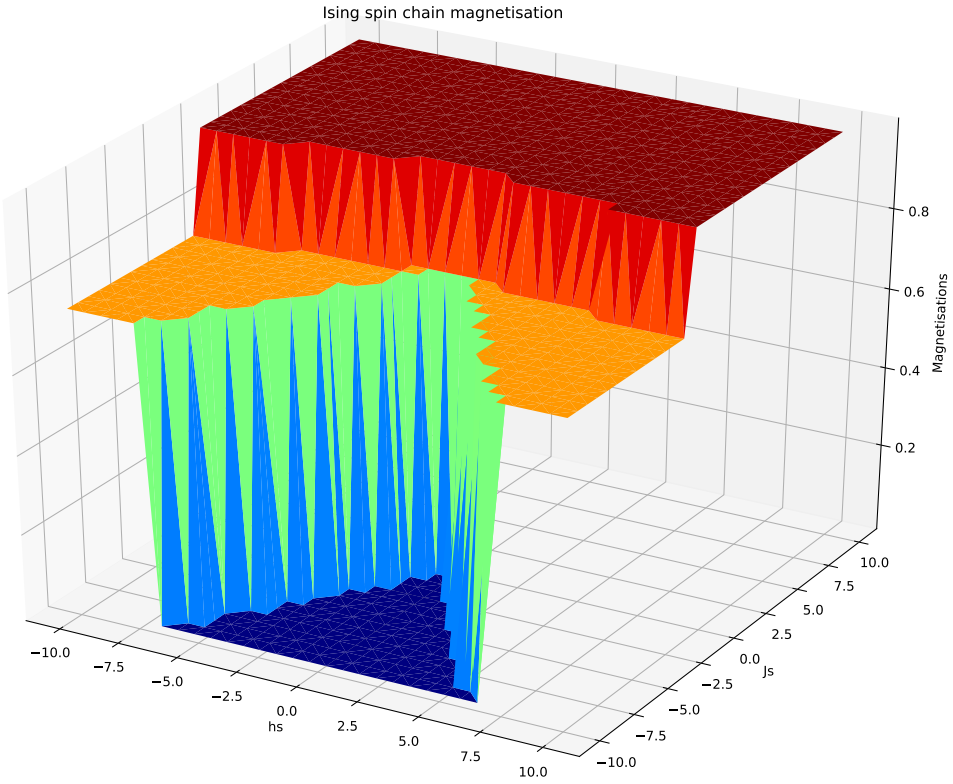

Figure 2.7: Visualisation of the magnetisations for an eight-qubit system with varying site-site coupling constant J and longitudinal magnetic field strength h

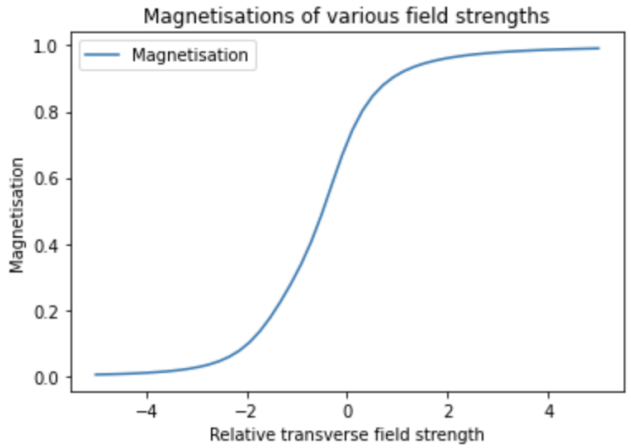

Figure 2.8: Visualisation of the magnetisations for an eight-qubit system with fixed site-site coupling $J = -1$. As $n \to \infty$ a discontinuity occurs at relative field strength $h = -1$ as expected given the analytic solution [47].

operator (4.4). The data generation work flow is as follows:

- Step 1: Using OpenFermion we created the Hamiltonian from (2.3), this is a relatively simple step since the package has all of the required methods to create either an explicit Hamiltonian or a sparse representation.

- Step 2: Determine the ground states of each Hamiltonian. This can be done by finding the lowest eigenvalue on a classical computer. For larger numbers of qubits this quickly become unfeasible, which is the reason that methods such as VQE are useful. The explicit use of a VQE is beyond the scope of this work however.

- Step 3: Using the OpenFermion package compute the choice of observable, in this case the magnetisation along the Z axis given by (2.4). We now have all of the data points that we require to continue.

2.2.4 Ising Classification

The classification circuit for quantum data is slightly different from the circuit for classical data due to the size of the data sets. The IRIS and Penguins datasets could be encoded onto two qubits, whereas the Ising chain will have up to 8 qubits with two ancillae, resulting in a state vector of length $2^{10} = 1024$. The classification circuit used a convolutional ansatz to reduce the effect of the Barren plateau problem and is shown in figure 2.9.

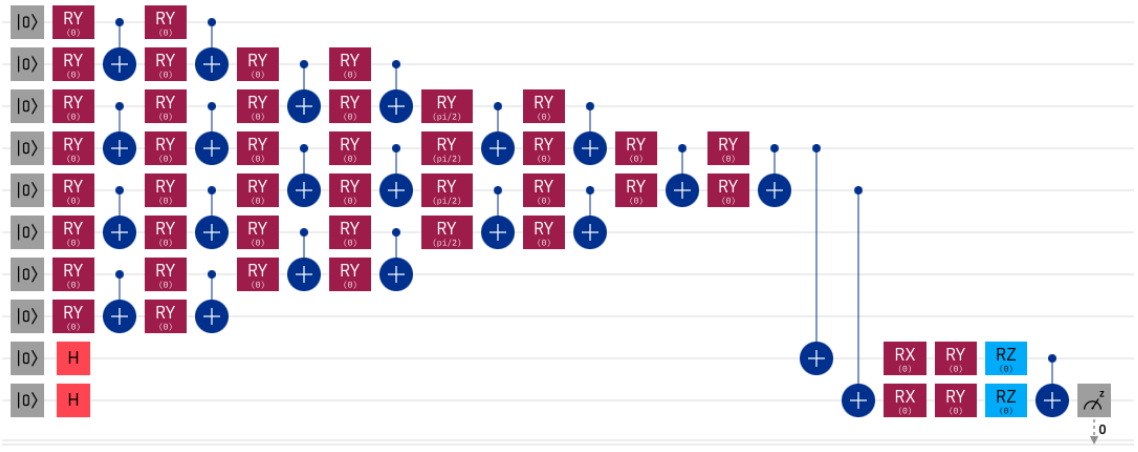

Figure 2.9: The classification circuit to be used for the eight qubit problem. There are ten qubits, two of which are ancilla and eight of which are computational qubits to encode and manipulate the ground states. The ancilla qubits are measured to determine the classification result.

Using a quantum classifier, we can try to learn the magnetisation of the quantum spin chain and, using our knowledge of the behaviour of the quantum phases, classify ground states according to their membership in a ferromagnetic or anti-ferromagnetic system.

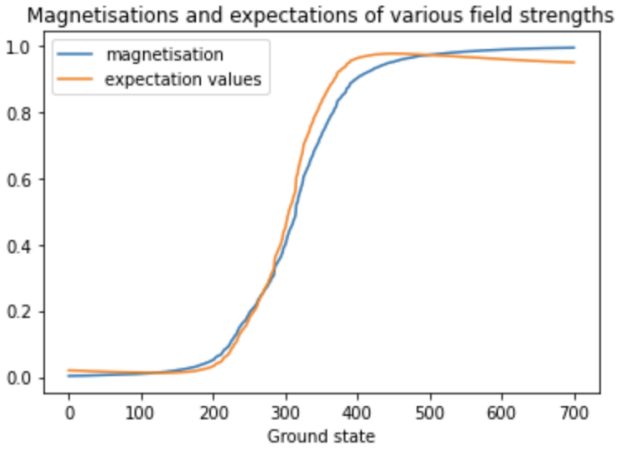

Figure 2.10: Visualisation of the magnetisations and expectations for an eight-qubit system with fixed site-site coupling $J = -1$ where the ground states are sorted by increasing magnetisation. As in the previous figure, the considered range of relative field values is $[-5, 5]$

We can see that the expectation of the quantum circuit can approximate the magnetisation observable of the Ising spin chain. It will be useful to use approximate observables to analyse the importance of the site-site coupling and the external magnetic field using LIME, as will be done in the subsequent subsections.

2.2.5 Ising classification Results

In this sections I present the results of the quantum and classical classification attempts for 3600 data points. My classification variable is the phase of the lattice, either anti-ferromagnetic or ferromagnetic, and I use the ground state of each Hamiltonian and the computed magnetisation to train the network. Ten thousand data points are created with various values for the site-site coupling constant and external magnetic field strength. The phase transition depends on the value of the site-site coupling constant and external magnetic field strength, which vary between -10 and 10. As the magnetic field grows we expect to see a phase transition at relative field strength $g = -1$ as seen in figures 2.8 and 2.10 . The classifiers can easily learn the classification boundary, and they are perfectly accurate for low qubits. As the qubit count increases, they encounter difficulty, but the QNN maintains its quality relative to the ANN, scoring equivalent accuracy across all qubit counts. The embedding circuit for the ground states is given by the identity embedding, which is the same as the amplitude embedding, where the input vector is the

ground state vector. The computational circuit is a convolutional circuit where each convolutional block is repeated n times.

Qubits	Training size	QNN Accuracy	ANN Accuracy
4	0.3	1.0000	1.0000
4	0.5	1.0000	1.0000
4	0.7	0.9997	1.0000
5	0.3	0.9992	1.0000
5	0.5	0.9995	1.0000
5	0.7	0.9992	1.0000
6	0.3	0.9985	0.9971
6	0.5	1.0000	1.0000
6	0.7	0.9966	1.0000
7	0.3	0.9912	0.9900
7	0.5	0.9889	0.9920
7	0.7	0.9942	0.9933
8	0.3	0.8734	0.9886
8	0.5	0.9128	0.9912
8	0.7	0.9356	0.9992

Table 2.1: Comparison of QNN and ANN accuracy for various numbers of qubits. We see the accuracy of both methods decrease as the size of the systems increases.

The entries are the average result over five runs of each classifier for each test-train split and each number of qubits using the ansatz given in figure 2.9, or a reduced version where the qubit count is lower. As we can see, the quality of the quantum classifier is high, but it starts to falter as the qubit count increases. This is likely due to the choice of ansatz and a better understanding of classical machine learning techniques. There is also some variation depending on the initial parameters of the order 0.01 of the accuracy, but this is negligible. The predicted landscape for the model is given in figure 2.11 with error given in figure 2.12. As we can see that the prediction faithfully represents the system. This is useful because we can query the model which has learned the magnetisation observable and apply explanations to the field constants. This means we can extract explanations on the field constants by using a model trained using complex ground states that are difficult for a human to interpret easily.

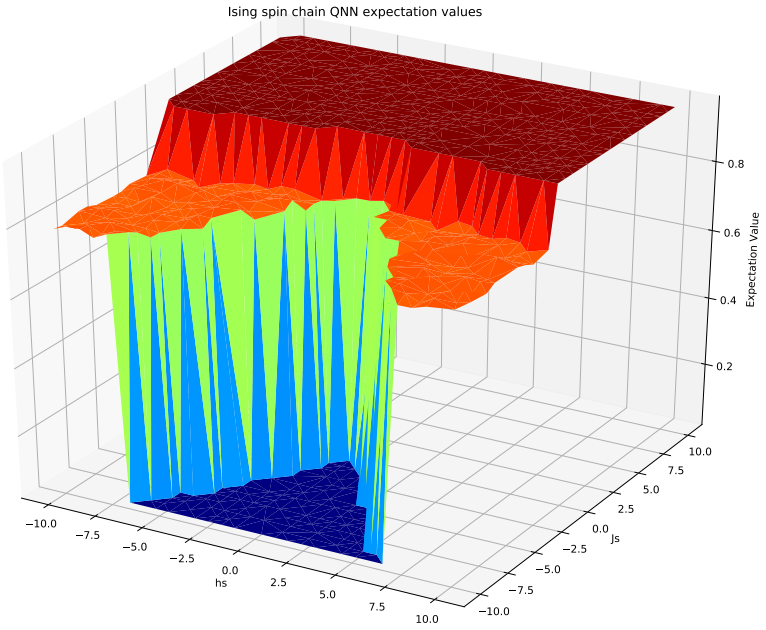

Figure 2.11: Visualisation of the predicted magnetisations for an eight-qubit system with varying site-site coupling constant J and longitudinal magnetic field strength h.

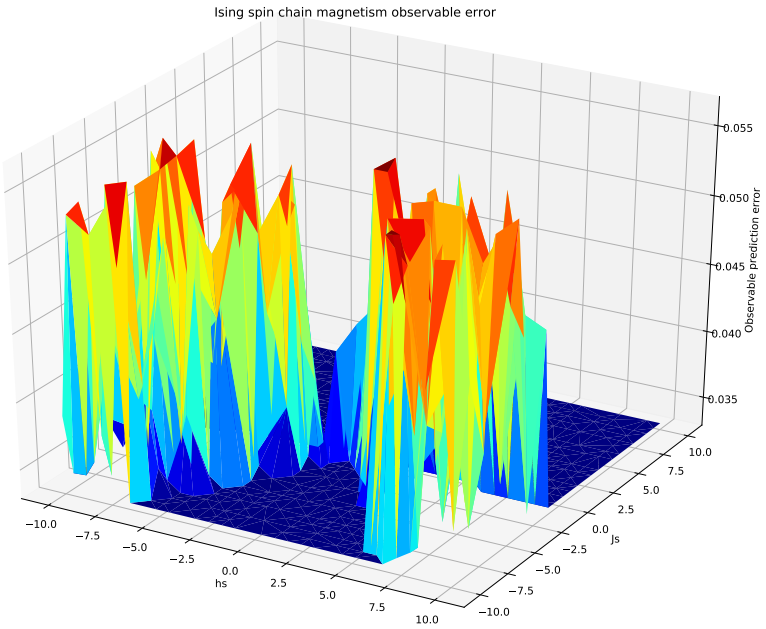

Figure 2.12: Visualisation of the error of the magnetisations for an eight-qubit system with varying site-site coupling constant J and longitudinal magnetic field strength h.

2.3 Results For Classical Data

2.3.1 LIME Results

We now present the LIME local explanations that may be used to gain a locally-valid heuristic understanding of the dynamics of the systems in question; the IRIS and Penguins data sets. LIME will give us a description of the influence of each of the n most strongly weighed parameters. In order to understand how we interpret the LIME figures, consider figure 2.13.

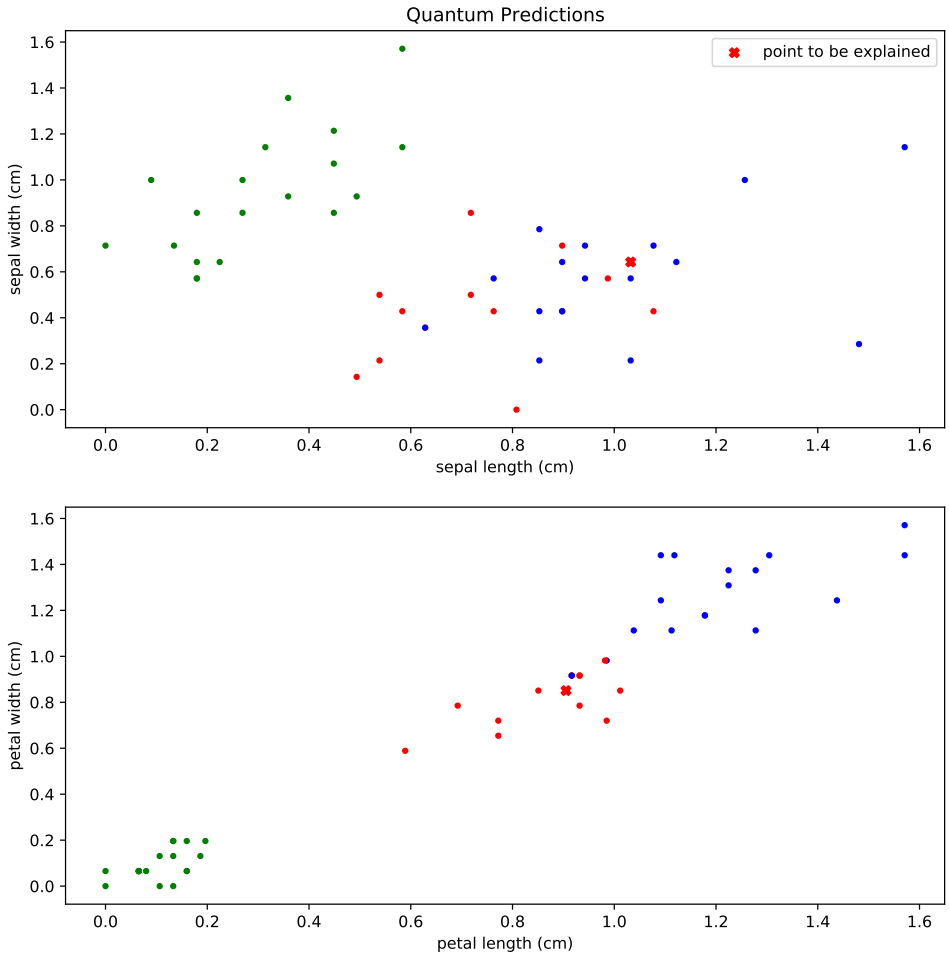

Figure 2.13: The red cross gives the data point we are explaining, in this case a correctly classified versicolor point.

Figure 2.13 is the representation of the data point one might wish to have an explanation for in terms of all of its features. We can see that the point can be separated in terms of its petal length and width but

less simply in terms of its sepal features. The assigned class probabilities for this data point as predicted by the quantum neural network are

$$(P(Setosa), P(Versicolor), P(Virginica)) = (0, 0.58, 0.42)$$

We can see this information presented in the "Prediction probabilities" tab of figure 2.14. By contrast, the LIME analysis for the random forest model using the same points is given in figure 2.15, the probabilities predicted by the random forest model are

$$(P(Setosa), P(Versicolor), P(Virginica)) = (0, 1, 0)$$

The probabilities of the model we wish to explain become the training data as per section 1.3.2.

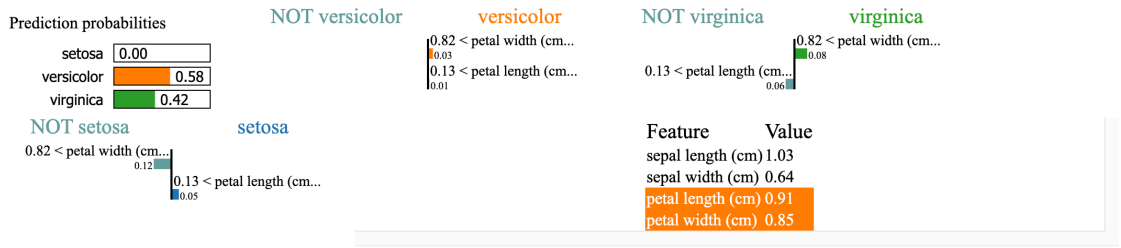

Figure 2.14: LIME explanation of the given data point based on the output of the quantum classification model using two features.

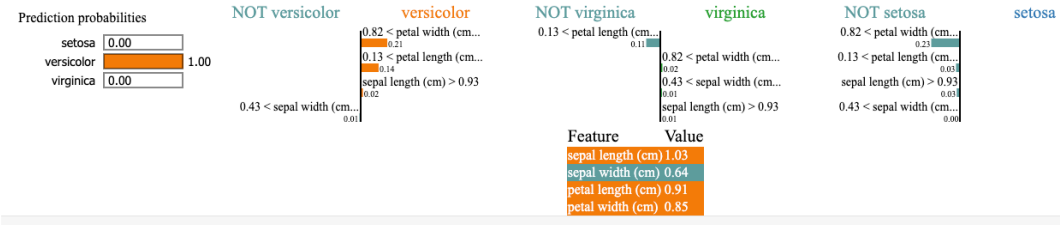

Figure 2.15: LIME explanation of the given data point based on the output of the classical random forest classification model using four features.

We can see an explanation using two features in 2.14. LIME determines the two most important features by fitting a regularised LASSO-regression to the data weighted by locality to the data point as described in the sections above. The two most important features are the petal width and length from figure 2.13. The prediction probabilities for each class are given in the "predictions probability" tab, and the explanations for why it has assigned each probability are the given bar charts. The weights next to the

bars indicate the parameters of the linear model that approximate the explanation in the area of locality of that point, the region in which this weight is valid is displayed above the bar, it also indicates how much the probability would change if that feature was not considered. If a weights is negative for one feature it may be positive for a different feature. The highlighted (orange in this case) features indicate the important features for each class. Depending on the number of features chosen, the number of highlighted features will change accordingly. Note that the features of the explanation will vary depending on the model used to generate the initial decision space (from which the interpretable model is derived). They behave quite differently when LIME is applied to a random forest classifier and to the quantum classifier. Using LIME, we can generate explanations of a chosen complexity by fixing the number of features. In this model, which uses tabular data, the best way to measure complexity is to count the number of features we use in our explanation. In figure 2.14 we see an explanation using two features. Since this is a simple data set, we may include all of the features to obtain a more detailed explanation as in figure 2.16

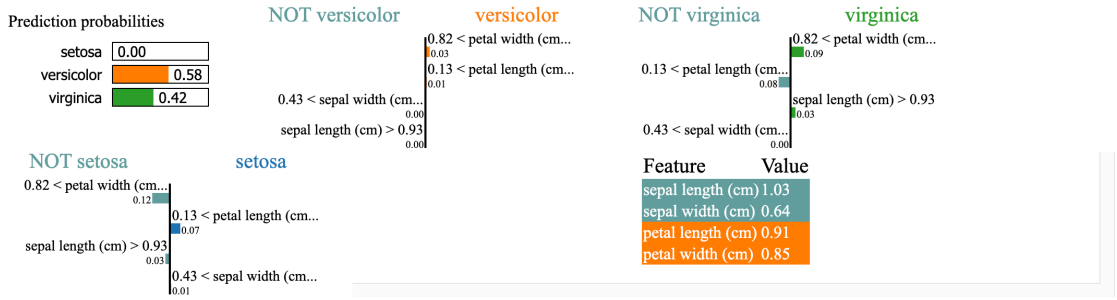

Figure 2.16: LIME explanation of the given data point based on the output of the quantum classification model using four features.

We can see that now that we have added the last two features, the model has found justifications for classifying the point into two of the three categories, with the sepal values suggesting that the class is virginica and the petal values suggesting versicolour. This reflects real-life decisions since it is not always clear how we should classify an object or what we should conclude. LIME can encapsulate that same kind of uncertainty by using a linear model that provides local explanations similar to those that a human might make. Hence they are humanly understandable. The same type of behaviour persists (as expected) for the classical random forest, considering the same data point given in figure 2.13 but using an explanation arising from the random forest model. The explanations are given in figures 2.17 and 2.18 give an almost certain probability of assignment to ”Versicolor” for both two and four features, respectively. The absolute assignment to this class does not look (as in from figure 2.13 to be as clear-cut as this explanation would suggest. This shows that LIME can outline the differences in the human explanation, which in practical

cases would be a subject matter expert, and that given by the model we have trained. For this data, the quantum model has more adequately captured the dynamics even though the quantum model is a slightly less accurate classifier. This is because the model has communicated the perceived uncertainty in classifying the data point, whereas the classical model has failed to do so. Despite the differences in the quantum and classical models, both explanations prefer using the same features, the petal width and length, as the primary classification features. The agreement of the two models could be enough to convince an expert to make the same classification.

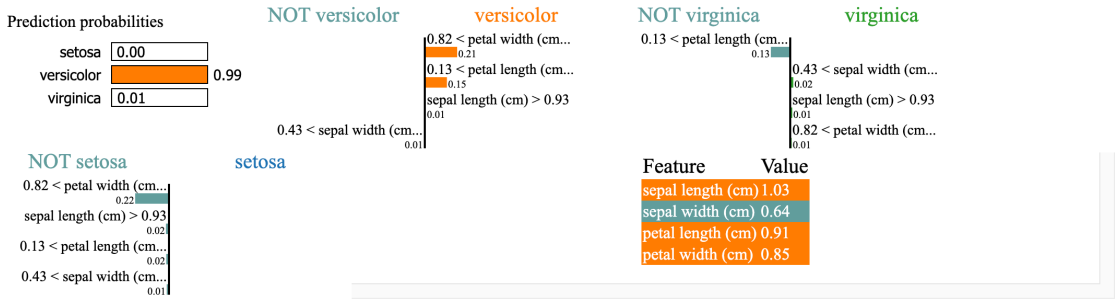

Figure 2.17: LIME explanation of the given data point based on the output of the classical random forest classification model using four features.

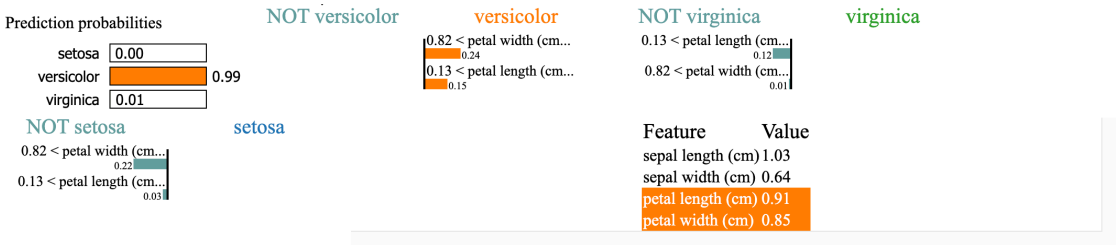

Figure 2.18: LIME explanation of the given data point based on the output of the classical random forest classification model using two features.

We see similar behaviour in the Penguins dataset. We now demonstrate an explanation for the quantum and classical models for various numbers of features on two different data points. Consider a quantum explanation of the data point in figure 2.19 with explanations 2.20 and 2.21.

Unlike the previous example, this explanation is right on top of a decision boundary, giving a prediction probability of 50/50 for 'Adelie' and 'Gentoo'. The key features are the 'bill depth' and 'flipper length'. In the four feature explanation, the 'body mass' is equally weighted with the 'bill depth' and 'flipper length'. This seems to suggest that there is not much in it in terms of the most important feature. Here the LIME explanation primarily demonstrates a huge uncertainty in this classification, which might

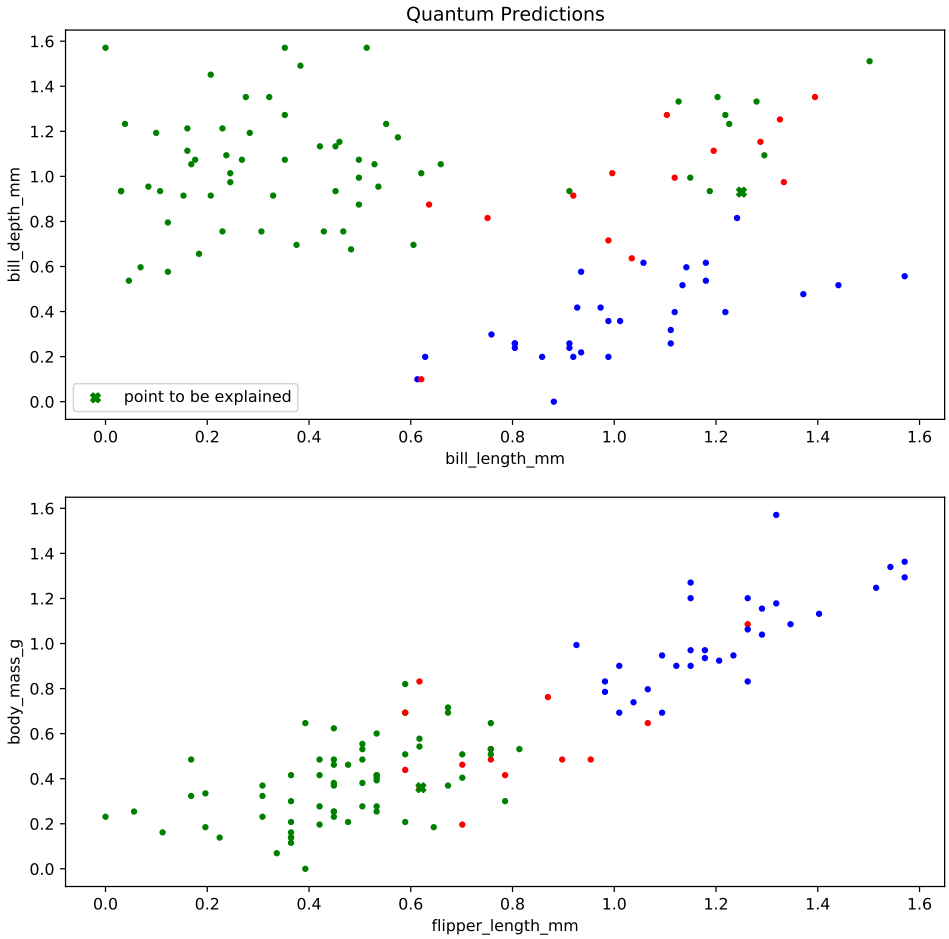

Figure 2.19: The green cross gives the data point we are explaining, in this case a correctly, in this case the data point is incorrectly classified Adelie.

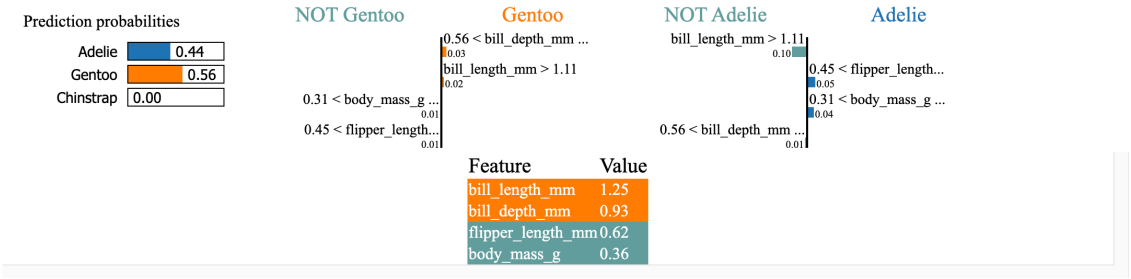

Figure 2.20: LIME explanation of the given data point based on the output of the quantum classification model using four features.

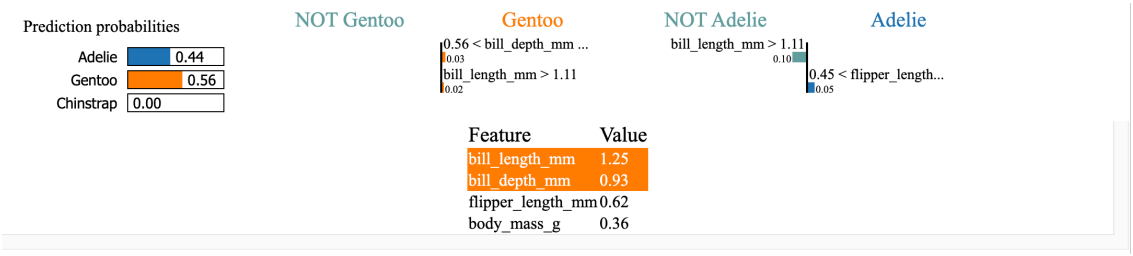

Figure 2.21: LIME explanation of the given data point based on the output of the quantum classification model using two features.

inform ornithologists[2] to examine this prediction more closely. Also consider the explanations of the classical model the the data point given in figure 2.22 with explanations 2.23 and 2.24.

This data point does not look immediately apparent, yet the explanation is classified with complete certainty by the random forest classifier. This may be entirely correct, but it does fly in the face of heuristic judgement. Such situations are likely to be more common as different professionals interact with AI more on the day-to-day, and it is thus important that we must be well equipped with techniques that may help us find flaws or curiosities in the models we use.

[2]They study penguins.

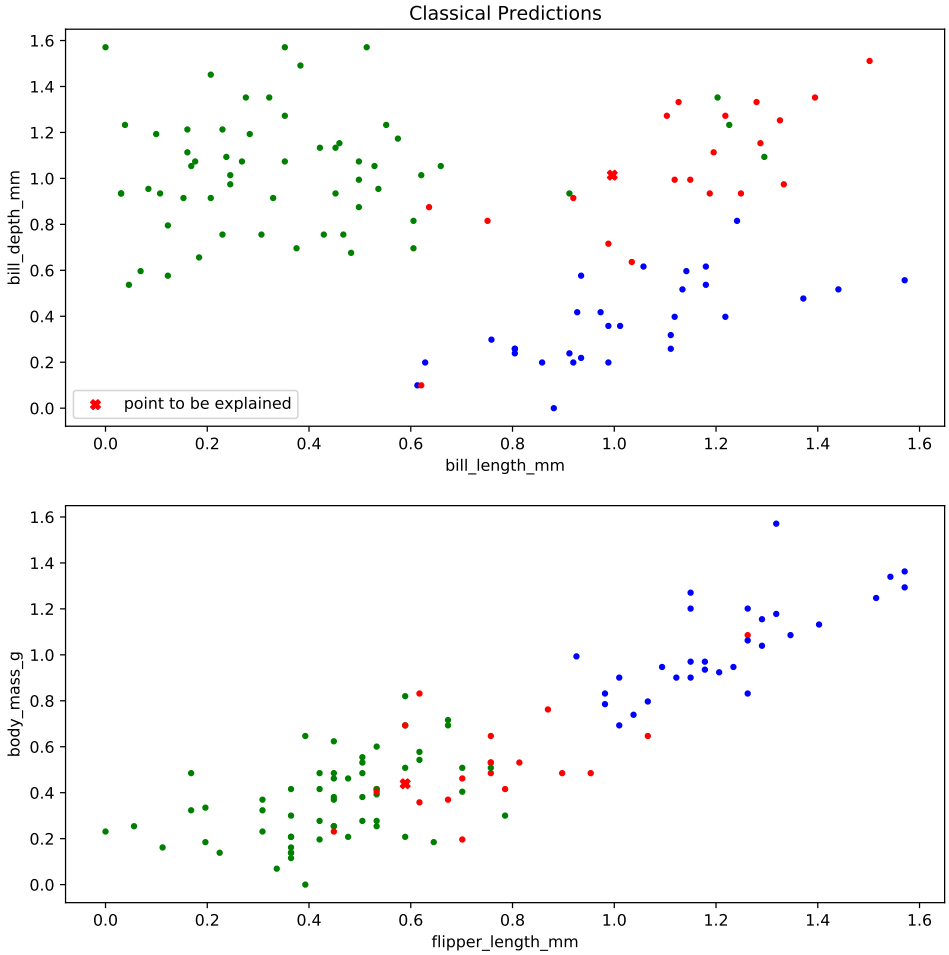

Figure 2.22: The red cross gives the data point we are explaining, in this case a correctly classified Gentoo.

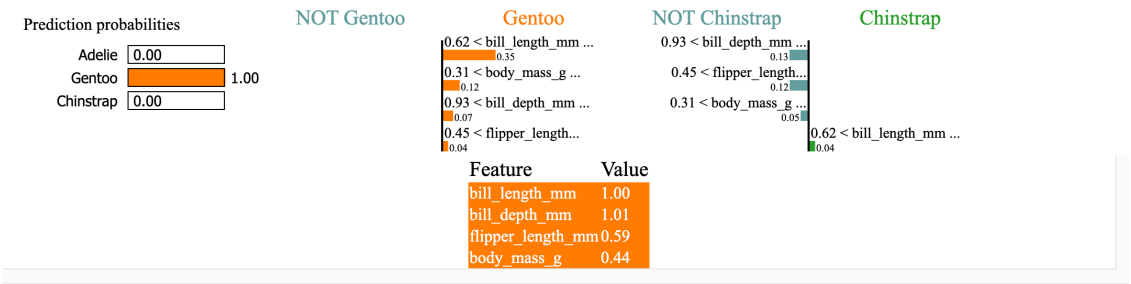

Figure 2.23: LIME explanation of the given data point based on the output of the quantum-classical random forest model using four features.

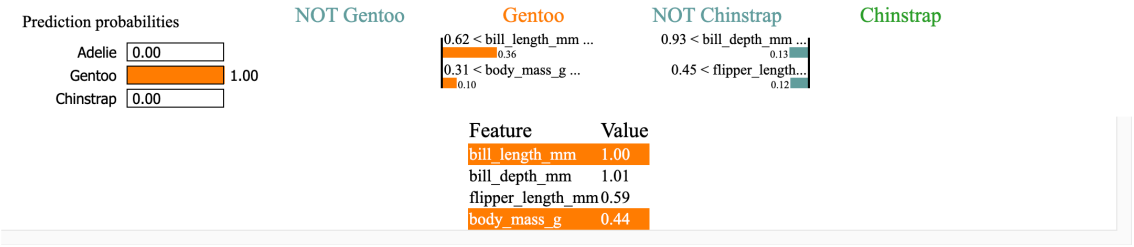

Figure 2.24: LIME explanation of the given data point based on the output of the classical random forest classification model using two features.

2.4 ISING Model Results

This section investigates XAI's use to investigate the properties of phase transitions in quantum lattice models. A quantum phase transition is an abrupt change in matter from one state to another characterised by a change of order of the constituent components of the system. Common examples of phase transitions are ordered phases where the magnetic moments of all lattice points have a global order and disorder phases where the magnetic moments are randomly aligned. Quantum phase transitions occur at low temperatures [47] where classical temperature-based variations do not hide their quantum signature. Therefore, for any quantum phase transition, we assume that the temperature is near absolute zero. The example to be considered is the well-known 1-d Ising model with an external longitudinal magnetic field. We will attempt to explain the phase of the system using a quantum classifier and then provide a human interpretable explanation using LIME as before.

2.4.1 Quantum Phase Transitions Explanations Using LIME

First, we compare two data points using the same number of features $(h, J) \in \{(1.9, -1), (1.23, -1)\}$ which are relatively close in terms of the scale of the external field strength.

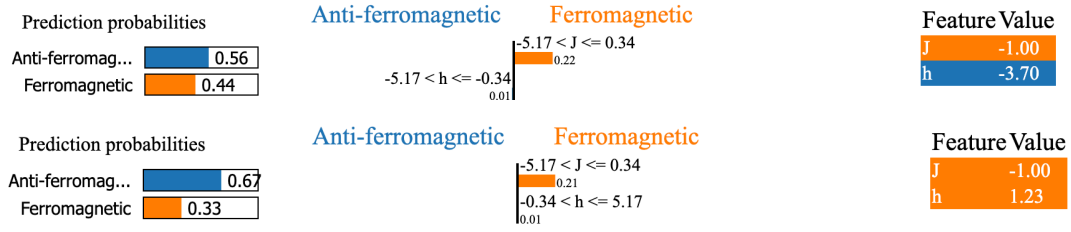

Figure 2.25: LIME output for the quantum neural network using two features on similar data points.

Figure 2.25 has explanation for two data points with a fixed site coupling strength. The data point $z_1' = (-1, -3.7)$ has a local explanation in which h correlates negatively to the ferromagnetic phase.

Meaning that if we increase the magnitude of h in the $-z$ direction, we are likely to see the probability of an anti-ferromagnetic phase fall in response. In fact, based on this explanation at at $z'_3 = (-1, -5.7)$ the predicted probability of an anti-ferromagnetic phase is 0.21, a decrease from the previous value of 0.44 at z'_1 as suggested by the interpretable model. Conversely the data point $z'_2 = (-1, 1.23)$ has a h that is positively correlated with a ferromagnetic phase, so now if h in the z direction increases so will the probability of a ferromagnetic phase. When calculated at the point $z'_4 = (-1, 2.5)$ the probability of seeing a ferromagnetic phase increases to 0.72. The coefficient of h is opposite in each of the locations, which means that both models are only accurate in one direction. The explanation in the top panel of figure 2.25 only sees a ferromagnetic phase if h becomes strongly negative and the bottom panel of figure 2.25 requires that h become large and positive. We can compare these points to figure 2.11 and notice that both data points are close to phase transitions, but from differing directions which explains the difference in the interpretable models. We now consider two points on opposite sides of the phase transitions. The points $z'_5 = (-1, 0.5)$ is anti-ferromagnetic and the points $z'_6 = (1, 0.5)$ is ferromagnetic

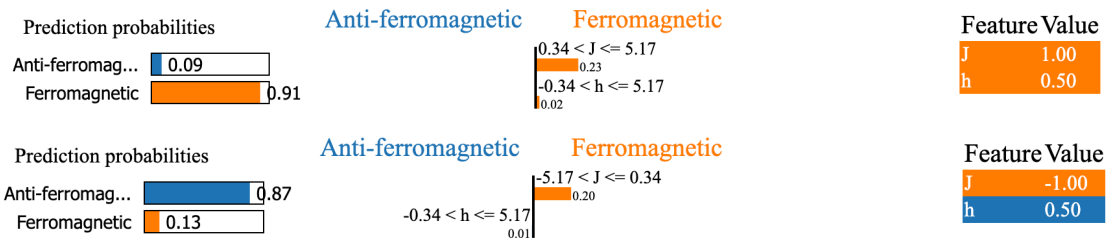

Figure 2.26: LIME output for the quantum neural network using two features on data points of the same quantum phases (anti-ferromagnetic).

We see from figure 2.26 that the site coupling strength J dominates the model at both points, which is reassuring since we are discussing a phase transitions near $h = 0$. The values for J, $0.32 < J \leq 5.17$ and $-5.17 < J \leq -0.34$ for z'_5 and z'_6 respectively. For z'_5 this means an increase in the filed strength from $J = -1$ to $J = -0.34$ would result in the prediction possibility of anti-ferromagnetism increase by 0.23. And the converse is true for the point z'_6. These simple conclusions demonstrate that the local models are able to provide valid explanations for a quantum phase due to a change in the site coupling strength. We can obtain analogous results on a phase transition driven by the magnetic field strength h. Consider the points $z'_7 = (-8, 3.5)$ and $z'_8 = (-8, 10)$. Using data points which are both in an anti-ferromagnetic phase we can see the effect of the change in local model between two pints. As before J has a positive coefficient since it is highly correlated to the appearance of a ferromagnetic phase. The top panel in

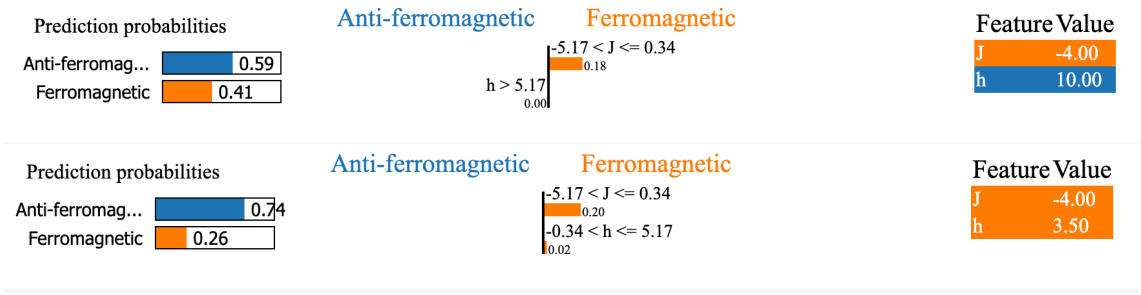

Figure 2.27: LIME output for the quantum neural network using two features on data points of the same quantum phases (anti-ferromagnetic).

2.27 has a field strength of $h = 10$ in the positive z direction, a strong value compared to the size of the site-site coupling strength $J = -4$. The bottom panel in figure 2.27 has magnetic field of $h = -3.5$ and $J = -4$ again. The negative coefficient for the magnetic field strength h in the top panel indicates that the probability of an anti-ferromagnetic phase will increase if h decreases to be 5.17. This is indeed the case and can be seen by the data point in the bottom panel which has a smaller magnetisation.

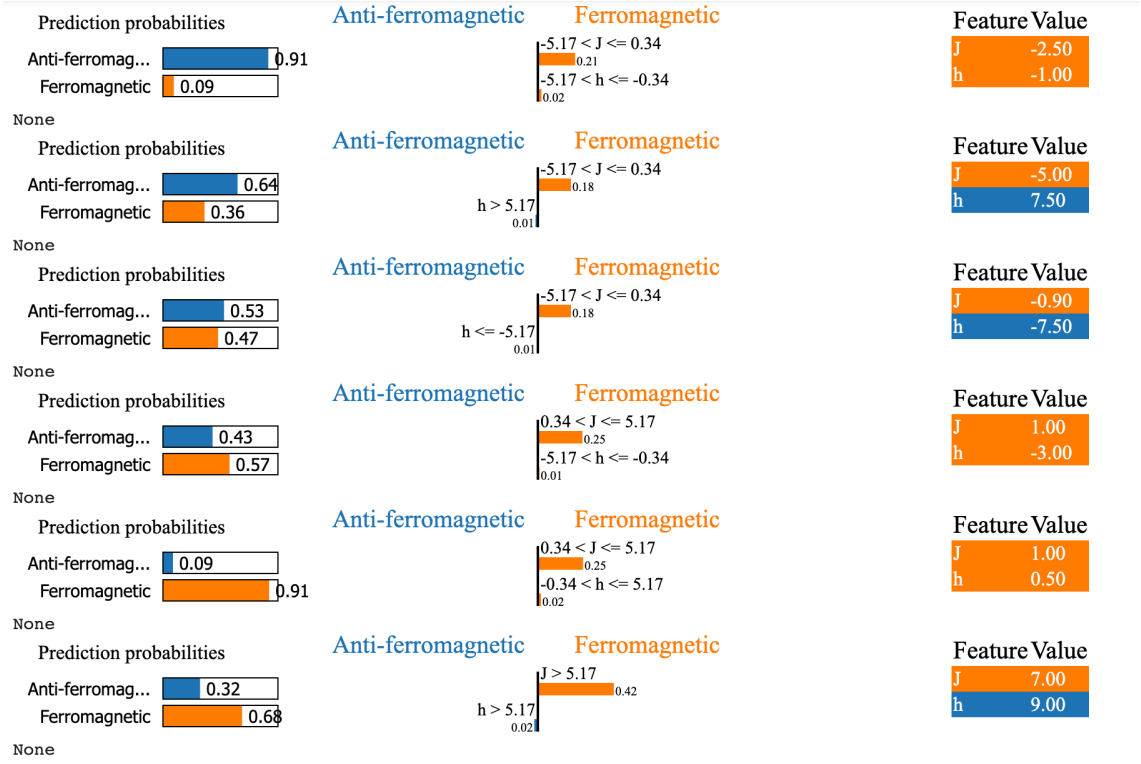

Figure 2.28: LIME output for the quantum neural network prediction for the magnetisation of the Ising chain using two features.

We see in figure 2.28 that the first two panels demonstrate the difference in the sign of the h coefficients when h is above or below -1 respectively. However since the value of J is negative both

of the top two panels are anti-ferromagnetic. Data points three and four seem to be separated by a phase transition as seen by the change of prediction probabilities from probably anti-ferromagnetic to ferromagnetic. The final two data points are both in a ferromagnetic phase due to the positive values of J. Each of these local explanations are valid, nut the collection of several local explanation starts to give us a picture of the entire decision landscape in the simple case of the 1-d Ising chain.

Chapter 3

Conclusion

Presently the field of Explainable Machine Learning is booming, with many new techniques under development and old ones being applied to new systems. Due to the complexity of QNNs and other quantum algorithms Quantum-XAI will likely begin to perform the same role in QML as it does in modern ML, namely to provide a simplified interface into the actions of otherwise highly complicated models. A better understanding of the models we use will lead to greater trust in the models and then to the ability to give AI algorithms additional responsibilities. The application of explainability to quantum computing tasks is important because of the complexities involved with modelling quantum systems. When quantum computers are able to support enough logical qubits to perform more useful calculations with low error rates, having methods that can quickly extract meaning from complicated QNN type models in the way we have in this work will become extremely important. Quantum XAI methods do not necessarily need to differ from classical methods, such as I have shown by applying LIME to a quantum classifier, but there is definitely room to produce quantum backpropagation methods or intrinsic quantum methods that utilise the structure given to us by Hilbert space. QXAI is likely to follow straight from classical XAI, with common existing methods simply being altered for the benefit of quantum circuit behaviours such as the use of fully reversible unitary gates and entanglement, for which we have no classical analogue. In this work we discussed the need for XAI for their uses in understanding the way models learn from data that they are created to predict in order to build trust or do research, demonstrated the use of a QNN to learn magnetisation observables of the Ising chain and the application of LIME to both a quantum classifier and the Ising model. Our novel contribution is the use of LIME on a QNN to analyse the magnetic quantum phase transitions of the Ising chain in 1-d. We found that LIME could be applied to the magnetic phases of the Ising model to create locally valid explanations for both ferromagnetic and anti-ferromagnetic phases.

Appendix A

Data

A.1 Tables

sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
5.0	3.6	1.4	0.2
5.4	3.9	1.7	0.4
...	...	...	...
6.3	2.5	5.0	1.9
6.5	3.0	5.2	2.0
6.2	3.4	5.4	2.3
5.9	3.0	5.1	1.8

Table A.1: IRIS dataframe

species	island	bill length mm	bill depth mm	flipper length mm	body mass g	sex	year
Adelie	Torgersen	39.1	18.7	181.0	3750.0	male	2007
Adelie	Torgersen	39.5	17.4	186.0	3800.0	female	2007
Adelie	Torgersen	40.3	18.0	195.0	3250.0	female	2007
Adelie	Torgersen	36.7	19.3	193.0	3450.0	female	2007
Adelie	Torgersen	39.3	20.6	190.0	3650.0	male	2007
...	...	...	...	...	...	...	...
Chinstrap	Dream	43.5	18.1	202.0	3400.0	female	2009
Chinstrap	Dream	49.6	18.2	193.0	3775.0	male	2009
Chinstrap	Dream	50.8	19.0	210.0	4100.0	male	2009
Chinstrap	Dream	50.2	18.7	198.0	3775.0	female	2009

Table A.2: Penguins dataframe

REFERENCES

[1] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014.

[2] Iris Cong, Soonwon Choi, and Mikhail D Lukin. "Quantum convolutional neural networks". In: *Nat. Phys.* 15.12 (2019).

[3] Tak Hur, Leeseok Kim, and Daniel K Park. "Quantum convolutional neural network for classical data classification". In: *arXiv:2108.00661* (2021).

[4] Héctor Iván García Hernández, Raymundo Torres Ruiz, and Guo-Hua Sun. "Image classification via quantum machine learning". In: *arXiv preprint arXiv:2011.02831* (2020).

[5] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. "An introduction to quantum machine learning". In: *Contemp. Phys.* 56.2 (2015).

[6] Sean D Holcomb et al. "Overview on deepmind and its alphago zero ai". In: *Proceedings of the 2018 international conference on big data and education*. 2018.

[7] Mogens Dalgaard et al. "Global optimization of quantum dynamics with alphazero deep exploration". In: *npj Quant. Inf.* 6.1 (2020).

[8] Mohammed AlQuraishi. "AlphaFold at CASP13". In: *Bioinformatics* 35.22 (2019).

[9] Kai Arulkumaran, Antoine Cully, and Julian Togelius. "Alphastar: An evolutionary computation perspective". In: *Proceedings of the genetic and evolutionary computation conference companion*. 2019.

[10] Hartmut Neven, Geordie Rose, and William G Macready. "Image recognition with an adiabatic quantum computer I. Mapping to quadratic unconstrained binary optimization". In: *arXiv preprint arXiv:0804.4457* (2008).

[11] Ning Xie et al. “Explainable deep learning: A field guide for the uninitiated”. In: (2020).

[12] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “” Why should i trust you?” Explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016.

[13] Gabriëlle Ras, Marcel van Gerven, and Pim Haselager. “Explanation methods in deep learning: Users, values, concerns and challenges”. In: *Explainable and interpretable models in computer vision and machine learning*. Springer, 2018.

[14] Ioannis Kakogeorgiou and Konstantinos Karantzalos. “Evaluating explainable artificial intelligence methods for multi-label deep learning classification tasks in remote sensing”. In: *International Journal of Applied Earth Observation and Geoinformation* 103 (2021).

[15] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. “Distilling the knowledge in a neural network”. In: *arXiv:1503.02531* 2.7 (2015).

[16] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural machine translation by jointly learning to align and translate”. In: *arXiv preprint arXiv:1409.0473* (2014).

[17] Oana-Maria Camburu et al. “Make up your mind! adversarial generation of inconsistent natural language explanations”. In: *arXiv preprint arXiv:1910.03065* (2019).

[18] Yequan Wang et al. “Attention-based LSTM for aspect-level sentiment classification”. In: *Proceedings of the 2016 conference on empirical methods in natural language processing*. 2016, pp. 606–615.

[19] Christoph Molnar. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. 2nd ed. 2022. URL: `https://christophm.github.io/interpretable-ml-book`.

[20] Sahil Verma, John Dickerson, and Keegan Hines. “Counterfactual explanations for machine learning: A review”. In: *arXiv:2010.10596* (2020).

[21] Sandra Wachter, Brent Mittelstadt, and Chris Russell. “Counterfactual explanations without opening the black box: Automated decisions and the GDPR”. In: *Harv. JL & Tech.* 31 (2017), p. 841.

[22] Richard P Feynman et al. “Simulating physics with computers”. In: *Int. j. Theor. phys* 21.6/7 (1982).

[23] John Preskill. “Quantum computing in the NISQ era and beyond”. In: *Quantum* 2 (2018).

[24] Zewen Li et al. “A survey of convolutional neural networks: analysis, applications, and prospects”. In: *IEEE Trans NN Learn. Sys.* (2021).

[25] Shuxiang Cao et al. “Cost-function embedding and dataset encoding for machine learning with parametrized quantum circuits”. In: *Phys. Rev. A* 101.5 (2020).

[26] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorisation and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (1997). DOI: 10.1137/s0097539795293172.

[27] Johannes Jakob Meyer, Johannes Borregaard, and Jens Eisert. “A variational toolbox for quantum multi-parameter estimation”. In: *npj Quant. Inf.* 7.1 (2021).

[28] Adrian Cho. *Google claims quantum computing milestone*. 2019.

[29] Frank Arute et al. “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574.7779 (2019).

[30] Lov K Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996.

[31] David Collins, KW Kim, and WC Holton. “Deutsch-Jozsa algorithm as a test of quantum computation”. In: *Phys. Rev. A* 58.3 (1998).

[32] Edward Farhi et al. “The quantum approximate optimization algorithm and the sherrington-kirkpatrick model at infinite size”. In: *arXiv:1910.08187* (2019).

[33] Diego García-Martín and Germán Sierra. “Five experimental tests on the 5-qubit IBM quantum computer”. In: *arXiv:1712.05642* (2017).

[34] Dennis Willsch et al. “Support vector machines on the D-Wave quantum annealer”. In: *Comp. phys. comm.* 248 (2020).

[35] Andras Gilyen, Srinivasan Arunachalam, and Nathan Wiebe. “Optimizing quantum optimization algorithms via faster quantum gradient computation”. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019.

[36] Yidong Liao, Min-Hsiu Hsieh, and Chris Ferrie. “Quantum optimization for training quantum neural networks”. In: *arXiv:2103.17047* (2021).

[37] Lennart Bittel and Martin Kliesch. “Training variational quantum algorithms is np-hard”. In: *Phy. Rev. Let.* 127.12 (2021).

[38] Andrew Arrasmith et al. “Effect of barren plateaus on gradient-free optimization”. In: *Quantum* 5 (2021).

[39] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. “Effect of data encoding on the expressive power of variational quantum-machine-learning models”. In: *Phys. Rev. A* 103.3 (2021).

[40] Sankha Subhra Mukherjee, Raka Chowdhury, and Siddhartha Bhattacharyya. “Image restoration using a multilayered quantum backpropagation neural network”. In: *2011 International Conference on Computational Intelligence and Communication Networks*. IEEE. 2011.

[41] Masaya Watabe et al. “Quantum circuit parameters learning with gradient descent using backpropagation”. In: *arXiv:1910.14266* (2019).

[42] Mario Motta and Shiwei Zhang. “Computation of ground-state properties in molecular systems: Back-propagation with auxiliary-field quantum Monte Carlo”. In: *J.Chem. Th. Comp.* 13.11 (2017).

[43] Wirawan Purwanto and Shiwei Zhang. “Quantum Monte Carlo method for the ground state of many-boson systems”. In: *Phys. Rev. Lett.* 70.5 (2004).

[44] Sandro Sorella. “Wave function optimization in the variational Monte Carlo method”. In: *Physical Review B* 71.24 (2005), p. 241103.

[45] Met Office. *Iris: A Python library for analysing and visualising meteorological and oceanographic data sets*. v1.2. Exeter, Devon, 2010 - 2013. URL: http://scitools.org.uk/.

[46] Allison Marie Horst, Alison Presmanes Hill, and Kristen B Gorman. *palmerpenguins: Palmer Archipelago (Antarctica) penguin data*. R package version 0.1.0. 2020. DOI: 10.5281/zenodo.3960218. URL: https://allisonhorst.github.io/palmerpenguins/.

[47] K. L. Zhang and Z. Song. “Quantum Phase Transition in a Quantum Ising Chain at Nonzero Temperatures”. In: *Physical Review Letters* 126.11 (2021). DOI: 10.1103/physrevlett.126.116401. URL: https://doi.org/10.1103%2Fphysrevlett.126.116401.