




Supercomputing leverages quantum machine learning and Grover's algorithm

Bikram Khanal¹ · Javier Orduz² · Pablo Rivas¹  · Erich Baker¹

Accepted: 30 October 2022 / Published online: 17 November 2022

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract

The complexity of searching algorithms in classical computing is a classic problem and a research area. Quantum computers and quantum algorithms can efficiently compute some classically hard problems. In addition, quantum machine learning algorithms could be an important avenue to boost existing and new quantum-based technology, reducing the supercomputing requirements for executing such problems. This paper reviews and explores topics such as variational quantum algorithms, kernel methods, and Grover's algorithm (GA). GA is a quantum search algorithm that achieves a quadratic speed improvement as a quantum classifier. We exploit GA or amplitude amplification to simulate rudimentary classical logical gates into quantum circuits considering AND, XOR, and OR gates. Our experiments in our review suggest that the algorithms discussed can be implemented and verified with relative ease, suggesting that researchers can investigate problems in the areas discussed related to quantum machine learning and more.

✉ Pablo Rivas
Pablo_Rivas@baylor.edu

Bikram Khanal
Bikram_Khanal1@baylor.edu

Javier Orduz
orduzja@earlham.edu

Erich Baker
Erich_Baker@baylor.edu

¹ Department of Computer Science, Baylor University, One Bear Place #97141, Waco, Texas 76712-7141, USA

² Department of Mathematics and Computer Science, Earlham College, 801 W Nat'l Rd, Richmond, IN 47374, USA

Keywords Quantum computing · Quantum machine learning · Grover's search algorithm · Variational quantum circuit classifier

1 Introduction

Quantum mechanics is a framework for understanding micro-universe. At a low scale of distance, we find several counter-intuitive phenomena, and physicists developed a framework to understand this universe, which is, Quantum mechanics. This framework provides information on a particle's state described by a wave function, labeled as $\psi(x, t)$. The Schrödinger equation describes the time evolution of this wave function, which contains all available information about the state [1]:

$$i\hbar \frac{\partial |\psi(t)\rangle}{\partial t} = \hat{H}(t) |\psi(t)\rangle, \quad (1)$$

where i is the imaginary unit, $\hbar = 1.054 \times 10^{-34}$ J s, which is a Planck's constant, $|\psi\rangle(t)$ is time-dependent state. Also, $\hat{H}(t)$ is the Hamiltonian operator which, for general purposes, represents the energy of the system. This formulation requires particular analysis due the notation, and mathematical concepts. In addition, we consider that physical magnitudes are in the real domain, \mathbb{R} , but the magnitudes in Eq. (1) are given by complex numbers, \mathbb{C} ; therefore, we should talk about new ways to define some physical concepts.

Quantum computing (QC) has a good formalization through mathematics and physics frameworks. QC is relevant in different fields, such as classical fields theory [2–5], computational security with quantum algorithms [6], physics [7], chemistry [8], biology [9], and learning from data as image recognition [10], and neural networks [11]. There are more contributions on medical and genetics.

Today, we find a quantum word in different contexts and systems. Some of these system require more analysis. Therefore, we share details and classification of different systems. Topological Quantum computer implements a finite-dimensional internal state space with no natural tensor product structure and in which the evolution of the state is discrete, namely the local Hamiltonian (H) is zero, $\hat{H} = 0$. This function has the same definition as in (1). It has good stability to create trapped quantum particles [12]. One-way quantum computer (or cluster state) prepares a cluster (entangled, graph) state performs single qubit measurements on it [13]. Quantum Turing machine computer implements an abstract machine to model the effects of a quantum computer. This is also called a universal quantum computer model which captures all of the power of quantum computation [14]. A quantum universal gate (QUG) model is a sequence of reversible transformation which are represented by gates, a.k.a. quantum gates [15]. The graphical depiction of quantum circuit elements is described using a variant of the Penrose graphical notation [16]. A quantum circuit is a representation of a quantum operation that is performed sequentially. Logic qubits are transported on wires (shown by horizontal lines), and quantum gates (represented by blocks) act on the qubits in a typical quantum circuit. The logical gate is a device that controls or processes data;

e.g., the Hadamard H and NOT X gates. Adiabatic quantum computation. This model relies on the adiabatic theorem and is closely related to quantum annealing [17]. Quantum machine learning (QML) is the integration of quantum algorithms within machine learning concepts. QML usually refers to ML algorithms for the analysis of classical data executed on a quantum computer, quantum-enhanced machine learning [17, 18]. We further recently found other classification based on speedup [9]. Exponential speedup. This model requires a quantum computer to make a better-quality approximation computationally tractable. In general, this is thought for high dimension, or degree of freedom systems. Polynomial speedup. This kind models do not usually need more qubits than are already needed to do calculations. Heuristic speedups. This model has great potential to be explored when quantum computers are working. We expect that our understanding of the performance of these heuristics will improve since the speedup level has some unknown issues. Interfacing with classical algorithms. This kind of paradigm has limitations rely on the ability to share the quantum and the classical computer. Big data and quantum RAM. This model has limitations associated with the superposition concept, and with the large dataset.

All above models are one scheme to perceive the quantum computing area, but this is an area with more details, and we will probably have other similar sketch soon.

This area and its paradigms such as adiabatic quantum computer, and QML can improve areas such as cybersecurity and cryptography which impacts in our communication systems. In addition, quantum algorithms can improve the performance of supercomputing. QC is one of the most popular concepts in the last decade, and we expect significant results and revolutionary ideas in this century.

As was discussed, QC has different models which can harness the laws of quantum mechanics to process information. We conceive supercomputing as the process of doing complex and large calculations using supercomputers. These supercomputers can use parallel processing. We can perform computations based on some quantum mechanical concepts such as superposition, entanglement, teleportation, and Dirac notation, among others to show the possible power of QC over classical computation that can possibly reduce the load in supercomputers. Many theoretical works prove a quantum advantage. However, current quantum devices are not at the stage to reflect these improvements in practices. We aim to review such approaches in QC and discuss possible implementations briefly.

In this paper, we review the Grover's algorithm and quantum machine learning. Our work is motivated by the amplitude amplification associated with Grover's Algorithms. We exploit amplitude amplification to propose a method to simulate classical logical circuits into quantum circuits. We use the CNOT and Toffoli gates to construct *AND*, *OR*, and *XOR* gates. In this review, we showed that we could construct an oracle for Grover's algorithm for provided classical circuit and calculate the initial input states such that the output of the classical circuit is one with high probability.

This paper contains the following sections: Context, Models, and Methods (Sect. 2) with a description of different quantum computing paradigms. We present few algorithms in the machine learning context with their performance. In

particular, this section discusses Grover's algorithm and its implementation. We share the implementation code with Qiskit [19], in python [20, 21]. We present related works in Grover's algorithm with brief introduction on variational quantum algorithm and kernel methods in Sect. 3. The experimental setup, proposed methods and algorithm is described in Sect. 4. We present results and discussion in Sect. 5, and future works and conclusions are drawn in Sect. 6.

2 Context and models

As mentioned in Sect. 1, there are different branches in QC. QC is an active research field, and we can expect regular changes. However, for the scope of this paper, we consider two paradigms: quantum machine learning and quantum search algorithms. Some relevant algorithms for QC, such as Deutsch, Bernstein–Vazirani, Simon, Grover, and Shor will be discussed in Sect. 2.2. The following subsection presents the quantum computing topics explored in this paper.

2.1 Quantum computing

Quantum computing is an application of quantum theory calculus to calculate the probabilities of the output of measurements on physical system [22]. Quantum algorithms implement transformations which are matrices, in terms of Linear Algebra, and those have their particular (Dirac) representation [23]. Each transformation requires an operator to create superposition, rotation (on state), or another change on the system. Operator act on states. Each of those states has the form $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$, which is the general state in the $\{|0\rangle, |1\rangle\}$ basis, where $\{\alpha, \beta\}$ are complex numbers, a.k.a. amplitudes [24]. In quantum circuit model, one operator, namely acting on states, has the form:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{2}$$

$$H = \frac{1}{\sqrt{2}} \left(|0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1| \right). \tag{3}$$

Where $|1\rangle = (0 \ 1)^T$, $|0\rangle = (1 \ 0)^T$, and $\langle 1| = |1\rangle^\dagger$, $\langle 0| = |0\rangle^\dagger$ live in the dual space. The product $|k\rangle\langle l|$ is an operator, $k, l \in \{0, 1\}$, and the $\frac{1}{\sqrt{2}}$ is a normalization factor. Equation. (2) is the matrix representation of Hadamard gate and (3) is its Dirac (bracket) representation. H transforms single-qubit as $|\psi\rangle$. Nonetheless, we can propose a generalization, it means if we apply H on each qubit in a system with N qubits, the system is now on superposition.

Table 1 Some quantum algorithms, mapping and a brief function description

Problem	Maps	Function
Deutsch	$f : \{0, 1\} \rightarrow \{0, 1\}$	$f(x)$ balanced and constant
Deutsch–Jozsa	$f : \{0, 1\}^n \rightarrow \{0, 1\}$	Black box oracle function.
Bernstein–Vazirano	$f : \{0, 1\}^n \rightarrow \{0, 1\}$	$f(x) = a \cdot x$
Simon	$f : \{0, 1\}^n \rightarrow \{0, 1\}^n$	$f(x) = a \oplus x$

Table 2 This table was inspired by ref. [25]. The column headed “Algorithm” refers the classical learning method. The column headed “Speedup” lists how much faster (if any) the quantum variant is compared with the best known classical version. And the column headed “Generalization Performance” indicates whether this quality of the learning algorithm was studied in the relevant articles

Algorithm	Speedup	Generalization performance
K-medians	Quadratic	No
Hierarchical clustering	Quadratic	No
K-means	Exponential	No
Principal components	Exponential	No
Associative memory		No
Neural Networks		Yes/Numerical
SVM	Quadratic/Exponential	Yes/Analytical
Nearest neighbors	Quadratic	Yes/Numerical
Regression	Quadratic	No
Boosting	Quadratic	Yes/Yes/Numerical

2.2 Quantum algorithms

Quantum algorithms can be of two types: those to run on quantum computers, or to run on classical computer with quantum concepts executed as subroutine in quantum device. Regardless the type, these algorithms should improve speedup or processing of information.

Table 1 shows four basic quantum algorithms, their mappings, and their functions.

Where Deutsch function maps from one qubit to one qubit, Deutsch–Jozsa and Bernstein–Vazirano function means maps from n qubits to one qubits, and Simon maps n qubits to one n qubits. On the other hand, \cdot and \oplus are product modulo 2, and exclusive OR, respectively.

Table 2 presents ten classical algorithms that are proven to achieve speedup with quantum computing.

Quantum algorithms are those algorithms that run on a quantum computer. These algorithms achieve performance or efficiency improvements over any classical counterparts. Quantum algorithms and their applications include cryptography, medicine, search and optimization, solving linear equations, and simulating quantum systems [26]. Shor’s algorithm was one of the first algorithms to deliver the application of quantum computers [27]. Given an integer $N = a \times b$, where a and b are the prime numbers, Shor’s algorithm factorizes this problem in $O(\log N)^3$ complexity. Most cryptography relies on the difficulty of integer factorization. Shor’s algorithm implies that

these systems are not safe against large quantum computer attacks. Possible applications and mathematical explanations of Shor's algorithm are beyond this paper's scope. Please refer [27–29] for detailed explanations and discussion on possible applications. Harrow et al. first proposed a quantum algorithm for linear system [30]. Deutsch and Jozsa algorithm make a single query to determine whether a function from \mathbb{Z}_2^n to \mathbb{Z}_2 is balanced or constant [23]. For this problem, any classical algorithm requires at least two function queries. Numerous quantum machine learning algorithms and applications have been proposed in recent years. In [31–34] the authors describe some of those algorithms with data encoding techniques, measurements, and their applications.

Grover's algorithm enables us to find an item x from an unstructured dataset of N item with $O(\sqrt{N})$ operations [35–37]. With the algorithm shown in Fig. 1, the goal is to find w , given an oracle U_f with $f : \{0, 1\}^n \rightarrow \{0, 1\}$,

$$f(x) = \begin{cases} 1 & \text{if } x = w \\ 0 & \text{else if,} \end{cases}$$

and

$$o(x) = \begin{cases} 0 & \text{if } x = 000\dots0 \\ 1 & \text{else,} \end{cases}$$

where the phase oracle is

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle,$$

where

$$U_f : \begin{cases} |w\rangle \rightarrow -|w\rangle \\ |x\rangle \rightarrow |x\rangle \end{cases} \quad \forall x \neq w.$$

Then

$$U_f = 1 - 2 |w\rangle \langle w|,$$

and

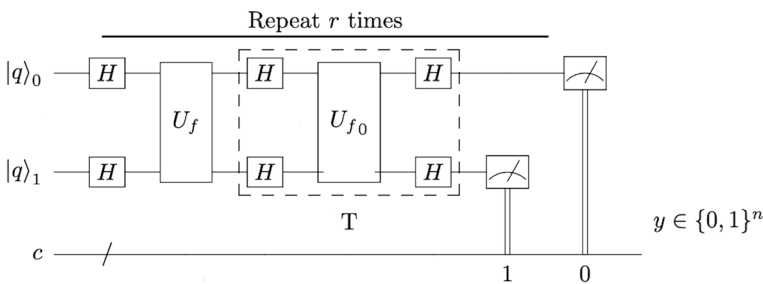


Fig. 1 Quantum circuit for the general version of Grover algorithm

$$U_{f_0} : \begin{cases} |0\rangle \rightarrow |0\rangle^{\otimes n} \\ |x\rangle \rightarrow -|x\rangle \quad \forall x \neq 00\dots000. \end{cases} \tag{4}$$

With this algorithm we want to find the input $x \in \{0, 1\}^n$ such that $f(x) = 1$. With $f : \{0, 1\}^n \rightarrow \{0, 1\}$ as an unknown function, where we implemented U_f as an oracle. $y = w$ with highest probability. $T = H^{\otimes n} U_{f_0} H^{\otimes n}$, from (4), we obtain $U_{f_0} = 2(|0\rangle\langle 0|)^{\otimes n} - I$. This result is known as reflection operator and it will be used soon [38].

Two registers used in Grover’s algorithm, n qubits in the first register and one qubit in the second register, are the critical architectural structure to achieve this speedup complexity over the classical algorithm [39]. We start the circuit for Grover’s Algorithm by creating a superposition of 2^n computational basis states in the top register (we show the general version of Grover’s algorithm Fig. 1). We initialized all the qubits in the first register to state $|0, \dots, 0\rangle$. After applying the n Hadamard gate, $H^{\otimes n}$, on the first, we have the state:

$$|\psi\rangle = H^{\otimes n} |0\rangle = \frac{1}{\sqrt{N}} \sum_{x \in \{0,1\}^n}^{N-1} |x\rangle. \tag{5}$$

Where $N = 2^n$. Note that $|\psi\rangle$ is the superposition here. If we start the second register with a single qubit in state $|0\rangle$ or $|1\rangle$, after the Hadamard we achieve the respective Hadamard basis [40]. Let $f : \{0, 1\}^N \rightarrow \{0, 1\}$ be a function defined as:

$$f(x) = \begin{cases} 1 & \text{if } x \text{ is the searched element} \\ 0 & \text{Otherwise.} \end{cases}$$

We define U_f as an oracle often referred to as a black box, defined as,

$$U_f(|i\rangle |j\rangle) = |i\rangle |j\rangle \otimes f(i). \tag{6}$$

When we apply U_f on the state $|\psi\rangle$, the state of the second register does not change (4), but the state of the first register changes, and we call this state $|\psi\rangle_1$. We assume that the Hadamard basis in the second qubit is $|-\rangle$. Here, $|\psi\rangle$ and $|\psi\rangle_1$ lives in \mathcal{H}^N . Equation. (7) defines the effect of an oracle on the achieved superposition.

$$|\psi\rangle_1 |-\rangle = U_f |\psi\rangle |-\rangle = \frac{1}{N} \sum_{i=0}^{N-1} (-1)^{f(i)} |i\rangle |-\rangle. \tag{7}$$

Due to *Quantum parallelism*, we can observe all the database elements simultaneously at the quantum level. If the position of the searched element is known, then it will be labeled as the negative value of i in equation (7). It is impossible to get this result at the classical level. Before we perform the measurement and collapse our superposition in the classical bits, we apply another set of Hadamard gates, unitary operator, and n Hadamard gates for $O(\sqrt{N})$ times. From (4), let us define this unitary operator, U_{f_0} as

$$U_{f_0} = 2 |\psi\rangle \langle \psi| - I. \tag{8}$$

When we apply this operator on state ψ_1 we have,

$$\begin{aligned} |\psi\rangle &= \left(2 |\psi\rangle \langle \psi| - I \right) |\psi\rangle_1 \\ &= \sqrt{\frac{2^n - 1}{2^n}} |w^\dagger\rangle + \sqrt{\frac{1}{2^n}} |w\rangle. \end{aligned} \tag{9}$$

Equation (9) is the state of the first register and the second register is still on state $|-\rangle$ by assumption. Notice, $|\psi\rangle$ is defined in terms of $\{w, w^\dagger\}$ states, w represents the summation of all states that can be solution to search problem; and w^\dagger is the summation of all states which are no solutions. Coefficients are factor normalization constrained by the normalization condition, $\sum_i |c_i|^2 = 1$.

Thus, we can measure this state on both the register to evaluate the function f and get the probability of finding the x record. Below, we present the implementation of this algorithm. Assuming we have an output of 1 for the function f with high probability, we calculate the probability of all possible input qubits on both the register. It is guaranteed to achieve the mentioned output. Many research are benefited from effective implementation of Grover's algorithm. We present such works in the following section.

3 Related works

This section discusses some works on Grover's algorithm and Quantum Machine Learning. In 2018, Mandviwalla et al. tested the capabilities of then-available IBM quantum computers via four qubit implementation of Grover's algorithm [41]. Their initial result showed then quantum computers could only solve small problems with a small amount of data accurately and that there is still some time before a quantum computer can surpass any classical computer. However, for the first time, in 2021, Zhang et al. presented benchmarking results for successful execution of five-qubits searching algorithm on IBM quantum processor [42]. The errors associated with the quantum devices and searching algorithms hinder the efficient implementation of these algorithms in NISQ devices. Zhang et al. proposed three strategies to improve the performance of quantum search algorithms: Hybrid classical-quantum search. Divide-and-conquer search. Quantum Search optimization via utilization of partial diffusion operator.

Currently, the efficient implementations of Grovers' algorithm on NISQ devices are limited to a few qubits. But there are impressive theoretical contributions based on the Grovers' search algorithm.

Schwabe and Westerbaan improved the complexity of solving multivariate quadratic \mathcal{MQ} over \mathbb{F}_2 from $\mathcal{O}(2^n)$ to $\mathcal{O}(2^{\frac{n}{2}})$ [43]. They evaluated the quadratic equations at a superposition implementing Grover's "oracle" for all possible inputs. They claim even ninety-two logical qubits can break \mathcal{MQ} instances. Chakraborty and Maitra achieved an exponential speedup in checking the resiliency property of

a Boolean function [44]. They analyze the Grover algorithm for quadratic improvement in query complexity. In the proposed strategy, quantum query complexity for resiliency analysis in terms of input variables requires polynomial measurements. Previously it was exponential in the worst case. The local search problem is often combined with Grover's algorithm for global optimization of a problem (system). Grover's algorithm promises a quadratic speedup for searching problems. Bulger combined the Grover algorithm with a local search technique to solve the problem that a local search technique can not solve alone [45]. The problem definition is mathematically dense, so we leave [45] as a reference for the interested reader. The amplitude amplification associated with Grover's algorithm has been applied in pattern recognition and quantum machine learning.

Tezuka et al. applied Grover search for image pattern matching [46]. They combined Amplitude Approximation Encoding (AAE), which uses a constant circuit-depth variational quantum circuit for data encoding into the quantum state [47] with the inversion-test operation that determines the projected quantum state. The projected state tries to match the targeted query set for pattern matching. The author claims the proposed framework benefits NISQ and FTQC devices. However, the author barely pays attention to the computational overhead in the AAE circuit. Grover Search had been implemented for learning in Quantum Neural Network. Du et al. reformulated the classification task as a searching problem [39]. They presented the application of Grover Search in Quantum Machine Learning (QML). We will briefly discuss QML in Sect. 3.1. The authors replaced the first oracle of Grover's algorithm with an authors-defined variational quantum circuit and multiple controlled qubits gates along Z-axis. They claim that the constructed circuit will reformulate the chosen classically hard classification task as a searching problem that can be executed with possible quantum advantage in NISQ devices [39]. Some applications of quantum machine learning can provide an advantage over classical counterparts. Support-vector machine (SVM) can reformulate and solve the classical problem for N features and M data in $\mathcal{O}(\log(\epsilon^{-1})poly(N, M))$ with ϵ accuracy [48]. However, Rebentrost et al. proved that training and classification can be done with $\mathcal{O}(\log NM)$ run time complexity using quantum SVM [49].

Grover's search algorithm has been used in cryptography, optimization, searching and sorting, machine learning classification, and many others that we can not argue to be aware of. The above-presented applications are some applications of Grover's application in different areas.

In the next section, we discuss the possible implementation of quantum computing from the Quantum machine learning perspective.

3.1 Potential applications in quantum machine learning

In this section, we briefly describe variational quantum algorithms and kernel methods. These are the potential models to implement quantum computing as a quantum classifier with application in Machine Learning.

3.1.1 Variational quantum algorithms

Variational Quantum Algorithms (VQA) address the circuit depth limit, and a limited number of qubits constrain in current (near-term) quantum devices by training the parameterized quantum circuit as a classifier. In practice, VQAs run the parameterized quantum circuit in the Quantum devices and parameter optimization on the classical optimizer. VQAs mitigate the noise of the quantum circuit because it keeps the depth of the quantum circuit shallow. VQA is considered the prime proposal to achieve the quantum advantage with near-term quantum devices [50]. Given any problem (we believe classification for our simplicity), the first step is to define the loss (or cost) function C . C encodes the solution to our problem. We then perform the quantum operation using ansatz to optimize the parameter θ . Define the optimization task as:

$$\theta^* = \arg \min_{\theta} C(\theta). \tag{10}$$

Equation (10) is trained in quantum-classical loop to obtain θ^* that is expected to approximate the real parameters. One thing to note here is that while we use the classical optimizer to train θ , the VQAs use quantum devices to estimate C . This behavior is often considered the trade-off of VQAs. Once we define the cost function and ansatz, we are ready to train the parameter θ and solve the problem defined by Eq. (10). Using the information in the C and optimization technique like gradient descent, it is proved that we can guarantee the speedup and convergence of optimizer for many optimization problems such as Eq. (10).

The most prominent implementation of VQA, sometimes also called *Quantum Neural Network (QNN)* is to tackle the classification task [51]. Here we briefly discuss the implementation of VQAs in the Grover search algorithm for classification. Du and Tao reformulated the classification task as the search algorithm using VQAs. The Grover-search-based quantum learning scheme (GBLS) dramatically reduces the number of measurements, and it outperformed the classical classifier in the measure of query complexity [39, 52]. Following the optimization problem in Eq. (10), we can define the update rule for θ as:

$$\theta^{(t+1)} = \theta^{(t)} - \frac{\eta}{B} \sum_{i=1}^B \nabla \mathcal{L}(\theta^{(t)}, B_i), \tag{11}$$

where η is the learning rate, B_i is the i -th batch for *batch gradient descent* and B is the total number of batches. We can use varied B for optimization of different quantum classifiers. One can use only Grover-based searching for the training classifier and the prediction is done using optimized Variational Quantum Circuit (VQC). Recall from Grover 1996 article [36], the algorithm finds the record, a , from the dataset of size N by iteratively applying a predefined oracle

$$U_{O_f} = I - 2 |a\rangle \langle a|, \tag{12}$$

and a diffusion operator defined on Eq. (8), and Eq. (5) as the input state. See Fig. 1 for the implementation of the circuit for the Grover algorithm. We can replace the predefined oracle U_p in Grover's Algorithm with VQC $U_{L_1} = \prod_{l=1}^L U(\theta^l)$, where L is the depth U_p , θ^l are the parameters to be optimized at layer l of U_p . More generally, the VQC consists of a data-encoding circuit $S(x)$ and parameterized circuit $W(\theta)$ applied to the computational basis state. The operation can be defined as $|\psi\rangle = W(\theta)S(x)|0\rangle^N$, where N is the number of qubits. A detailed implementation of VQC can be found on [53, 54]. the VQA with VQC is easy to implement in NISQ. However, the training and optimization of parameters can be troublesome. The next section briefly discusses the different strategy, kernel methods.

3.1.2 Kernels

The kernels method is an eminent tool in patterns analysis to identify nonlinear relationships in any given dataset [55]. The fundamental of kernel methods lies in data embedding into higher dimensional Hilbert space where they are easy to analyze. The kernel method uses kernel functions that estimate the similarity between data in higher dimensional space by calculating their inner product. We can switch between different kernel methods simply by switching between the kernel functions. In Quantum computing, this approach corresponds to changing the data encoding strategy.

Here we define a data encoding strategy. Let $\phi : \mathcal{X} \rightarrow \mathcal{F}$ be a feature map for a input space \mathcal{X} and $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{C}$ be a real or complex values positive definite functions for two data points.

Definition 1 (Modified from Def. 2 of [32]): Quantum Kernel is defined as the inner product between two data encoding feature vectors with $x, x' \in \mathcal{X}$

$$k(x, x') = |\langle \phi(x') | \phi(x) \rangle|^2 \quad (13)$$

We define $\langle \cdot | \cdot \rangle$ as the inner product of two pure quantum states.

Quantum models are often considered linear models in feature space. We can estimate Eq. (13) using quantum computers that can calculate the inner product between two pure quantum states.

Let us define a Hermitian operator \mathcal{M} acting on a vectors in Hilbert space \mathcal{H} . We can define \mathcal{M} as

$$\mathcal{M} = \sum_i \alpha_i |\mu_i\rangle \langle \mu_i|, \quad (14)$$

where α_i are the eigenvalues of \mathcal{M} [56]. $|\mu_i\rangle \langle \mu_i|$ is the outer product and $|\mu_i\rangle$ is an orthonormal basis in \mathcal{H} . \mathcal{M} is an observable or a Hamiltonian. α_i s are the measurements

Table 3 An example of input to an algorithm: input Number of qubits and clauses

Qubits	Clauses		
	3		
4	0 AND 1	1 XOR 2	2 OR 3

associated with $|\mu_i\rangle \langle \mu_i|$. Now we define quantum models as a function f of data input x :

$$f(x) = \langle \phi(x) | \mathcal{M} | \phi(x) \rangle. \tag{15}$$

Notice that Eq. 15 is in the form $\langle | \rangle$ and can be calculated as an inner product which we have defined as kernel methods. Thus, any quantum models can be considered kernel methods, and those models are a.k.a. quantum neural networks; however, based on the definition 1 those are closely related to kernel methods. Reference. [31, 32, 55, 57] investigates in-depth on quantum kernels. The scope of this paper is not to construct a quantum classifier but to relate quantum classifiers as kernel methods. The mathematical definition of VQA is closely related to the kernel methods. In both approaches, we analyzed the data in higher-dimensional Hilbert space. Previously we discussed how VQAs could reformulate classification tasks as searching problems. Constructing an oracle is an essence of implementing Grover's algorithm for quantum machine learning.

Below, we present an elementary implementation for constructing an oracle using the universal gates, AND, XOR, and OR gates.

4 Algorithm implementation

4.1 Experimental setup

Below we present a method and experiment for the proposed work. We ran the experiment using the Qiskit library, IBM open-source software for working with quantum computers (refer [58] for details). Our experimental results are divided into two parts; a) Simulation Results. b) Quantum Computer Results. While simulator experiments are executed "qasm_simulator" and utilized up to eight qubits, due to the limited available qubits in real quantum computers, we restricted ourselves to two qubits experiment.

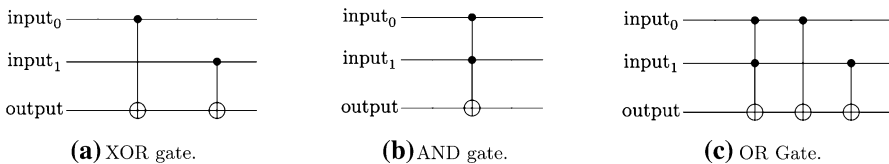


Fig. 2 Gates and their representation in a circuit

4.2 Methods

The proposed algorithm 1 requires users to initialize the number of qubits. With the number of the qubits fixed, users can define operations between these qubits as clauses in format *qubit_number*, *Universal_gates*, *qubit_number*. Based on individual clause, one of two or three qubits gates, (see Fig. 2b, a, c for gates visualization) is applied to circuit in sequence. The algorithm combines these gates to construct a circuit, U . U is applied as the first oracle to Grover's algorithm. Exploiting the amplitude amplification, the final circuit will output probabilities of possible states in $\mathcal{O}(n + c)$ time complexity in term of number of qubits n and number of clauses c . The state with the highest probabilities guarantees to yield 1 on measurement for the provided clauses. One example for possible input sequence is provided in Table 3.

The table represents the circuit initialization with four qubits and three clauses. For these three clauses the algorithm first applies AND gate Fig. 2b to the state $H^{\otimes 4} |0\rangle$, followed by XOR gate Fig. 2a, and OR gate Fig. 2c in sequential order. We aim to predict the input sequence based on these clauses, so the final measurement is 1. As mentioned before, we limit ourselves to universal gates. Below we present the construction of these gates.

We implemented the XOR gate Fig. 2a using two CNOT gates with two circuit depth. The input qubits are control qubits, and the ancilla qubit is a target qubit. In our example of 1 XOR 2, (1, 2), are the control qubits. Using two CNOT gates, we forced our circuit to result 1 on measurement if only either control qubit is 1.

The AND gate Fig. 2b is implemented using a Toffoli gate with one circuit depth. Similar to XOR gate, the input qubits are the control qubits and ancilla qubit is a target qubit. If both control qubits are in $|1\rangle$ state, the output will be one, else zero.

The OR gate Fig. 2c is implemented using a Toffoli and 2 CNOT gates with three circuit depth. For the selected gates, output is one if and only if one of the input gates has value 1.

Apart from initialized qubits, n , for each clause we add one extra qubit. For our example with three clauses, we need three extra qubits, c , here $c = 3$, resulting total of $(n + c)$ qubits circuit. These additional requirements of ancilla qubits, and up to five qubits access from IBM, limit us to implement two/three qubits inputs with two clauses in a real quantum computer. The oracle, U , is constructed under these abstractions and mathematical definition described in the introduction section. A Hadamard operation, $H^{\otimes(n)} |0\rangle$ is applied resulting state $|\Psi\rangle$. We perform $U|\Psi\rangle$ operation and apply a diffuser, U_S , before the measurement. U_S is universal. Thus, any two oracles with the same number of input qubits can use the same diffuser. An example of a complete circuit with four qubits and three universal clauses, $|q_0\rangle$ AND $|q_3\rangle$, $|q_1\rangle$ XOR $|q_2\rangle$, and $|q_2\rangle$ AND $|q_3\rangle$, is shown in Fig. 3. For a circuit of 4 qubits and three clauses, we can see that we need three additional qubits. These qubits are essential part of an algorithm. The first layer, l_1 , has Hadamard operations, followed by UU_S and measurement.

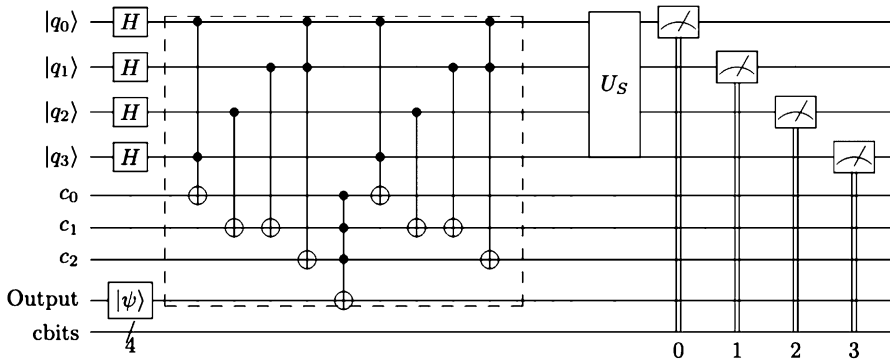


Fig. 3 Quantum circuit for Grover algorithm with user-defined oracle on clauses $|q_0\rangle$ AND $|q_3\rangle$, $|q_1\rangle$ XOR $|q_2\rangle$, and $|q_2\rangle$ AND $|q_3\rangle$. Note that we need seven qubits in total. For c clauses, three in this case, we require c additional qubits and one output qubit, resulting $(n + c + 1)$ qubits quantum circuit. The circuit depth for an oracle is $(2 \sum_{i=1}^c G_i) + 1$, where G_i is individual gate depth. Refer to figures in Fig. 2 for actual circuit depth for each gate

Algorithm 1 To construct a defined oracle.

Input: q_1, q_2, \dots, q_n and c_1, \dots, c_k

Parameter: Set of universal clauses.

Output: Input bits that output 1.

- 1: **while** input q_i **do**
 - 2: $|\psi_1\rangle \leftarrow \text{Hadamard}(q_i)$
 - 3: **end while**
 - 4: **while** clause c_j **do**
 - 5: ORACLE:
 - 6: $O_i \leftarrow c_i(q_j, q_k)$
 - 7: out $\leftarrow \text{Toffoli}(O_1, O_2, \dots, O_n)$
 - 8: **end while**
 - 9: Grover(ORACLE)
-

Based on our algorithm and methods defined we compare and argue on the results from simulation and quantum computers in the next sections.

5 Results and discussion

As mentioned previously and shown in Fig. 3, we need c extra qubits to process c clauses, where c is the number of clauses. Thus for experimental purposes on real quantum computers, we are limited to implementing two qubits circuit with two clauses as an input under five qubits constraint from IBM. Under these settings, we executed our experiments. We experimented on “ibmq_bogota” and “ibmq_santiago” backend for quantum computers and on “aer_simulator” for simulation. On both quantum computer backend, our submitted job had a queue of approximately four hundred. So, each experiment took about half an hour to execute.

We acknowledge that some institutions might provide better quantum computers and achieve better results. But for this research purpose, we limit ourselves to the IBM quantum laboratory.

Consider these set of clauses: 0 AND 1 1 XOR 2 in a 3 qubits circuit. Upon the input of 110 the circuit yields 1 analytically. Given such clauses, our oracle, U , is guaranteed to find these input sequence with a high probability, 0.776 in this case, (Fig. 4). To clarify, consider the first clause 0 AND 1. One can immediately conclude that it yields 1 if and only if both the inputs are 1. Combining this result with the second clause, 1 XOR 2, it is only possible to obtain 1 as an output if and only if the third qubit is 0. So, without looking at any result, one can confirm that the possible input to obtain 1 as output for given clauses is 110. Our algorithm obtains the same result. We provide a histogram Fig. 5 for four qubits circuits inputs with three clauses; 0 AND 3 1 XOR 2 2 AND 3. We get 1011 input sequence with a probability of 0.453. One can verify that these inputs hold following previously explained clarification.

In two of the provided examples, we observe that the probability of the right input sequences decreases as we increase the number of qubits. However, this hypothesis does not always hold. Furthermore, as we increase the number of qubits, we will have different options for right input sequence. Tables 4, 5 provide the result for five qubits with the clauses: 0 AND 1 1 XOR 4 2 OR 4 3 XOR 0 and eight qubits circuit with clauses: 0 AND 1 1 OR 2 2 XOR 3 3 XOR 4 4 OR 6 5 AND 7. One can look in the table and find various possible answers. We only highlight the one with the highest probability. The only constraint it respects is the sum of probabilities has to be 1. We do not have any limitations on the number of clauses. The only constrain on our algorithm is that the control qubit precedes universal gate followed by target qubit. The simulations results looks satisfying. However, the result is very different when we run such experiments in real quantum computers. Below we compare three results obtained on simulations and quantum computers.

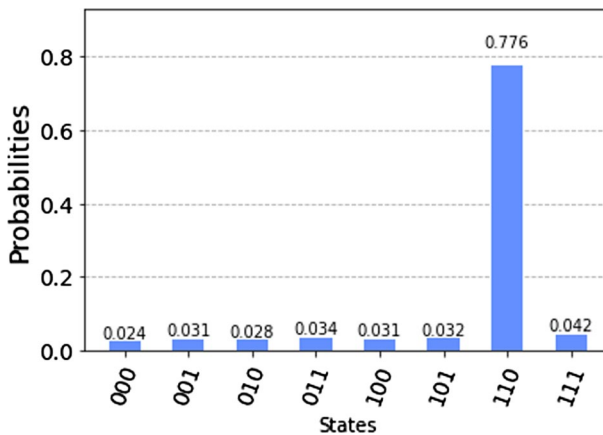


Fig. 4 Histogram for clauses 0 AND 1, 1 XOR 2 on 3 qubits circuit

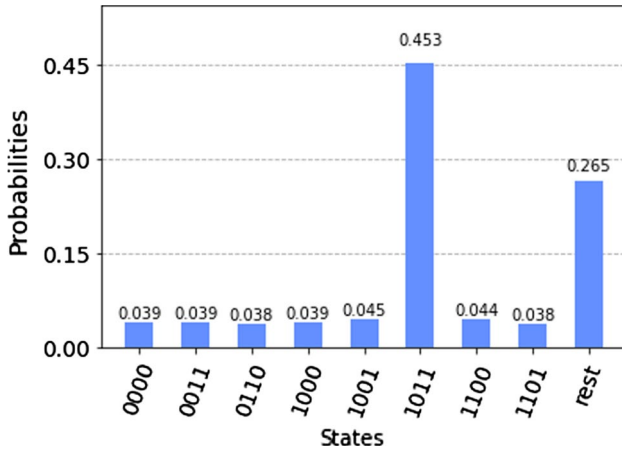


Fig. 5 Histogram for clauses 0 AND 3,1 XOR 2, 2 AND 3 on 4 qubits circuit. *rest* is the probabilities of the states that are not shown in histogram. $states \in \{0, 1\}^4$

1. An experiment for a clause 0 AND 1 and 1 XOR 0. Figure 6 illustrates a histogram with probabilities for a clause 1 on simulation and quantum computer. We can visualize that histograms (Fig. 6a, b) gives different results under the same operations. This is because the simulation has a higher degree of fault-tolerant and is prone to provide better results than the actual quantum computer. Error mitigation in quantum devices is an active area of research. Some of the work can be found on [59–61].
2. An experiment for a clause 0 XOR 1 and 1 AND 0. Figure 7 illustrates the histogram of state probabilities for the above clause. We see while simulation provides the resulting state as $|11\rangle$ with the highest probability of 0.267 followed by state $|01\rangle$ with a probability of 0.26, real quantum provides different results, state $|10\rangle$ with a probability of 0.281 followed by a state $|01\rangle$ with a probability of 0.256. If we follow the descriptions from Sect. 4.2 and replace Universal gates in clauses 2 by the quantum gates provided in Fig. 2, we can analytically verify

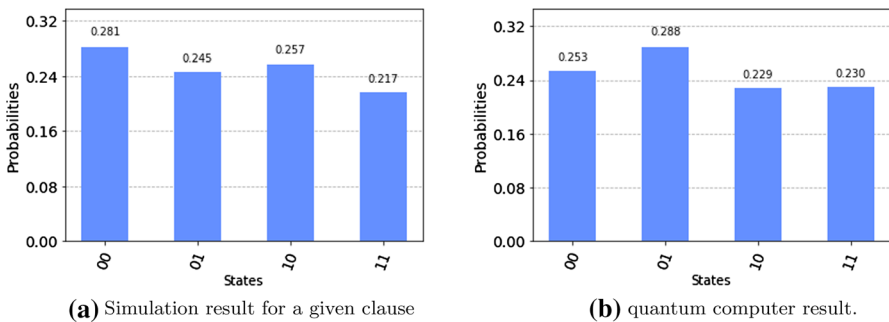


Fig. 6 Simulation and quantum computer results for 0 AND 1 and 1 XOR 0 clauses

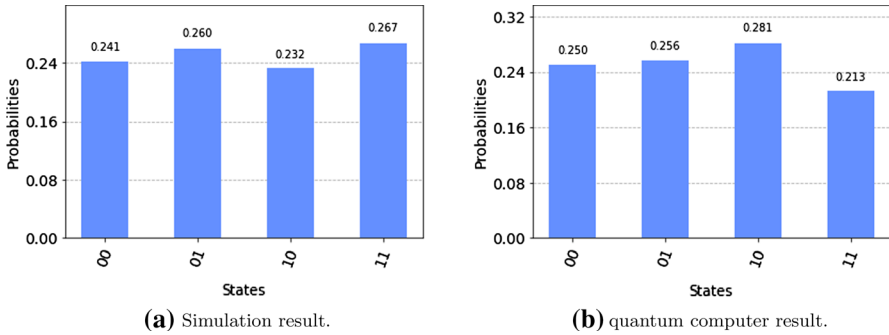


Fig. 7 Simulation and quantum computer results for 0 XOR 1 and 1 AND 0 clauses

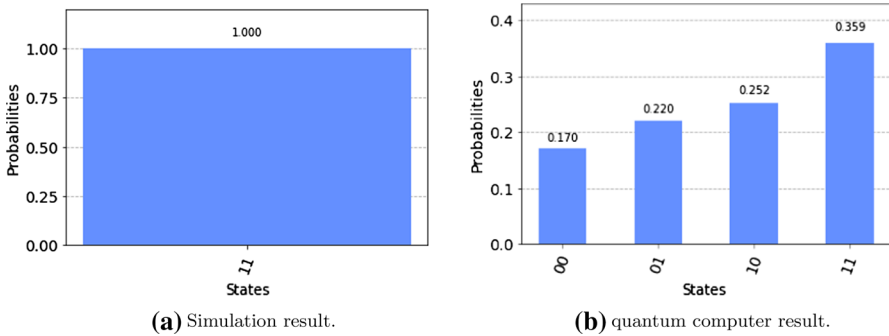


Fig. 8 Simulation and quantum computer results 0 AND 1 clause

Table 4 Probabilities values for five qubits

String	Probability	String	Probability	String	Probability	String	Probability
11111	0.02051	11010	0.01465	00000	0.02637	10000	0.02441
01100	0.02148	10011	0.02441	10110	0.02441	11001	0.01953
01011	0.02832	01110	0.02637	00010	0.03516	11100	0.25879
11011	0.01465	01000	0.03125	01111	0.02930	01010	0.03418
01001	0.03027	00101	0.01855	10111	0.01660	00100	0.02344
11000	0.02734	10101	0.02051	01101	0.02734	00011	0.02441
10001	0.02441	00111	0.01563	00110	0.01953	11110	0.02051
10100	0.03223	11101	0.02051	10010	0.02539		

Table 4 presents the probabilities values for five qubits circuit with four clauses: a. 0 AND 1 b. 1 XOR 4 c. 2 OR 4 d. 3 XOR 0

Bold values indicate largest probability

that state $|11\rangle$ satisfy the clause. In practice, the results from quantum computers are probabilistic rather than deterministic. This means the same experiment is conducted multiple times, and the result is sampled from the generated probabil-

Table 5 Probabilities values for eight qubits

String	Probability	String	Probability	String	Probability	String	Probability
00110111	0.001953125	00100110	0.003906250	00100100	0.002929688	11011110	0.004882813
00001101	0.000976563	00011101	0.002929688	00110000	0.002929688	01010010	0.006835938
01010000	0.000976563	00111000	0.004882813	00011010	0.000976563	11011000	0.001953125
11101110	0.001953125	01111100	0.004882813	00010101	0.004882813	10011011	0.007812500
00100001	0.002929688	11011101	0.005859375	11100101	0.003906250	01000100	0.001953125
01010100	0.003906250	00011000	0.001953125	00000110	0.004882813	10011111	0.003906250
10110011	0.001953125	01000011	0.003906250	11001010	0.001953125	01101010	0.001953125
01000110	0.001953125	01000001	0.002929688	11110111	0.002929688	11011001	0.005859375
10111101	0.001953125	00000000	0.002929688	00101010	0.003906250	11110000	0.002929688
01001011	0.001953125	10010001	0.003906250	10110000	0.005859375	10001111	0.007812500
00000001	0.005859375	10000100	0.007812500	01111000	0.001953125	00001100	0.004882813
10011100	0.001953125	10011000	0.004882813	11011010	0.008789063	10111110	0.003906250
10001100	0.002929688	10001101	0.003906250	11100111	0.031250000	11000011	0.003906250
10010000	0.002929688	01111011	0.003906250	11111000	0.000976563	01110000	0.002929688
10001001	0.002929688	00010001	0.005859375	01100100	0.005859375	00011001	0.002929688
00110011	0.000976563	00101001	0.005859375	01011000	0.002929688	01010011	0.001953125
00011110	0.000976563	01001101	0.002929688	10101110	0.004882813	11110010	0.002929688
01100111	0.000976563	11101011	0.001953125	11110001	0.009765625	01000010	0.003906250
10100111	0.003906250	10001011	0.004882813	01110010	0.002929688	10001110	0.003906250
10111100	0.001953125	11100001	0.005859375	01011100	0.004882813	10110010	0.003906250
11000001	0.004882813	11101001	0.002929688	11010010	0.001953125	01100010	0.004882813
01110100	0.000976563	01101000	0.003906250	01010111	0.004882813	00000011	0.006835938
10100100	0.003906250	10101111	0.003906250	11000110	0.003906250	10100011	0.003906250
11010110	0.002929688	11101000	0.002929688	00010110	0.005859375	10011010	0.004882813
11111001	0.004882813	11001110	0.001953125	10110101	0.002929688	01011011	0.005859375
00101100	0.000976563	01010101	0.003906250	11011100	0.002929688	1101101	0.043945313
00101000	0.003906250	00000010	0.005859375	01000000	0.002929688	10000110	0.001953125
01001110	0.002929688	11011111	0.003906250	11010001	0.003906250	01010001	0.004882813
11000111	0.002929688	01011010	0.002929688	00010100	0.000976563	01011101	0.003906250
00001111	0.002929688	11111111	0.001953125	01001010	0.006835938	11111011	0.005859375
10100101	0.002929688	10010110	0.003906250	01111010	0.005859375	01100001	0.003906250
10010111	0.001953125	10010101	0.005859375	00001110	0.002929688	10000111	0.001953125
11100110	0.002929688	11101111	0.038085938	10100001	0.004882813	11010111	0.002929688
01001111	0.001953125	11010100	0.000976563	10111010	0.002929688	01101111	0.003906250
01110101	0.003906250	00011111	0.005859375	11100010	0.003906250	10100110	0.003906250
11110100	0.000976563	11001000	0.003906250	00110101	0.004882813	11010011	0.003906250
10111011	0.003906250	00101111	0.003906250	10001000	0.001953125	10101000	0.003906250
11101100	0.001953125	00010010	0.005859375	01010110	0.002929688	11000000	0.005859375
11011011	0.002929688	00111101	0.002929688	00101101	0.003906250	11100011	0.001953125
01100101	0.002929688	11111010	0.002929688	00001000	0.002929688	01101110	0.004882813
10110111	0.002929688	11110011	0.001953125	00111011	0.001953125	11110101	0.006835938
01100110	0.001953125	01100011	0.004882813	10000010	0.003906250	00000100	0.000976563
00111111	0.002929688	00100010	0.004882813	01111101	0.002929688	10110110	0.002929688
10101010	0.001953125	00011011	0.004882813	11100100	0.003906250	10101100	0.001953125
00111110	0.002929688	00100000	0.002929688	10011101	0.001953125	01011111	0.002929688

Table 5 (continued)

String	Probability	String	Probability	String	Probability	String	Probability
10011110	0.003906250	01110001	0.005859375	00000111	0.004882813	10000101	0.002929688
00111001	0.001953125	10101101	0.002929688	11000010	0.003906250	11101010	0.005859375
10101011	0.003906250	01111111	0.002929688	00100101	0.000976563	00111100	0.003906250
10000001	0.002929688	10000011	0.001953125	11001100	0.002929688	00010011	0.000976563
01100000	0.004882813	10110001	0.001953125	10100000	0.001953125	10111001	0.002929688
00100011	0.003906250	11001101	0.000976563	01000101	0.005859375	10101001	0.002929688
00001011	0.002929688	00100111	0.001953125	10010011	0.005859375	01110011	0.002929688
01111110	0.000976563	10010100	0.005859375	01110111	0.003906250	10011001	0.003906250
11000100	0.004882813	10000000	0.003906250	10111111	0.003906250	00010000	0.002929688
00001010	0.002929688	01011001	0.006835938	00110001	0.004882813	00101011	0.002929688
01101100	0.001953125	00111010	0.001953125	11100000	0.001953125	11111101	0.001953125
01101011	0.004882813	11111110	0.001953125	11110110	0.003906250	00110010	0.001953125
00101110	0.001953125	11010101	0.004882813	01101101	0.001953125	11001111	0.005859375
01101001	0.003906250	01001100	0.003906250	00001001	0.002929688	10010010	0.005859375
01001001	0.006835938	00000101	0.004882813	00110100	0.002929688	00011100	0.001953125
10110100	0.006835938	01110110	0.003906250	11000101	0.001953125	11010000	0.002929688
01001000	0.002929688	11001001	0.004882813	00110110	0.003906250		
00100110	0.003906250	00011101	0.002929688	00010111	0.004882813		

Table 5 presents the probabilities values for eight qubits circuit with six clauses: a. 0 AND 1 b. 1 OR 2 c. 2 XOR 3 d. 3 XOR 4 e. 4 OR 6 f. 5 AND 7

Bold values indicate largest probability

ity distribution. Measurements in quantum devices are beyond the scope of this paper. Interested readers can refer to [56, 62].

- An experiment for a clause 0 AND 1. Figure 8 illustrates the surprising result. Analytically, one can immediately verify that for a clause 0 AND 1, $|11\rangle$ is the valid input state. While Fig. 8a yields the targeted state with 1 probability, surprisingly, Fig. 8b shows that the probability of obtaining the targeted state $|11\rangle$ is approximately to 36% when the experiment is executed on actual quantum computers. This probability is lower than flipping a coin. Due to probabilistic approach for quantum measurement, the circuit was executed for 5000 shots. Unfortunately, current quantum computers are noisy/error-prone [63, 64] and significantly affects the results. There are numerous works for error-corrections and building fault-tolerant quantum devices. Decoherence Noise, Control Noise, Pulse Shape noise, Cross talk noise, and Leakage noise are few noises classes that accompany errors in quantum devices. Although quantum computers' results tries to mimic the simulation results, these results vary due to various errors and noises.

The difference in simulation and real quantum computers results brings the issue of noise and errors in current quantum devices. We briefly discussed that due to these various errors, listed in 3, the measurements/outcomes of quantum computers are probabilistic. The experiment provided in this paper are trivial and targeted at

the audience with no/few prior quantum computing experience. Our algorithm can help the users to simulate any classical logical circuits built using universal gates into a quantum system. Provided with the parsed operations, (e.g., A AND B), our algorithms can provide the valid input results in $\mathcal{O}(\frac{n+c}{2})$ complexity in number of qubits n and clauses c . The results showed that Grover's algorithm with a universal gates-defined oracle could obtain the input states that satisfy the circuit. The Grover's Algorithm has shown promising results in secure secret-sharing in two qubits case [65], extracting key Advanced Encryption Standard (AES) plaintext-ciphertext pairs [66], constructing a gate for conditional phase-shift [67], cryptography [68] and many searching problems. There are quantum algorithms that solve many of the classical algorithms efficiently, but we lack the quantum computers to execute these algorithms. It is well known even a few qubits circuit is not fault-tolerant. We have shown it in the sections of our results too. We believe quantum computing exploits machine learning and cryptography, and there are different works to prove it. Refer [31–33, 68–72] for various works on quantum cryptography and quantum machine learning. Although the current limitations of quantum computers and qubits do not serve any real-world purpose, with the current advancement, we believe it will surpass classical computers in solving many classically hard problems efficiently in the future.

6 Conclusion

This paper discusses the basics of quantum computing, including Grover's algorithm, and quantum machine learning. We focused on Grover's algorithm and gave a presented an experiment to simulate classical logical circulate exploiting amplitude amplification. We consider that quantum computing and quantum machine learning can provide an advantage over some classically challenging problems by leveraging quantum computers' efficiency, reducing the need for supercomputing power to execute such programs.

Furthermore, we describe two approaches for quantum machine learning, namely, variational quantum circuits and kernel-based quantum machine learning, as potential applications in supercomputing and other technologies. Quantum algorithms and their applicability in machine learning problems is an area that necessitates further research to identify equivalent solutions and improvements over classic machine learning computing. We encourage the reader to keep track of research in these areas as we believe there are significant opportunities for kernel-based approaches to succeed in the short-term future of quantum machine learning and supercomputing.

Our future work plans include testing our algorithm with actual data of the classical circuit. We will also parse the circuit and supply it to our algorithm, c.f. Sect. 5. We will work toward improving computational complexity with respect to the classical approach. We believe the classification problems in machine learning can be formulated as a searching problem that can be executed using Grover's Algorithm. Finally, we will attempt to reformulate this problem as a kernel method that can be efficiently solved as a searching problem. We believe this future work can successfully exploit Grover's algorithm advantages.

Acknowledgements We would like to thank the Department of Computer Science and Baylor.AI laboratory at Baylor University for their support. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM or the IBM Quantum team. J. O. executed most of this work while at Baylor University.

Funding This work was supported in part by the National Science Foundation under grants CNS-2136961, and CNS-2210091.

Data availability Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of interest All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. David McMahon (2007) Quantum computing explained. Wiley, New Jersey
2. Hill DP, Harper A, Malcolm J, McAndrews MS, Mockus SM, Patterson SE, Reynolds T, Baker EJ, Bult CJ, Chesler EJ et al (2019) Cisplatin-resistant triple-negative breast cancer subtypes: multiple mechanisms of resistance. *BMC Cancer* 19(1):1–13
3. Bubier J, Hill D, Mukherjee G, Reynolds T, Baker EJ, Berger A, Emerson J, Blake JA, Chesler EJ (2019) Curating gene sets: challenges and opportunities for integrative analysis. *Database* 2019
4. Benton ML, Abraham A, LaBella AL, Abbot P, Rokas A, Capra JA (2021) The influence of evolutionary history on human health and disease. *Nat Rev Genet* 22(5):269–283
5. Islam SA, Sajed T, Kearney CM, Baker EJ (2015) Predstp: a highly accurate svm based model to predict sequential cystine stabilized peptides. *BMC Bioinform* 16(1):1–11
6. Gidney C, Ekerå M (2021) How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits. *Quantum* 5:433. <https://doi.org/10.22331/q-2021-04-15-433>
7. Sadana S, Maccione L, Sinha U (2021) Quantum computers to test fundamental physics or viceversa
8. Arrazola JM, Jahangiri S, Delgado A, Ceroni J, Izaac J, Száva A, Azad U, Lang RA, Niu Z, Matteo OD, Moyard R, Soni J, Schuld M, Vargas-Hernandez RA, Tamayo-Mendoza T, Aspuru-Guzik A, Killoran N (2021) Differentiable quantum computational chemistry with PennyLane
9. Emami PS, Warrell J, Anticevic A, Bekiranov S, Gandal M, McConnell MJ, Sapiro G, Aspuru-Guzik A, Baker JT, Bastiani M et al (2021) Quantum computing at the frontiers of biological sciences. *Nat Methods* 4:1–9
10. Khanal B, Rivas P, Orduz J (2021) Human activity classification using basic machine learning models. In: 2021 international conference on computational science and computational intelligence (CSCI) . Accepted, to be published soon
11. Rivas P, Zhao L, Orduz J (2021) Hybrid quantum variational autoencoders for representation learning. In: 2021 international conference on computational science and computational intelligence (CSCI). Accepted, to be published soon
12. Freedman MH, Kitaev A, Wang Z (2002) Simulation of topological field theories by quantum computers. *Commun Math Phys* 227(3):587–603
13. Raussendorf R, Browne DE, Briegel HJ (2003) Measurement-based quantum computation on cluster states. *Phys Rev A* 68:022312. <https://doi.org/10.1103/PhysRevA.68.022312>
14. Benioff P (1980) The computer as a physical system: a microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *J Stat Phys* 22(5):563–591
15. Kitaev AY (1997) Quantum computations: algorithms and error correction. *Uspekhi Matematicheskikh Nauk* 52(6):53–112
16. Penrose R (1971) Applications of negative dimensional tensors. *Combinatorial mathematics and its applications* 1, 221–244 . See PDF: <https://www.mscs.dal.ca/~selinger/papers/graphical-bib/public/Penrose-applications-of-negative-dimensional-tensors.pdf>

17. Farhi E, Goldstone J, Gutmann S, Sipser M (2000) Quantum computation by adiabatic evolution. [arXiv: Quantum Physics](#)
18. Childs AM, Farhi E, Preskill J (2001) Robustness of adiabatic quantum computation. *Phys Rev A* 65(1):012322
19. IBM: Qiskit. <https://qiskit.org/>
20. Anonymous: Grover implementation. Anonymous repository. <https://anonymous.4open.science/r/NoRemoving-8DFF/groverAlgo.ipynb>
21. IBM: Grover's algorithm. website. <https://qiskit.org/textbook/ch-algorithms/grover.html> (2021)
22. Leifer MS, Poulin D (2008) Quantum graphical models and belief propagation. *Ann Phys* 323(8):1899–1946
23. Deutsch D (1985) Quantum theory, the church-turing principle and the universal quantum computer. *Proc R Soc Lond A Math Phys Sci* 400(1818):97–117
24. Mermin ND (2003) From cbits to qbits: teaching computer scientists quantum mechanics. *Am J Phys* 71(1):23–30. <https://doi.org/10.1119/1.1522741>
25. Wittek P (2014) Quantum machine learning: what quantum computing means to data mining. Academic Press, Massachusetts
26. Montanaro A (2016) Quantum algorithms: an overview. *npj Quantum Inf* 2(1):1–8
27. Shor PW (1999) Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev* 41(2):303–332
28. Lator C, Manssur L, Portugal R (2003) Grover's algorithm: Quantum database search. arXiv preprint [quant-ph/0301079](https://arxiv.org/abs/quant-ph/0301079)
29. Vartiainen JJ, Niskanen AO, Nakahara M, Salomaa MM (2004) Implementing shor's algorithm on josephson charge qubits. *Phys Rev A* 70(1):012319
30. Harrow AW, Hassidim A, Lloyd S (2009) Quantum algorithm for linear systems of equations. *Phys Rev Lett* 103(15):150502
31. Schuld M, Killoran N (2019) Quantum machine learning in feature hilbert spaces. *Phys Rev Lett* 122(4):040504
32. Schuld M (2021) Supervised quantum machine learning models are kernel methods. arXiv preprint [arXiv:2101.11020](https://arxiv.org/abs/2101.11020)
33. Biamonte J, Wittek P, Pancotti N, Rebentrost P, Wiebe N, Lloyd S (2017) Quantum machine learning. *Nature* 549(7671):195–202
34. Lloyd S, Mohseni M, Rebentrost P (2013) Quantum algorithms for supervised and unsupervised machine learning. arXiv preprint [arXiv:1307.0411](https://arxiv.org/abs/1307.0411)
35. Chen G, Fulling SA, Lee H, Scully MO (2001) Grover's algorithm for multiobject search in quantum computing. *Directions in quantum optics*. Springer, Berlin, pp 165–175
36. Grover LK (1997) Quantum mechanics helps in searching for a needle in a haystack. *Phys Rev Lett* 79:325–328. <https://doi.org/10.1103/PhysRevLett.79.325>
37. Grover LK (1996) A fast quantum mechanical algorithm for database search. In: *Proceedings of the twenty-eighth annual acm symposium on theory of computing*, pp 212–219
38. Rungta P (2009) The quadratic speedup in grover's search algorithm from the entanglement perspective. *Phys Lett A* 373(31):2652–2659
39. Du Y, Hsieh M-H, Liu T, Tao D (2021) A grover-search based quantum learning scheme for classification. *New J Phys* 23(2):023020. <https://doi.org/10.1088/1367-2630/abdefa>
40. Nielsen MA, Chuang I (2002) Quantum computation and quantum information. American Association of Physics Teachers
41. Mandviwalla A, Ohshiro K, Ji B (2018) Implementing grover's algorithm on the ibm quantum computers. In: *2018 IEEE International Conference on Big Data (Big Data)*, pp 2531–2537. IEEE
42. Zhang K, Rao P, Yu K, Lim H, Korepin V (2021) Implementation of efficient quantum search algorithms on nisq computers. *Quantum Inf Process* 20(7):1–27
43. Schwabe P, Westerbaan B (2016) Solving binary MQ with grover's algorithm. In: *International conference on security, privacy, and applied cryptography engineering*. Springer, pp 303–322
44. Chakraborty K, Maitra S (2016) Application of grover's algorithm to check non-resiliency of a boolean function. *Cryptogr Commun* 8(3):401–413
45. Bulger DW (2007) Combining a local search and grover's algorithm in black-box global optimization. *J Optim Theory Appl* 133(3):289–301
46. Tezuka H, Nakaji K, Satoh T, Yamamoto N (2022) Grover search revisited: application to image pattern matching. *Phys Rev A* 105(3):032440

47. Nakaji K, Uno S, Suzuki Y, Raymond R, Onodera T, Tanaka T, Tezuka H, Mitsuda N, Yamamoto N (2022) Approximate amplitude encoding in shallow parameterized quantum circuits and its application to financial market indicators. *Phys Rev Res* 4(2):023136
48. Tsang IW, Kwok JT, Cheung P-M, Cristianini N (2005) Core vector machines: fast svm training on very large data sets. *J Mach Learn Res* 6(4)
49. Reberntrost P, Mohseni M, Lloyd S (2014) Quantum support vector machine for big data classification. *Phys Rev Lett* 113(13):130503
50. Cerezo M, Arrasmith A, Babbush R, Benjamin SC, Endo S, Fujii K, McClean JR, Mitarai K, Yuan X, Cincio L et al (2021) Variational quantum algorithms. *Nat Rev Phys* 1–20
51. Goodfellow I, Bengio Y, Courville A (2016) *Deep learning*. MIT Press, Massachusetts
52. Khanal B, Rivas P, Orduz J, Zhakubayev A (2021) Quantum machine learning: a case study of grover's algorithm. In: The 19th International Conference on Scientific Computing (CSC 2021)
53. Team TQ (2022) Simulating molecules using VQE. Data 100 at UC Berkeley . <https://qiskit.org/textbook/ch-applications/vqe-molecules.html>
54. Team TP Variational classifier. https://pennylane.ai/qml/demos/tutorial_variational_classifier.html
55. Park DK, Blank C, Petruccione F (2020) The theory of the quantum kernel-based binary classifier. *Phys Lett A* 384(21):126422
56. Schuld M, Petruccione F (2021) *Machine learning with quantum computers*. Springer, Berlin
57. Scholkopf B, Smola AJ (2001) *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, Cambridge, MA, USA
58. IBM Q (2021) IBM Quantum. Website: <https://www.ibm.com/quantum-computing/>
59. Endo S, Benjamin SC, Li Y (2018) Practical quantum error mitigation for near-future applications. *Phys Rev X* 8(3)
60. Endo S, Cai Z, Benjamin SC, Yuan X (2021) Hybrid quantum-classical algorithms and quantum error mitigation. *J Phys Soc Jpn* 90(3):032001
61. Strikis A, Qin D, Chen Y, Benjamin SC, Li Y (2021) Learning-based quantum error mitigation. *PRX. Quantum* 2(4):040330
62. Elben A, Vermaersch B, van Bijnen R, Kokail C, Brydges T, Maier C, Joshi MK, Blatt R, Roos CF, Zoller P (2020) Cross-platform verification of intermediate scale quantum devices. *Phys Rev Lett* 124(1):010504
63. Lidar DA (2008) Towards fault tolerant adiabatic quantum computation. *Phys Rev Lett* 100(16):160506
64. Shor PW (1995) Scheme for reducing decoherence in quantum computer memory. *Phys Rev A* 52(4):2493
65. Hsu L-Y (2003) Quantum secret-sharing protocol based on grover's algorithm. *Phys Rev A* 68(2):022306
66. Grassl M, Langenberg B, Roetteler M, Steinwandt R (2016) Applying grover's algorithm to aes: quantum resource estimates. *Post-quantum cryptography*. Springer, Berlin, pp 29–43
67. Fujiwara S, Hasegawa S (2005) General method for realizing the conditional phase-shift gate and a simulation of grover's algorithm in an ion-trap system. *Phys Rev A* 71(1):012337
68. Aumasson J-P (2017) The impact of quantum computing on cryptography. *Comput Fraud Secur* 2017(6):8–11
69. Mavroeidis V, Vishi K, Zych MD, Jøssang A (2018) The impact of quantum computing on present cryptography. arXiv preprint [arXiv:1804.00200](https://arxiv.org/abs/1804.00200)
70. Brassard G (1994) Quantum computing: the end of classical cryptography? *ACM SIGACT News* 25(4):15–21
71. Brassard G, Lütkenhaus N, Mor T, Sanders BC (2000) Limitations on practical quantum cryptography. *Phys Rev Lett* 85(6):1330
72. Adcock J, Allen E, Day M, Frick S, Hinchliff J, Johnson M, Morley-Short S, Pallister S, Price A, Stanic S (2015) Advances in quantum machine learning. arXiv preprint [arXiv:1512.02900](https://arxiv.org/abs/1512.02900)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.