QEML (Quantum Enhanced Machine Learning)
Using Quantum Computing to Enhance ML Classifiers and Feature Spaces

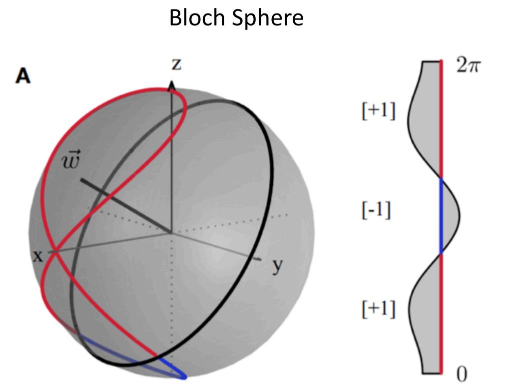
Quantum Computing

- New paradigm of computing, initially proposed by Richard Feynman
- Quantum systems cannot be efficiently simulated by current classical computers
- Useful for understanding systems of electrons and the probabilities of their location
- Ex. - If we have two electrons constrained at two points (A and B), then there are 4 possible probabilities of their location (both at A, one A – one B, one B – one A, both at B, etc.). For 3 electrons, there are 8 probabilities, for 10 electrons, there are 1,024 probabilities, and at 20 electrons, there are 1,048,576 probabilities.
- Thus, it is easy to see that measurements get out of hand quickly!
- Quantum Computing rapidly solves problems which even the most advanced parallel systems cannot model within a reasonable time

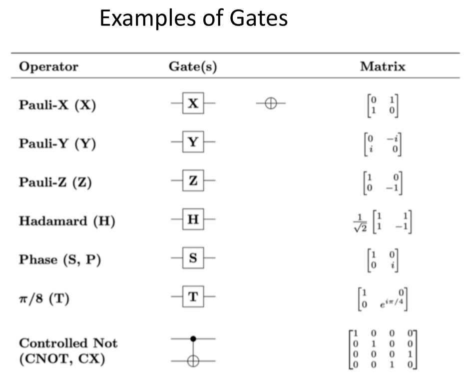
Qubits and Quantum Gates

- Qubits: fundamental building blocks of quantum circuits (a vector in the 2-D Hilbert space)
- Has the ability to be in a state of **superposition** (both |0⟩ and |1⟩)
- Superposition can be modeled as: $|\phi\rangle = a|0\rangle + b|1\rangle$
- Each qubit can be modeled as a Bloch Sphere (shown below)
- In a quantum computer, qubits lie in a "quantum register"
- We perform quantum operations and apply superposition within this register (ex. 2 qubit quantum register is the tensor product of the state of 2 qubits)
- Quantum gates are the quantum analogs of logic gates and apply combinations of qubits to model logical behavior (reversible gates)
- Mathematically represented by unitary matrices: $UU^{\dagger} = I$
- Transformation of one quantum state into another: $U|\psi_1\rangle = |\psi_2\rangle$

Bloch Sphere



Examples of Gates

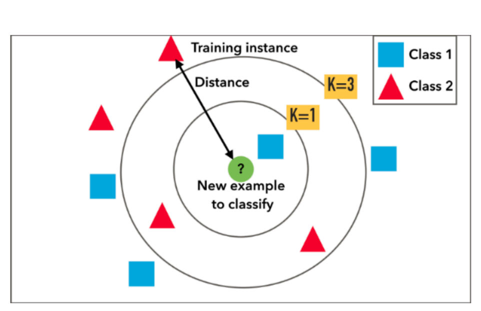


Machine Learning and Kernel Methods

- Problem: How do we improve current machine learning kernel methods?
- Classifiers such as the Support Vector Machine (SVM) suffer from issues in performance due to a **limited feature space**
- Kernel functions are also still computationally expensive to estimate and on higher dimensional datasets, most machine learning algorithms are quite expensive to run
- We can solve this problem using quantum computing since properties like superposition and entanglement allow us to take an exponentially large feature space and reduce it to a linear size
- Since the quantum variant of the SVM (QSVM) has already been implemented in the public domain, this research attempts to model the quantum variant of the K-nearest neighbors classifier

K-Nearest Neighbors Classifier

- Simple example of supervised learning classification (map input to a discrete output via a function)
- **Assumption**: datapoints with similar behavior exist near each other
- Thus, KNN captures the concept of **proximity** or closeness
- Algorithm computes distance to the nearest data points and assigns the most frequent value to the unlabeled example
- Key Idea: Based on a distance metric, typically Euclidean Distance $\sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$
- Two parameters which affect performance of KNN the most: value of *K* and number of dimensions (n)
- Typically *K* is assigned the value of $\sqrt{N}$ where N is the number of training data points



Visualization of K-nearest neighbors classifier

Issue with KNN: "Curse of Dimensionality"

- As earlier mentioned KNN is quite computationally expensive, especially for large values of K and n
- If we want an algorithm with 1,000 data points to find 5 neighbors, in 1-D space, the average distance is $5/1000$ or 0.005
- In 2-D space, the average distance is $(0.005)^{1/2}$
- In n-dimensional space, the average distance is $(0.005)^{1/n}$
- We can thus leverage quantum computing to promote the concept of an **enhanced feature space** where the distance can be computed in a more efficient and scalable manner which also works in higher dimensions (minimize CPU time and cost of storage)
- We may need to use a quantum metric like Hamming Distance

Solution and Contribution: Quantum-Enhanced Machine Learning (QEML)

Goal: Use quantum computing to enhance previously existing supervised learning classifiers like the KNN algorithm and implement a QKNN in the IBM Q Experience platform

- Implement idea of an enhanced feature space to reduce the cost of data storage
- Use quantum techniques like Hamming distance and Grover's algorithm to calculate the distance between points in a QKNN
- QKNN has not yet been publicly implemented (QSVM has)

QEML Property 2: Acceleration of quantum algorithms

- The speed and execution time can be much more efficient for QEML algorithms since they employ the property of **quantum parallelism** (arises from superposition)
- Ex. For a classical computer to manipulate 2^n states, it would require exponential time and memory, but for a quantum computer it would take linear time and space due to quantum parallelism

QEML Property 1: Enhanced Feature Space:

- The storage space can be exponentially reduced through the property of **quantum superposition** since qubits can hold an extra simultaneous state (both |0⟩ and |1⟩)
- Ex. In a quantum computer, all binary numbers in the set $\{0, 1, \ldots, 2^{n-1}\}$ exist in an N qubit register, whereas in a classical computer, only one binary number in the set $\{0, 1, \ldots, 2^{n-1}\}$ can exist in an N bit register

Experiment: Perform 3 separate experiments to simulate QEML on the IBM Q platform using the Qiskit python library

- Experiment 1: Simulate the concept of fidelity (cosine similarity in classic ML) in a basic circuit and demonstrate Grover's algorithm (a quantum minimum search algorithm)
- Experiment 2 (Main Contribution): Construct a QKNN and compare it to a KNN to evaluate success

Initial Experiments: Simulating Fidelity and Grover's Algorithm

Experimental Overview

Simulate Fidelity → Simulate Grover's Algorithm → Design and compare QKNN

Simulating Grover's Algorithm

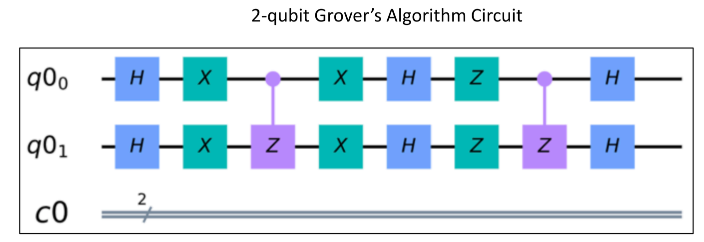
1. What is Grover's Algorithm?

- An example which demonstrates the physical speed-up that quantum computers provide
- A quantum minimum path searching algorithm
- The application to QKNN is that when we calculate distances in a quantum-enhanced feature spaces, we find the shortest distance (nearest point) using such an algorithm
- If we use a classic computer to search through a dataset of dimension n, the time it takes the locate a certain point is $O(N)$
- However, with a quantum computer, the time it takes is only $O(\sqrt{N})$

2. Simulation Results

- It was possible to build a simple circuit on IBM Q to simulate Grover's algorithm on a 2- qubit register $(q_1, q_2,)$ with Pauli gates (X) and Hadamard (H) gates

2-qubit Grover's Algorithm Circuit

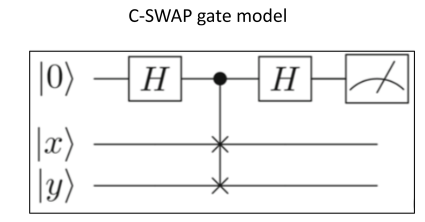


Simulating Fidelity

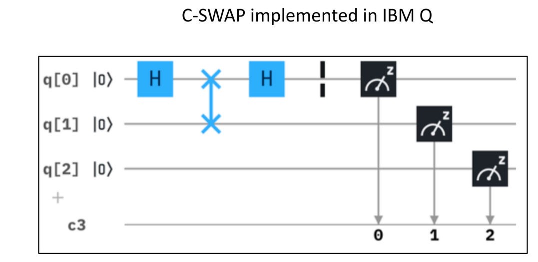
1. What is Fidelity?

- Before we compute distance for the QKNN, we must simulate a key property of classical machine learning known as **fidelity**
- Fidelity allows us to navigate past the issue of quantum collapse or partial loss of quantum information (lack of fault-tolerance)
- Fidelity measures the similarity of two quantum states (|ϕ⟩,|ψ⟩)
- QEML is not possible without fidelity since it is the quantum analog of **cosine similarity** (a technique commonly used in classic ML to represent the difference between 2 vectors)
- We can build a $C-SWAP$ circuit which simulates Fidelity

C-SWAP gate model



C-SWAP implemented in IBM Q



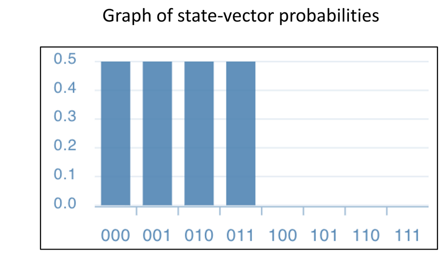
2. Simulation Results

- It was possible to build and execute a C-SWAP circuit on a quantum computer for a register of 3 qubits (q_1, q_2, q_3) and 3 bits
- The results were promising when simulated with the state-vector and BasicAer backend since when the states were orthogonal, the output was 0.5, whereas when they were identical, it was 0

Unitary State-Vector Output

```
[ 0.5+0j, 0.5+0j, 0.5+0j, 0.5+0j,
0+0j, 0+0j, 0+0j, 0+0j ]
```

Graph of state-vector probabilities

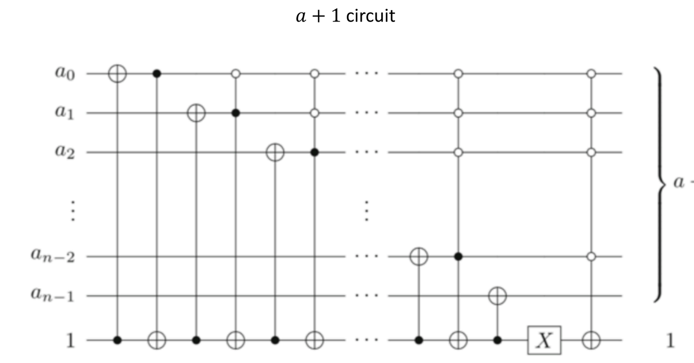


Main Experiment: Constructing and Simulating a Quantum K-Nearest Neighbors (QKNN)

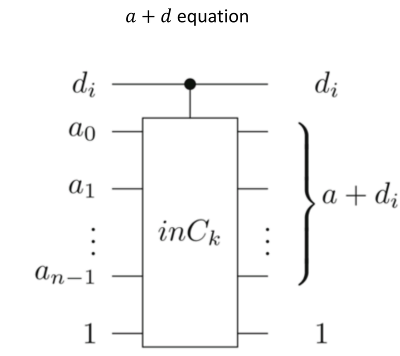
Technique: Use Hamming Distance as Metric

- Used in manipulating quantum state such as phase estimation and tomography. As earlier discussed, to build a Quantum K-nearest neighbors algorithm (QKNN), we need a method to find the distances between vectors (data points) in a quantum feature space
- **Hamming distance** was found to be the most optimal metric since it works well in higher dimensions and solves the "*Curse of Dimensionality*"
- Hamming Distance is defined as the number of positions at which the corresponding symbols of two-bit vectors of equal length are different
- This is best illustrated with a few examples: 00101↔00101 = 0, 00101↔00111 = 1, 00101↔10111 = 2
- The Hamming distance metric always meets 3 criteria: 1.) non-negative, 2.) symmetric in nature, 3.) satisfies the triangle inequality.

$a + 1$ circuit



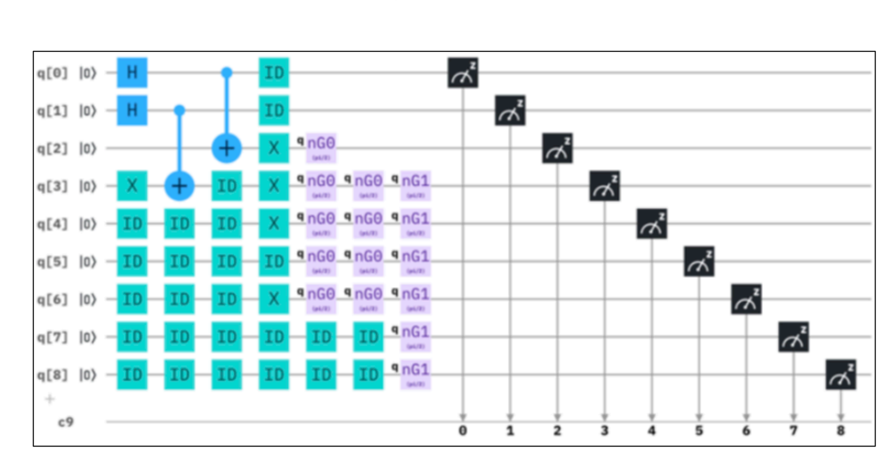
$a + d$ equation



Implementation: Use Fidelity and Hamming Distance

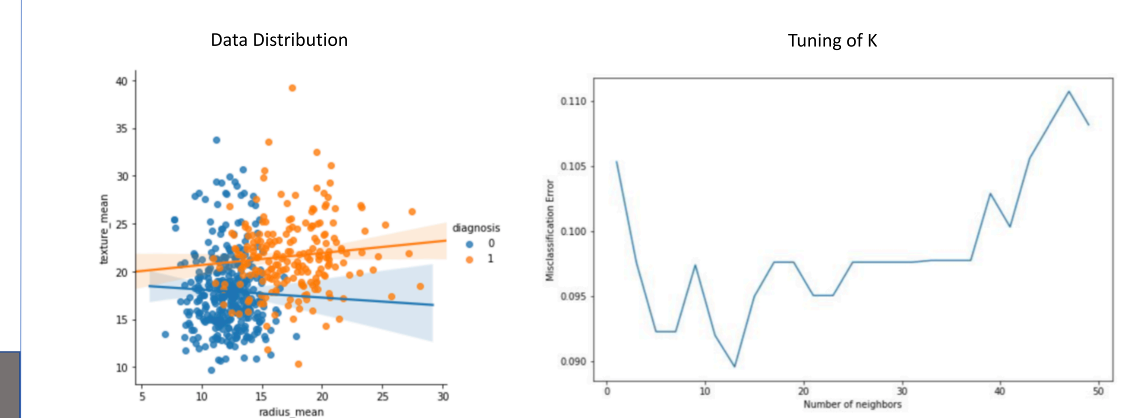
- Step 1: Map the features of the data to their ground quantum states in Hilbert space
  - Makes it easier to select K nearest neighbors
- Step 2: Instantiate a 9-qubit quantum register in order to demonstrate quantum parallelism
- Step 3: Build the core module $a + 1$ circuit (from Kaye)
- Step 4: Do addition between ancillary qubits (i.e. if qubit a_0 is flipped, it is transformed to the next qubit $a + 1$
  - If $a[i]$ is flipped from 1 to 0, then addition continues, otherwise reset $a[i]$ to 1
- Step 5: Expand from core $a + 1$ circuit to "$a + d$" circuit by applying CNOT and X gate
- Step 6: Label nearest neighbors after checking Hamming distances $(\sum_i d^p{}_i)$
- Step 7: Measure final outputs for each qubit via Z-measurement gates

Final QKNN variational circuit



Comparison to KNN on Breast-Cancer Dataset

- It is necessary to compare the performance of the implemented QKNN against its traditional counterpart to identify whether quantum computing truly provides a benefit for machine learning
- We previously summarized the benefits of QEML as: 1.) **more storage capabilities and smaller solution spa**ce (due to enhanced feature space via superposition) 2.) **faster execution of quantum algorithms** due to quantum parallelization.
- The chosen dataset for this comparison was the *Wisconsin Breast Cancer Dataset* since it has n = 10 dimensions (multivariate)
- 10 dimensions is considered optimal for gauging the performance of QEML against traditional machine learning since it is enough dimensions such that QEML should provide a tangible benefit but is not too large in that both ML and QEML will become computationally expensive.
- We can first implement a KNN algorithm for the classification of tumors as either malignant (1) or benign (0).

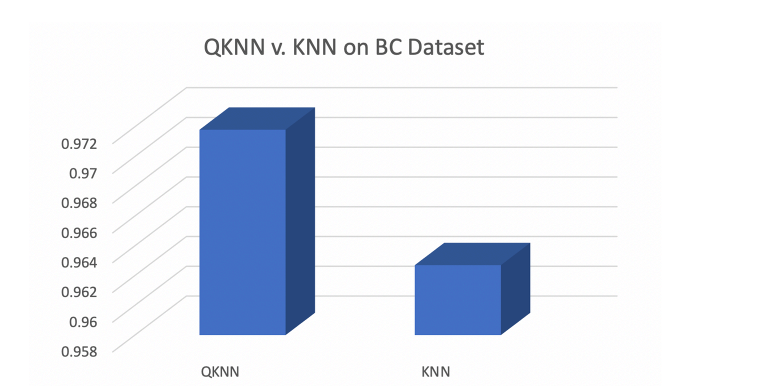


- Algorithm uses 65% train, 35% test split and utilizes Euclidean distance
- Did not require hyperparameter optimization since the only value that needed to be adjusted was *K*
- In this case, the optimal *K* (that produces minimal loss and misclassification) was *K* = 13 (shown in the tuning graph above)

Results

Used the Scikit-Learn Accuracy Score metric and IBM QASM simulator (1024 shots)

QKNN v. KNN on BC Dataset



QKNN: 0.9718
KNN: 0.9627

Interpretation of Results

- Although this paper has proved that quantum computing applied to machine learning (QEML) does provide some intermediate benefit, we cannot definitively argue that QEML is superior to machine learning and will be the future of big data computation.
- These results were somewhat randomized and were computed on a limited dataset of n = 10 dimensions, so the conclusions can only be taken with a grain of salt. (not statistically significant)
- However, we were definitively able to simulate both the property of fidelity (analogous to cosine similarity in classic machine learning) and the path of Grover's algorithm for 2 qubits (analogous to minimum path search in classic machine learning).
- It was not possible to do a time execution analysis since it was not a fair comparison to compare the arbitrary time for scikit-learn's KNN classifier against the time for the QKNN circuit on a variable number of shots
- Necessary to do a **cost-analysis** since it allows us to conclude whether the benefit is worth-it or not

Thoughts and Future Research

- We do not yet have access to fault-tolerant quantum computers and **decoherence** (partial collapse of quantum information) is still a common issue
- Quantum computers today have advanced in that they can simulate the noise and perturbations which affect physical quantum systems
- The success of the above experiments is a positive step for the field of quantum computing since it represents the synchronization of machine learning in quantum computing to benefit society.
- Using an enhanced feature space, we can harness the power of quantum computing to improve both the **efficiency** and **scalability** of machine learning algorithms

Selected References

- [1] Vojtech Havlicek, Antonio D. Corcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum enhanced feature spaces. arXiv preprint arXiv: 1804.11326, 2018.
- [2] Grover, L.K.: A fast quantum mechanical algorithm for database search. ACM. 212-219 (1996)
- [3] Feynman, R.P.: Simulating physics with computers. Int. J. Theor. Phys. 21(6), 467-488 (1982)
- [4] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum algorithms for supervised and unsupervised machine learning. arXiv preprint arXiv:1307.0411, 2013.
- [5] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big feature and big data classification. arXiv preprint arXiv:1307.0471, 2013.
- [6] Maria Schuld, Nathan Killoran. Quantum machine learning in feature Hilbert spaces. arXiv preprint arXiv: 1803.07128, 2018
- [7] Yue Ran, Xiling Xue, Heng Liu, Jianing Tan, Xi Li. Quantum Algorithm for K-Nearest Neighbors Classification Based on the Metric of Hamming Distance. International Journal of Theoretical Physics 56(4) DOI: 10.1007/s10773-017-3514-4. 2017.
- [8] Ning Yang. KNN Algorithm Simulation Based on Quantum Information. Proceedings of Student-Faculty Research Day Conference, CSIS, Pace University, May 3rd, 2019.
- Additional figures (KNN) and tuning of K are from TowardsDataScience and MLMastery