

# Quantum Machine Learning Applied to Quantum Mastermind

Milena Hakobyan

A thesis presented for the degree of  
BS in Computer Science

Supervisor: Dr. Suren Khachatryan  
College of Science and Engineering  
American University of Armenia  
18 of May, 2024

# Quantum Machine Learning Applied to Quantum Mastermind

Milena Hakobyan

May 18, 2024

## Abstract

Quantum machine learning (QML) is drawing significant interest as a new research area, aiming to speed up and scale machine learning by leveraging the capabilities of quantum computing. This paper investigates the CQ setting, where quantum algorithms process classical data, demonstrating the synergy between classical preprocessing and quantum computation. Our study emphasizes quantum neural networks (QNNs) and their application in binary classification tasks, with a specific focus on their structure, execution, and advantages, considering the limitations posed by noisy intermediate-scale quantum (NISQ) devices. Additionally, we propose a quantum algorithm for solving the binary Mastermind problem. This algorithm combines aspects of the classical Serial algorithm with a quantum equivalent, utilizing quantum parallelism and specific bit-flip operations to narrow down the set of candidate next guesses. Additionally, it suggests a classification formulation to enhance the effectiveness of the guessing process. Through this exploration, the paper underscores the potential of current quantum approaches in both machine learning and combinatorial problem-solving within the NISQ era.

## 1 Introduction

Quantum machine learning (QML) is an interdisciplinary field that merges principles from quantum information processing and classical learning.

There are four main approaches based on how the classical (C) and quantum (Q) systems are combined to perform machine learning tasks [4]. The algorithms discussed in the paper fall under the CQ setting which refers to the case of utilizing quantum computing to process classical data. This hybrid approach aims to combine the best of both classical and quantum paradigms, applying quantum algorithms and computations where they offer a distinct advantage while still relying on classical data and preprocessing methods.

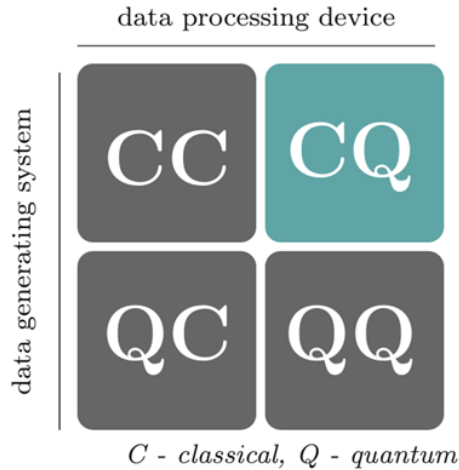


Figure 1: The four approaches combining quantum computing and machine learning [4].

Quantum computing harnesses the laws of quantum mechanics to process information in fundamentally different ways than classical computers. This includes using principles such as superposition, entanglement, tunneling, and quantum coherence, introducing novel approaches to problem-solving. The ability of quantum algorithms to outperform their classical counterparts while solving certain computation tasks is known as quantum speedup [17]. Since the 1990s, many quantum algorithms and subroutines have been discovered and optimized [19]. The “quantum breakthrough” was Shor’s factorization algorithm that provides exponential advantage over classical algorithms [12, 13, 18]. This discovery was pivotal in quantum computing and transformed the field from a relatively niche topic into a major research area. Grover’s unstructured search algorithm marked another significant milestone in quantum computing by offering quadratic speedup compared to classical alternatives [12, 13, 15].

The research in the field of QML is aimed towards using the phenomenal capabilities of quantum computing to improve and further accelerate the learning process. There have been developed quantum versions of some famous ML algorithms, such as quantum k-nearest neighbor methods, quantum support vector machines, quantum clustering, and quantum neural networks [20].

QML algorithms can efficiently explore a high-dimensional tensor-product space for solutions through quantum parallelism and interference, significantly reducing the computational time required for training and inference tasks. The drawback here is that the presence of noise and decoherence in current quantum hardware can adversely affect the accuracy of QML algorithms. As a result, testing these algorithms on real quantum computers is currently impractical. This difficulty is compounded by the complexity of error correction for quantum computers; it is a huge engineering challenge still not resolved and open to ongoing research. Acknowledging the current limitations of quantum hardware, the current state of quantum computing is described as noisy intermediate-scale quantum (NISQ) era. Intermediate-scale devices, while offering a glimpse into the potential of quantum computation, operate with a limited number of qubits and suffer from high error rates due to noise and decoherence. In addition to error correction, researchers are also investigating techniques to optimize quantum algorithms for NISQ devices. There has been promising progress in the development of quantum algorithms that have shown resilience to noise.

This paper focuses on quantum algorithms for binary classification tasks, particularly quantum neural networks (QNNs), exploring their design, implementation, and potential advantages over classical methods within the current constraints of NISQ-era devices. Additionally, we propose an algorithm to solve the binary version of the Mastermind problem, demonstrating the potential of quantum approaches in combinatorial problem-solving.

## 2 QML Review

### 2.1 Variational Quantum Algorithms and Parameterized Quantum Circuits for QML

In the last two decades, various classical-quantum hybrid models have been developed and efficiently implemented on near-term quantum devices [5]. Of these models, variational quantum algorithms (VQA) have showcased quantum supremacy in the NISQ era by providing tangible means to implement machine learning algorithms [3].

Parameterized or variational quantum circuits are the building blocks of VQAs that depend on free parameters; they are often referred to as quantum neural networks. In practice, the iterative training process of VQAs consists of two stages - executing and evaluating the circuit model on a quantum system and optimizing the parameters using classical optimization algorithms. The hybrid approach imposes significantly lower demands on the number of qubits and the depth of the circuit, as a substantial portion of the computation is offloaded to classical computers [10], thus making the training process feasible on near-term quantum devices.

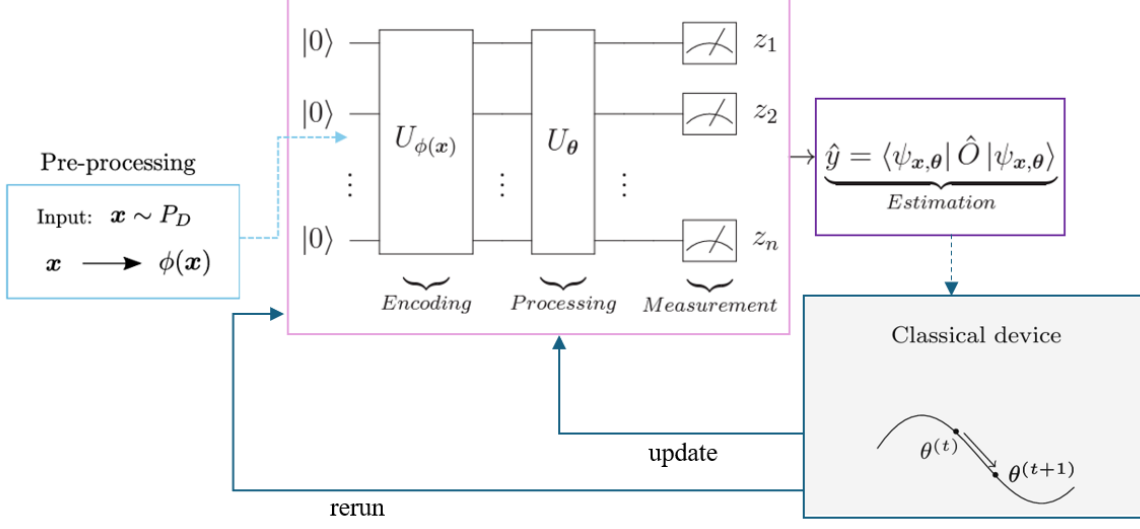


Figure 2: Hybrid quantum-classical training algorithm. The structure of PQCs.

The construction of QNNs (PQCs) typically involves three steps: feature encoding, parameterization, and measurement (Fig. 2). The feature vector  $x$  is encoded into a quantum state via state preparation routine or feature map:

$$|\psi_x\rangle := U_{\phi(x)} |0\rangle^{\otimes n}.$$

Several state preparation techniques exist, such as basis encoding, superposition encoding, angle encoding, and amplitude encoding [16]. Moreover, other quantum feature maps offer additional methods for encoding classical data, including qubit encoding, RZZ encoding [10], etc. The choice of the feature map depends on the specific problem and is aimed at improving the performance of the quantum model [11].

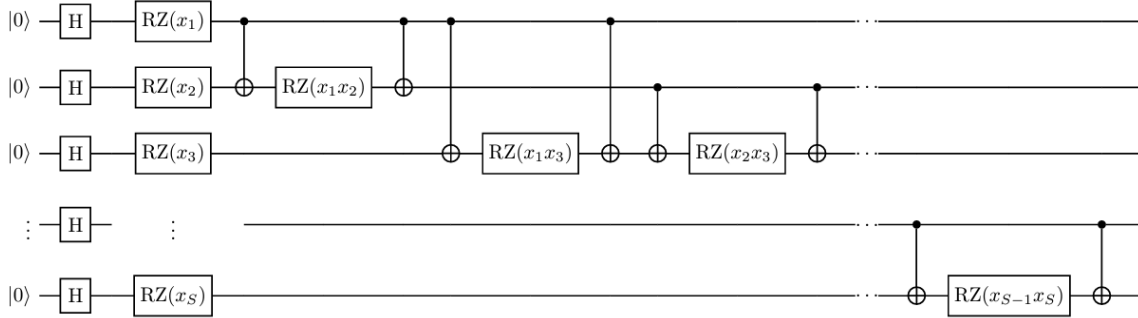


Figure 3: Example of a hardware-efficient feature map used in QNN construction [11].

Next, a parameterized gate  $U_{\theta}$  is applied to  $|\psi_x\rangle$ , where  $\theta = \{\theta_1, \theta_2, \dots, \theta_n\}$ . The resulting transformed state is the following:

$$|\psi_{x,\theta}\rangle := U_{\theta} |\psi_x\rangle.$$

The parameterized quantum circuit used in a variational model is also referred to as *ansatz*.  $U_\theta$  can be understood as a sequence of linear layers, i.e.  $\prod_{l=1}^L U(\theta^l)$ . Each layer  $U_\theta$  is typically constructed using single-qubit rotation gates such as RY and RZ, each parameterized with its own parameter, and CNOT, CZ and other controlled gates to create full entanglement within the system. While the single-qubit unitary operators perform linear transformations on quantum states, the layers of parameterized gates introduce non-linear behavior similar to that of classical neural networks.

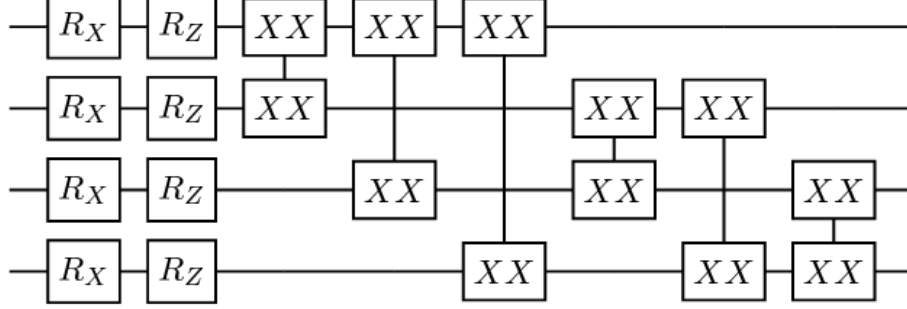


Figure 4: Example of a hardware-efficient layer used as an ansatz. This construction uses single-qubit rotations about X and Y, and a fully-connected pattern of XX entangling gates [3].

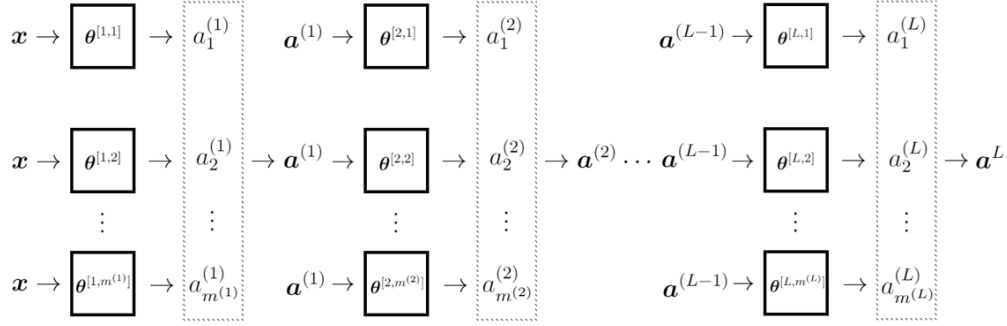


Figure 5: General structure of parameterized layers of QNNs [10].

The last step is the measurement of the circuit to obtain the model output, which is usually done by estimating the expectation value of some appropriate observable  $\hat{O}$ :

$$\hat{y} = f_{QNN}(x; \theta) = \langle \psi_{x,\theta} | \hat{O} | \psi_{x,\theta} \rangle.$$

Measurement results from parameterized quantum circuits usually undergo classical post-processing before being optimized. This classical post-processing involves interpreting the measurement outcomes, often mapping them to specific class labels or real values, depending on the task at hand. These outcomes are then utilized to compute gradients for the optimization process, guiding the adjustment of PQC parameters towards minimizing a predefined loss function.

## 2.2 Perceptron training using Grover's algorithm

Some researchers proposed quantum algorithms to solve classification tasks with the help of Grover's search algorithm. As mentioned above, Grover's algorithm offers quadratic improvement over classical algorithms for unstructured search problems. In addition, integrating a variational learning approach with the Grover search algorithm yields further quantum advantage compared to conventional Grover's algorithm [7].

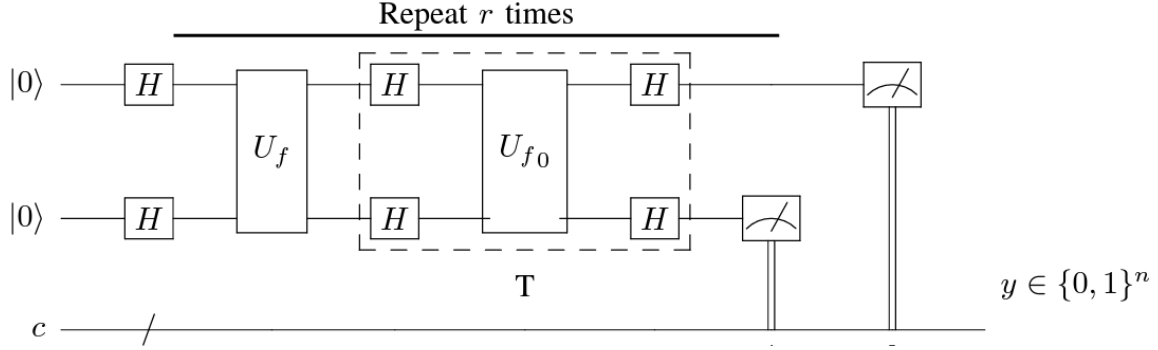


Figure 6: Quantum circuit for the general version of Grover’s algorithm [8].

In their paper, Ostroukh and Pronin demonstrate the idea of leveraging Grover’s algorithm to construct quantum circuits for neural network training [1]. Their work is focused on perceptron-training (Figure 7), where the final output is expected to be the weight values solving the inequality  $I_1 w_1 + I_2 w_2 \geq A_c$ . Here,  $I_1$  and  $I_2$  are the input features,  $w_1$  and  $w_2$  are their corresponding weights, and  $A_c$  is the known threshold value required to obtain the label “1.” The activation function is linear.

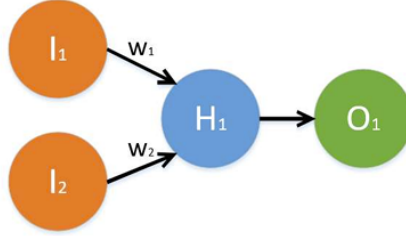


Figure 7: The perceptron example modeled using a quantum circuit[1].

To simplify the circuit, the authors reduced the feature set to a binary domain and their focus was on determining the minimum weights that meet the condition outlined in the activation function  $I_1 w_1 + I_2 w_2 = A_c$ . In the circuit, Grover’s algorithm is used to amplify the amplitudes of the target weights. Instead of applying parameterized gates on the input features, the weights are encoded as a superposition of two qubits with equal probabilities. Grover’s oracle in the circuit is designed in accordance with the characteristics of the activation function. After measurement, the correct weights will be observed with very high probability, thanks to the amplification effect of Grover’s algorithm. Figure 9 demonstrates the circuit implementation of the proposed algorithm.

This paper presented an example of how Grover’s algorithm can be employed to train neural networks, focusing on a simple perceptron example. While this approach showcases the potential of quantum algorithms in classification tasks, it is worth noting that scaling up to more complex architectures, such as multilayer neural networks, would require a significant increase in the number of qubits. However, given the current limitations of NISQ-era devices, implementing full-scale quantum neural networks remains challenging.

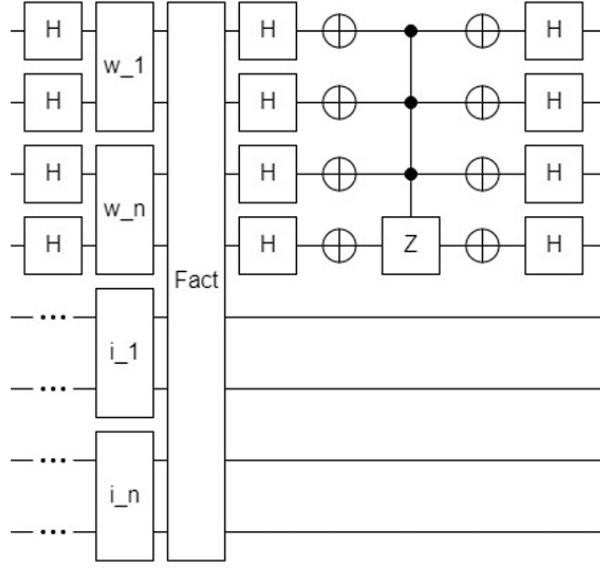


Figure 8: Quantum circuit for perceptron training based on Grover's algorithm [1].

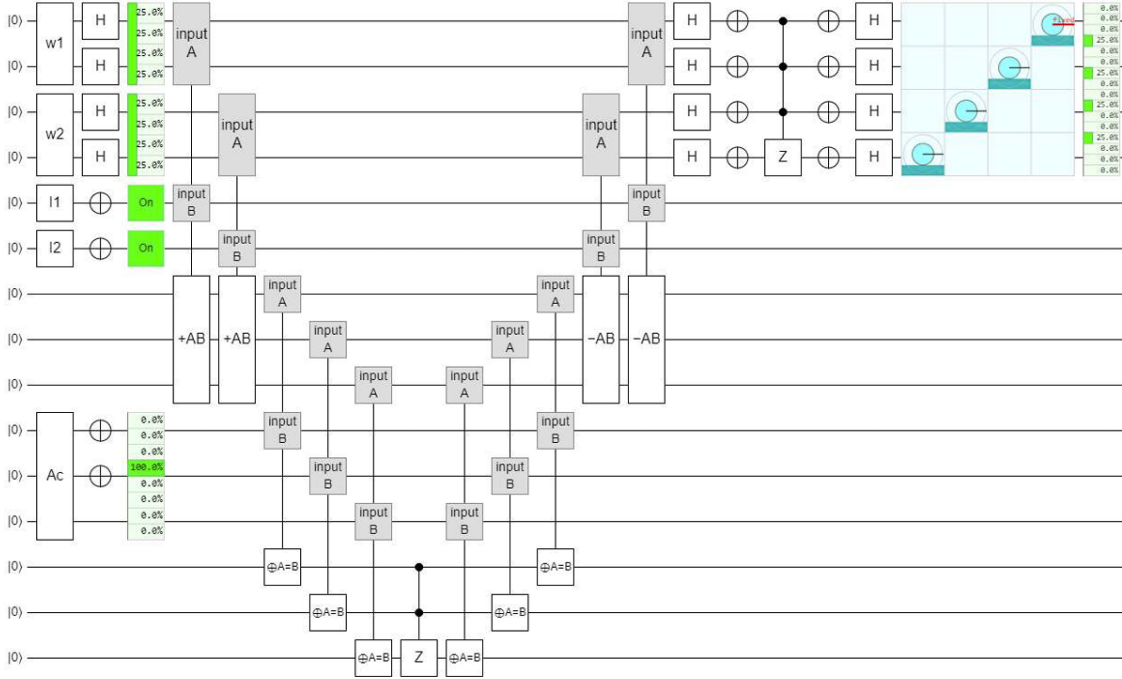


Figure 9: Perceptron training quantum circuit for  $A_c = 3$  that utilizes the principles of Grover's algorithm, based on the concept on fig. 8 [1].

### 2.3 Grover-search based quantum learning scheme for classification

Du et al. address the two main problems with quantum neural network models developed to accomplish supervised classification tasks: the lack of exploration into quantum classifiers that can strike balance between computational cost and learning performance, and the uncertainty regarding whether quantum classifiers can outperform their classical counterparts in certain problem domains. In their work, the authors tackle the aforementioned issues within the NISQ setting by formulating the classification task as a search problem.

The proposed Grover-search based quantum learning scheme for binary classification (GBLS) is a flexible and efficient learning structure that allows optimization of various quantum classifiers with different batch sizes. Together with batch processing, the hybrid nature of the learning scheme ensures a reduction in the number of measurements required. Through numerical experiments, the authors prove that in optimal settings, GBLS can achieve quadratic reduction in query complexity with competitive performance compared to classical counterparts when applied to some binary classification tasks [6].

Solving a binary classification task implies developing a model capable of accurately predicting the class labels of the input data points, after the training on a dataset  $\hat{D} = \{(x_i, y_i)_{i=1}^{N-1} \in R^{N \times M}, \{0, 1\}^N\}$ . Before the GBLS circuit is implemented, the data undergoes a custom pre-processing stage. First, two subsets  $D(0)$  and  $D(1)$  are created, each only containing points from the dataset with labels '0' and '1' respectively ( $D^{(0)} \cup D^{(1)} = \hat{D}$ ). Then, depending on the batch size  $K$ , an extended dataset  $\hat{D}$  is created, where each datapoint is enlarged to a vector of size  $K$ :

$$\mathcal{D}_k = [(\mathbf{x}_k^{(0)}, y_k^{(0)}), (\mathbf{x}_k^{(1)}, y_k^{(1)}), \dots, (\mathbf{x}_k^{(K-1)}, y_k^{(K-1)})]$$

The last pair in each vector  $D_k$  corresponds to the  $k$ -th example of the original dataset and has label  $y_k \in \{0, 1\}$ , while the first  $K-1$  pairs are uniformly sampled from the subset with opposite label ( $D^{(1-k)}$ ). If, for example, the  $k$ -th example has label '1', the last element is the  $k$ -th example itself and the first  $K-1$  pairs are sampled from  $D^{(0)}$ .

After the data has been pre-processed, it is encoded into a quantum state through one of the mapping techniques discussed above. The quantum transformation of the  $k$ -th datapoint  $D_k$  is described as:

$$U_{data} |\mathbf{0}\rangle := |\Phi^k\rangle_{F,I} = \frac{1}{\sqrt{K}} \sum_{i=0}^{K-1} |h(\mathbf{x}_i)\rangle_F |i\rangle_I$$

Here, 'F' and 'I' correspond to the feature and index registers of the circuit and  $h$  is the encoding function.

Parameterization of the circuit is performed via  $U_{L1}$  - a sequence of parameterized layers. Based on the label of  $D_k$ ,  $U_{L1}$  applies different transformations. If  $y_k = 1$ , the first qubit of the feature register ( $|\psi_i^{y_k}\rangle$ ) becomes  $|1\rangle$ :

$$(U_{L1} \otimes \mathbb{I}) |\Phi^k(y_k = 1)\rangle_{F,I} = \frac{1}{\sqrt{K}} \sum_{i=0}^{K-1} |\psi_i^{(1)}\rangle_F |i\rangle_I$$

Else, if  $y_k = 0$ ,  $|\psi_i^{y_k}\rangle$  becomes  $|0\rangle$

$$(U_{L1} \otimes \mathbb{I}) |\Phi^k(y_k = 0)\rangle_{F,I} = \frac{1}{\sqrt{K}} \sum_{i=0}^{K-1} |\psi_i^{(0)}\rangle_F |i\rangle_I$$

So,  $|\psi_i^{y_k}\rangle$  acts as a flag qubit to guide the subsequent Grover-like operations. The goal is to learn a hyperplane that separates the last pair in  $D_k$  with label  $y_k$  from datapoints of the opposite class. A multi-controlled-Z (MCZ) gate, controlled by the flag qubit and the whole index register, is applied, followed by  $U_{L1}^\dagger$  and  $U_{data}^\dagger$  gates that reverse the feature register to its original state. Lastly, the Grover operator  $U_{init}$  is applied, completing one cycle of GBLS training:



$$(U_{L_1} \otimes \mathbb{I}) |\Phi^k(y_k = 0)\rangle_{F,I} = \frac{1}{\sqrt{K}} \sum_{i=0}^{K-1} |\psi_i^{(0)}\rangle_F |i\rangle_I$$

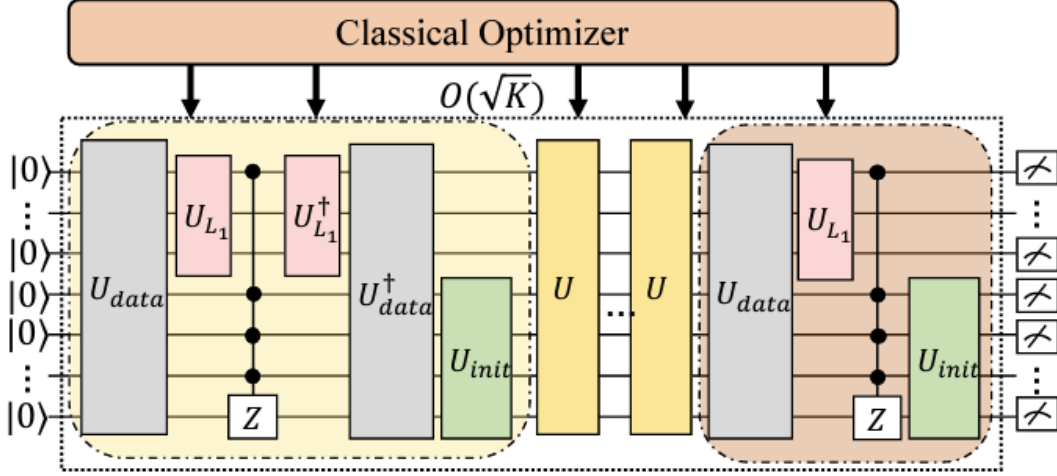


Figure 10: The paradigm of GBLS [5].

After completing the training of GBLS, the adjusted parameterized gate  $U_{L_1}$  can be utilized directly to predict the labels of unseen data points with constant query complexity.

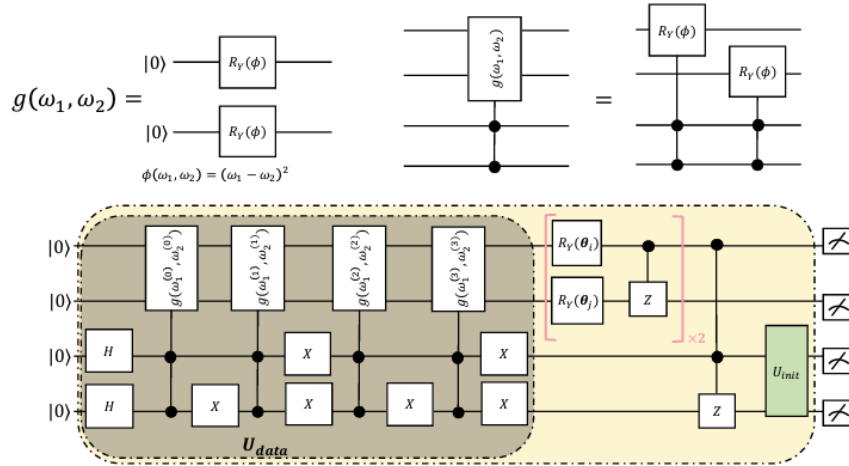


Figure 11: Circuit example for batch size  $K = 4$ .  $g(w1, w2)$  is the mapping function and the grey region illustrates the process of encoding multiple features into a quantum state using NOT and controlled gates. The number of parameterized layers is 2 [6].

The results of the conducted numerical experiments in the paper were on par with established benchmarks and proved GBLS to be comparable with other quantum classifiers while requiring a reduced number of measurements.

## 2.4 Quantum Discriminator for Binary Classification

Another novel approach for binary classification is proposed by Date and Smith. The designed Quantum Discriminator is a hybrid quantum-classical model that harnesses the capability of quantum computers to function effectively within high-dimensional spaces. The training process operates in

$\mathcal{O}(N \log N)$  time and requires  $\mathcal{O}(N \log N)$  classical bits, while the inference step exhibits linear complexity, showcasing favorable computational efficiency. Both steps require  $\mathcal{O}(\log n)$  qubits [8].

The algorithm involves two steps: feature extraction, and classification utilizing a discriminant function.

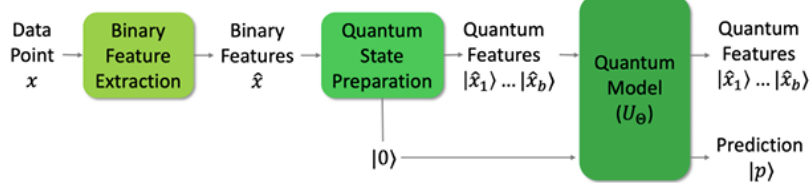


Figure 12: Model workflow [8].

Figure 12 outlines the workflow of the discriminator. Given a datapoint, a domain-specific feature extraction technique is applied to obtain a binary feature set. For example, Histogram of Oriented Gradients and Scale-Invariant Feature Transform are commonly used in computer vision applications. The binary feature extraction may also incorporate Principal Component Analysis for dimensionality reduction. Since the data is translated into binary domain, basis encoding, which is the most straightforward state preparation routine, can directly map classical bits to qubits using the computational basis states  $|0\rangle$  and  $|1\rangle$ . The algorithm will be presented and thoroughly explained in the proceeding section.

### 3 Quantum Mastermind

Mastermind is a classic code-breaking game that challenges players to deduce a hidden sequence of symbols within a limited number of attempts. Given  $n$  different colors, the first player – the keeper – secretly forms a sequence of  $n$  colored pins, where several pins may share the same color, or all of them may be of different colors. The task of the second player – the guesser – is to disclose the hidden sequence with minimal guesses [21].

For each guess, the keeper provides a score that corresponds to the number of pins that are both the correct color and in the correct position. This feedback helps the guesser refine their subsequent guesses, aiming to accurately determine the keeper’s sequence as efficiently as possible.

In the binary version of Mastermind, the principles remain the same, but the colors are replaced with binary digits (0s and 1s), simplifying the game while retaining the core challenge of code-breaking through logical deduction.

### Quantum Mastermind Algorithm

Before transitioning to the classification formulation of the binary Mastermind problem, the focus was on designing a quantum-equivalent of the classical Serial algorithm [21]. The circuit designed to solve the problem receives the guess and its score as input, and has a register that, after applying Hadamard gates, becomes a superposition of all candidate next guesses. After getting the input, the circuit calculates the bitwise NOT-XORs of the guess and each candidate, based on which corresponding scores are calculated and kept in the register allocated for the candidates’ scores. In the Serial algorithm, once the scores of the candidate next guesses are calculated, only the guesses with the same score as the current guess are kept for further consideration. Given that quantum states do not operate like classical array lists, straightforward removal of an element is impossible. A creative strategy involves calculating the difference between the scores of the guess and each candidate, then performing bit-flip operations based on the obtained value and the already computed NOT-XORs. If the difference is positive, the number of qubits equal to the difference are flipped among the matching qubits, starting

from the left. If the difference is negative, the number of qubits equal to the absolute value of the difference are flipped among the non-matching qubits, again starting from the left. By implementing these bit-flips in the candidates' register, each score is adjusted to match that of the current guess. The superposition is measured at this step, and a next guess is proposed.

## Algorithm Analysis

The design of the quantum algorithm encountered specific challenges. While choosing certain candidates as the next guess effectively solves the problem, others can lead to a deadlock. In such deadlock scenarios, repeating the bit-flip operations for the new guess fails to reduce the superposition to a smaller set. Experiments were conducted on 4-bit sequences. When a solution-leading choice was made as the second guess, the 6-term superposition was reduced to 4 terms (Fig.12(a)). However, in deadlock cases, repeating the bit-flip operations for the new guess did not change the set size (Fig.12(b)).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	A	E	A	C	AD	
1	hidden num										candits		score		not-xor						candits		score		not-xor		ampl						
2	0	0	0	1							0	0	0	0	3	1	1	0	1		0	0	0	1	2	1	1	0	0	2	further reduction		
3											0	0	0	1	2	1	1	0	0		1	1	1	0	2	0	0	1	1	2			
4	guess		score								0	0	1	0	4	1	1	1	1		0	1	0	0	2	1	0	0	1	2			
5	0	0	1	0	2	1	1	0	0		0	0	1	1	3	1	1	1	0		1	0	1	1	2	0	1	1	0	2			
6											0	1	0	0	2	1	0	0	1														
7	all trials										0	1	0	1	1	1	0	0	0														
8	1	0	0	0							0	1	1	0	3	1	0	1	1														
9	0	0	1	0							0	1	1	1	2	1	0	1	0														
10											1	0	0	0	2	0	1	0	1														
11											1	0	0	1	1	0	1	0	0														
12	0	0	1	0	solution						1	0	1	0	3	0	1	1	1														
13	1	1	0	1							1	0	1	1	2	0	1	1	0														
14	1	0	1	1	deadlock						1	1	0	0	1	0	0	0	1														
15	0	1	0	0							1	1	0	1	0	0	0	0	0														

(a)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	A	E	A	C	AD	AE
1	hidden num										candits		score		not-xor						candits		score		not-xor		ampl						
2	0	0	0	1							0	0	0	0	3	1	0	1	1		0	0	0	1	2	1	0	1	0	1	no further reduction obtained		
3											0	0	0	1	2	1	0	1	0		0	0	1	0	2	1	0	0	1	1			
4	guess		score								0	0	1	0	2	1	0	0	1		1	0	0	0	2	0	0	1	1	2			
5	0	1	0	0	2	1	0	1	0		0	0	1	1	1	1	0	0	0		0	1	1	1	2	1	1	0	0	2			
6											0	1	0	0	4	1	1	1	1		1	1	0	1	2	0	1	1	0	1			
7	all trials										0	1	0	1	3	1	1	1	0		1	1	1	0	2	0	1	0	1	1			
8	1	0	0	0							0	1	1	0	3	1	1	0	1														
9	0	1	0	0							0	1	1	1	2	1	1	0	0														
10											1	0	0	0	2	0	0	1	1														
11	0	0	1	0	solution						1	0	0	1	1	0	0	1	0														
12	1	1	0	1							1	0	1	0	1	0	0	0	1														
13	1	0	1	1	deadlock						1	0	1	1	0	0	0	0	0														
14	0	1	0	0							1	1	0	0	3	0	1	1	1														
15											1	1	0	1	2	0	1	1	0														
16											1	1	1	0	2	0	1	0	1														

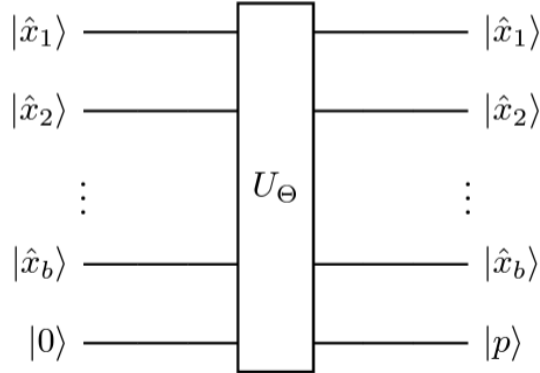
(b)

Figure 13: (a) A candidate guess leading to a solution. (b) A candidate guess leading to deadlock.

## The classification problem arising in the Quantum Mastermind Algorithm

We propose partitioning the second-guess candidates into two distinct classes: Class 0, representing candidates that may potentially lead to a deadlock, and Class 1, indicating candidates likely to lead to a solution. Throughout the training stage, we utilize input data consisting of fixed first guesses paired with candidate next guesses, forming (guess, candidate) pairs. These pairs are associated with binary labels (0 or 1), representing their classification. Leveraging this dataset, we adjust the parameters of the circuit to accurately label datapoints for other first guesses, thereby facilitating effective decision-making in subsequent problem-solving scenarios. Through this approach, we aim to enhance the effectiveness and reliability of the second guess selection process, thereby improving overall problem-solving outcomes.

The classification algorithm can be implemented by drawing from the concept and workflow of the quantum discriminator. The quantum discriminator matrix introduced in Date and Smith's paper is presented in Figure 14(a). The unitary matrix  $\Theta$  is parameterized by the parameter set  $\Theta = \{\theta_1, \theta_2, \dots, \theta_B\}$  ( $\theta_i \in \{0, 1\}, \forall i = 1, \dots, B$ ). Here,  $B = 2^b$  is the number of unique states that can be attained using  $b$  qubits, with  $b$  indicating the dimension of the datapoints in the dataset.



(a)

$$U_{\Theta} = \begin{bmatrix} 1 - \theta_1 & \theta_1 & 0 & 0 & \dots & 0 & 0 \\ \theta_1 & 1 - \theta_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 - \theta_2 & \theta_2 & \dots & 0 & 0 \\ 0 & 0 & \theta_2 & 1 - \theta_2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 - \theta_B & \theta_B \\ 0 & 0 & 0 & 0 & \dots & \theta_B & 1 - \theta_B \end{bmatrix}$$

(b)

Figure 14: (a) The quantum discriminator and (b) The parameterized matrix used in the algorithm [8].

The algorithm performing the classification task is straightforward. The inputs are the binary feature vectors and their labels. The parameter vector is initialized as the zero vector, and a quantum circuit of size  $b+1$  is set up to accommodate the  $b$  features of each point, along with an additional qubit in state  $|0\rangle$ . So, the initial input state  $|\hat{x}\rangle$  is in a superposition comprising all  $2B$  potential states, meaning each feature set is represented twice -  $|\hat{x}0\rangle$  and  $|\hat{x}1\rangle$ . The algorithm processes each vector from the training set individually. If a feature vector belongs to Class 1, the associated parameter is updated, so the corresponding sub-matrix in  $U$  becomes the Pauli-X gate. The output of measuring the prediction label will be  $|1\rangle$ . If the vector belongs to class 0, the sub-matrix remains the same with no parameter update, thus the measurement result will be  $|0\rangle$ .

In the paper, numerical experiments were conducted on the Iris dataset, examining training sizes of  $N = 80$ ,  $N = 8$  and  $N = 4$ . For  $N = 80$ , the average validation accuracy reached 99.15% using the QASM simulator but dropped to 89.13% on the IBM Jakarta processor. Reducing the training set to  $N = 8$  resulted in an average validation accuracy of 94.98% in simulation and 82.37% on the Jakarta processor. With  $N = 4$ , the average model accuracy was 84.41% across 600 trials, showcasing the potential for achieving high accuracy with a small training set. Additionally, false negative rates averaged 0.3074, with an average recall of 0.6925, indicating promising results for classification performance. The practical assessment of the algorithm showed that the discriminator is capable of generating highly accurate and precise models for separable datasets [8]. As such, it can be viewed as a promising candidate for implementing the classification component of the quantum Mastermind algorithm.

## 4 Conclusions and Future work

This study explored several algorithms in quantum machine learning (QML) tailored for binary classification tasks. We worked on an algorithm and managed to formulate a part of it as a classification problem and implement it with the Quantum Discriminator. Both parts of the algorithm had been simulated in a classical environment with limited computational resources using Python. The performance of the learning algorithm was estimated by using a sequence of shortcuts arising by the symmetry of the problem. For future work, we propose conducting full-scale simulations. Even when considering a relatively small version of Mastermind involving 4-bit sequences, implementing the classification part will demand a register of at least four qubits to store the guess, another 4-qubit register to represent the candidates, and an additional qubit for the label. Consequently, each datapoint will necessitate a minimum of nine qubits, resulting in quantum gates of size  $2^9 \times 2^9$ . These requirements highlight the challenges posed by current NISQ devices.

## 5 References

- [1] C. B. Pronin and A. V. Ostroukh, “Development of quantum circuits for perceptron neural network training, based on the principles of Grover’s algorithm,” arXiv.org, Oct. 15, 2021. <https://arxiv.org/abs/2110.09891>
- [2] K. Beer, “Quantum neural networks,” arXiv.org, May 2022, doi: 10.15488/11896.
- [3] M. Benedetti, E. Lloyd, S. Sack, and M. Fiorentini, “Parameterized quantum circuits as machine learning models,” Quantum Science and Technology, vol. 4, no. 4, p. 043001, Nov. 2019, doi: 10.1088/2058-9565/ab4eb5.
- [4] M. Schuld and F. Petruccione, (n.d.). Supervised Learning with Quantum Computers. Springer. <https://link.springer.com/book/10.1007/978-3-319-96424-9>
- [5] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, “Circuit-centric quantum classifiers,” Physical Review. A/Physical Review, A, vol. 101, no. 3, Mar. 2020, doi: 10.1103/physreva.101.032308.
- [6] Y. Du, M.-H. Hsieh, T. Liu, and D. Tao, “A Grover-search based quantum learning scheme for classification,” New Journal of Physics, vol. 23, no. 2, p. 023020, feb 2021. [Online]. <https://doi.org/10.1088/1367-2630/abdefa>
- [7] M. E. S. Morales, T. Tlyachev, and J. Biamonte, “Variational learning of Grover’s quantum search algorithm,” Physical Review. A/Physical Review, A, vol. 98, no. 6, Dec. 2018, doi: 10.1103/physreva.98.062333, <https://doi.org/10.48550/arXiv.1805.09337>
- [8] P. Date and W. Smith, “Quantum discriminator for binary classification,” arXiv.org, Sep. 02, 2020. <https://arxiv.org/abs/2009.01235>
- [9] B. Khanal, P. Rivas, J. Orduz, and A. Zhakubayev, “Quantum Machine Learning: a case study of Grover’s algorithm,” IEEE Conference Publication — IEEE Xplore. <https://ieeexplore.ieee.org/document/9798970>
- [10] K. Wold, “Parameterized Quantum Circuits for Machine Learning,” MA thesis, University of Oslo, 2021. [Online]. Available: <https://www.duo.uio.no/bitstream/handle/10852/89534/Thesis05092021.pdf?sequence=1&isAllowed=y>
- [11] A. Abbas, D. Sutter, C. Zoufal, A. Lucchi, A. Figalli, and S. Woerner, “The power of quantum neural networks,” Nature Computational Science, vol. 1, no. 6, pp. 403–409, Jun. 2021, doi: 10.1038/s43588-021-00084-1, <https://arxiv.org/abs/2011.00027>
- [12] A. Ekert, P. Hayden, and H. Inamori, “Basic concepts in Quantum Computation,” in Springer eBooks, 2007, pp. 661–701. doi: 10.1007/3-540-45338-5\_10, <https://doi.org/10.48550/arXiv.quant-ph/0011013>
- [13] E. Rieffel and W. Polak, Quantum Computing: A Gentle Introduction. Massachusetts Institute of Technology, 2011. [Online]. Available: <https://archive.org/details/quantumcomputing0000rief>
- [14] W. Zhao, Y. Wang, Y. Qu, H. Ma, and S. Wang, “Binary Classification Quantum Neural Network model based on optimized Grover algorithm,” Entropy, vol. 24, no. 12, p. 1783, Dec. 2022, doi: 10.3390/e24121783, <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9777537/>
- [15] “Grover’s algorithm — IBM Quantum Learning.” <https://learning.quantum.ibm.com/course/fundamentals-of-quantum-algorithms/grovers-algorithm>
- [16] M. Rath and H. Date, “Quantum Data Encoding: A comparative analysis of Classical-to-Quantum mapping techniques and their impact on machine learning accuracy,” arXiv.org, Nov. 17, 2023. <https://arxiv.org/abs/2311.10375>

[17] J. D. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, “Quantum machine learning,” *Nature*, vol. 549, no. 7671, pp. 195–202, Sep. 2017, doi: 10.1038/nature23474, <https://arxiv.org/abs/1611.09347>

[18] “Phase-estimation and factoring — IBM Quantum Learning.” <https://learning.quantum.ibm.com/course/fundamentals-of-quantum-algorithms/phase-estimation-and-factoring>

[19] S. Jordan, “Quantum Algorithm Zoo.” <https://quantumalgorithmzoo.org/>

[20] M. Schuld, I. Sinayskiy, and F. Petruccione, “An introduction to quantum machine learning,” *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, Oct. 2014, doi: 10.1080/00107514.2014.964942

[21] S. Jilavyan, “Parallel Implementation of Master Mind and Duty Scheduling,” MA thesis, American University of Armenia, 2005.