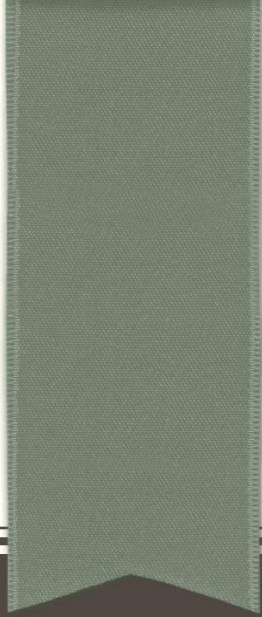


**QUANTUM MACHINE
LEARNING ALGORITHMS**
الگوریتم های یادگیری ماشین کوانتومی

Mohammad Ali Jafarizadeh
with
Marziyeh Yahyavi, Ahmad Heshmati, Naser Karimi
University of Tabriz



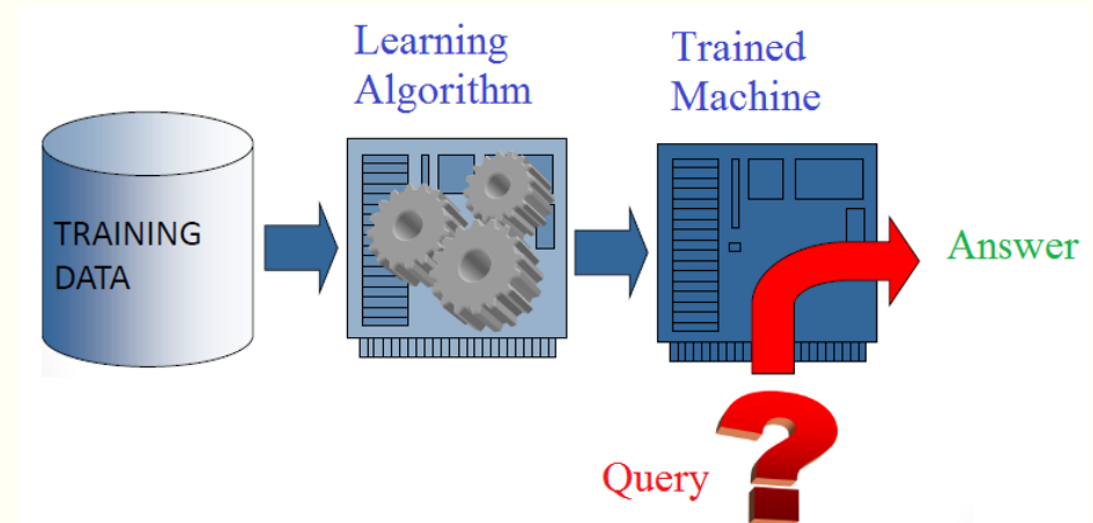


MACHINE LEARNING

What is Machine learning?

- “Learning is any process by which a system improves performance from experience.”

-Herbert Alexander Simon



Types of Learning

- **Supervised learning:**

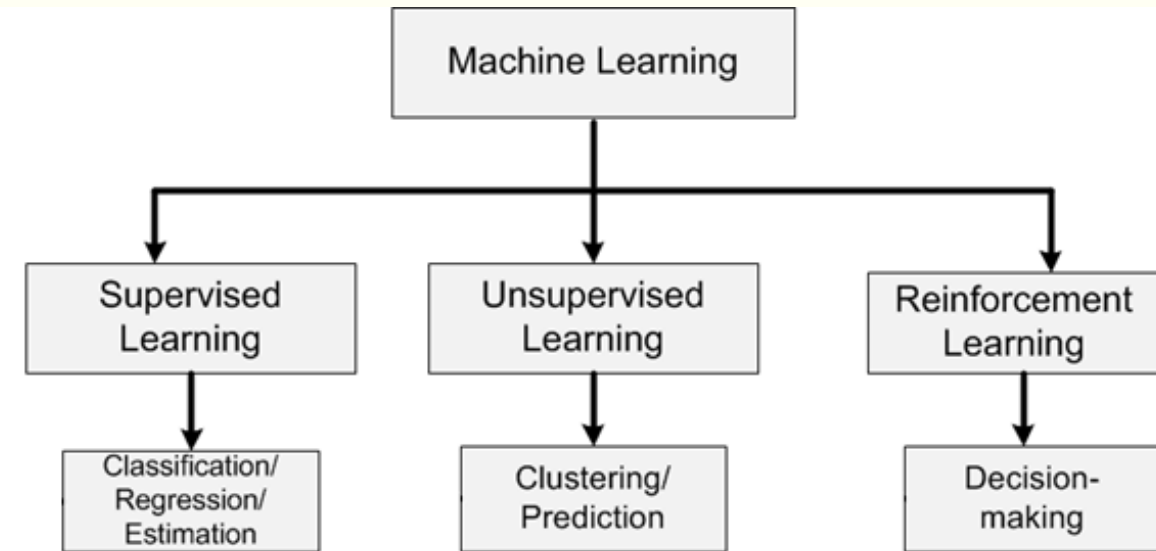
- The training examples along with the class labels are utilized to generate a predictive model.

- **Unsupervised learning:**

- Patterns or structures are found in data and labelled appropriately.

- **Reinforcement learning:**

- A Machine Learning method that is concerned with how agents should take actions in an environment.



Examples of Supervised & Unsupervised Learning

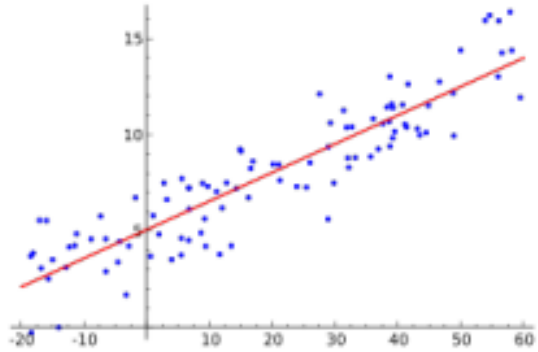


Figure 1: Linear Regression

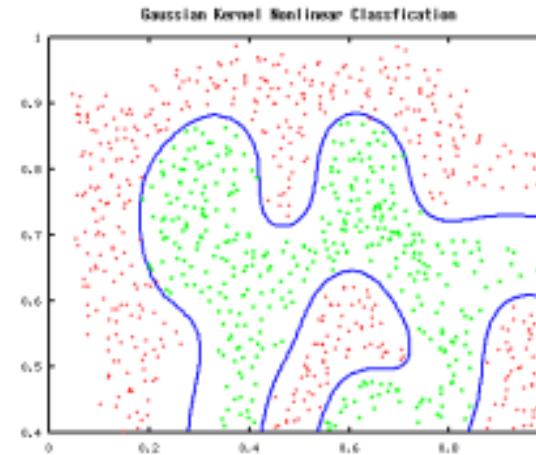


Figure 2: Classification

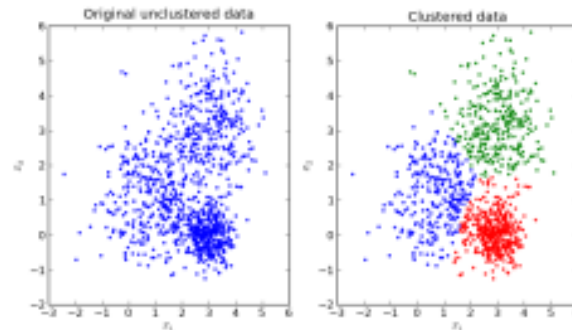


Figure 3: Clustering

Example of Reinforcement Learning

- How should a robot behave so as to optimize its “performance” ?

(Robotics)



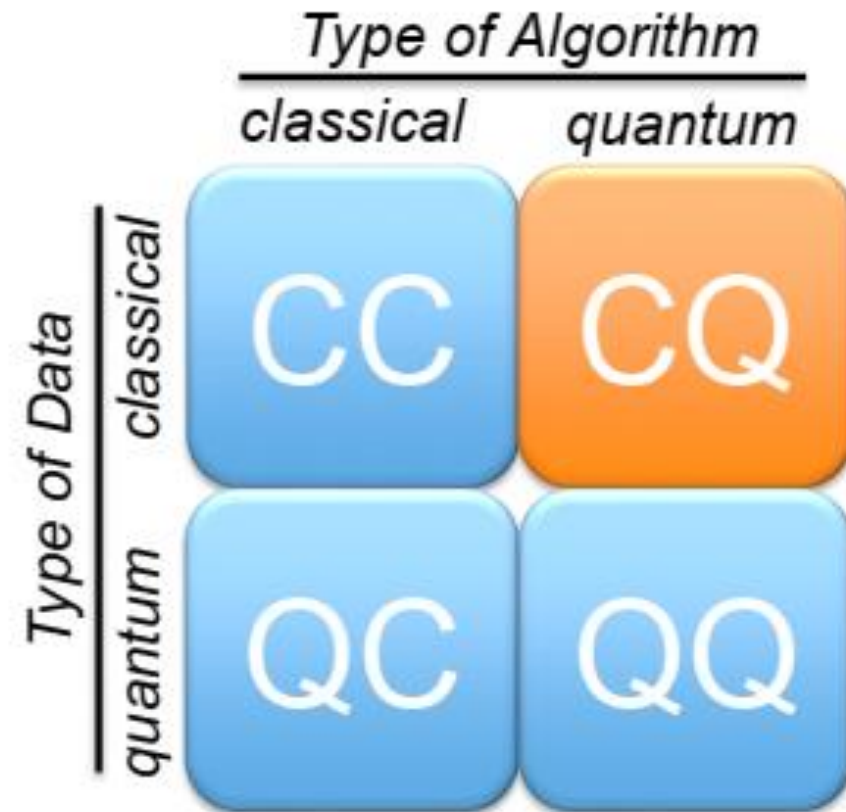


QUANTUM MACHINE LEARNING

Methods of Combining Quantum Computing & Machine Learning

In [1] four approaches are distinguished for combining quantum computing and machine learning

- **CC:** Classical data being processed classically
- **QC:** Quantum data being processed by classical computers
- **CQ:** Using quantum computing to process classical datasets
- **QQ:** Quantum data being processed by quantum computers





QC CASE

Exploring the quantum entanglement via the machine learning approach

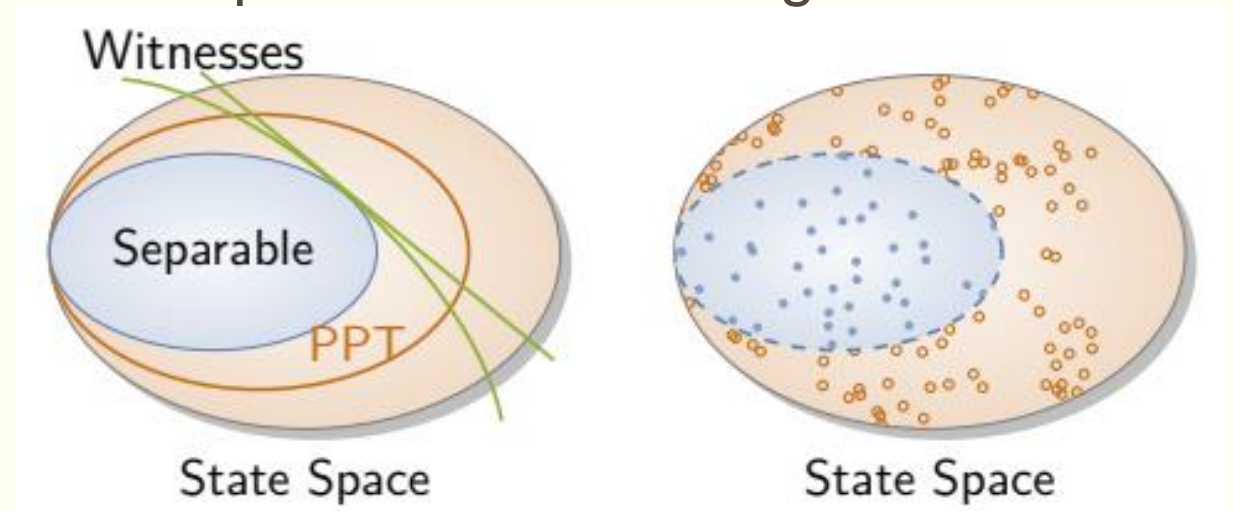
Methods of Detecting Bipartite Entanglement

- **Positive Partial Transpose (PPT) Criterion**

A separable state must have a PPT. however, it is only necessary and sufficient when $d_A d_B \leq 6$

- **Entanglement Witnesses**

The different entangled states often require different entanglement witnesses.



Machine Learning: Feature Vector Representation

- Any quantum state ρ as a density operator acting on $H_A \otimes H_B$ can be represented as a real vector x in $\mathcal{H} = \mathcal{R}^{d_A^2 d_B^2 - 1}$
- We refer to x as the **feature vector** of ρ and \mathcal{H} as the feature space.
- To represent a $n \times n$ density matrix ρ as a real vector x , we take the **generalized Gell-Mann matrices** and the **identity matrix** as the Hermitian orthogonal basis.

Generalized Gell-Mann Matrices & Feature Vectors

Example

- Considering the computational basis of the n dimensional Hilbert space $\{|1\rangle, \dots, |n\rangle\}$ and $E_{j,k} = |j\rangle\langle k|$, three collections of matrices can be defined

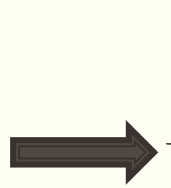
- Symmetric: $S_{j,k} = E_{j,k} + E_{k,j}$ for $1 \leq j < k \leq n$
- Antisymmetric: $A_{j,k} = -i(E_{j,k} - E_{k,j})$ for $1 \leq j < k \leq n$
- Diagonal: $d_l = \sqrt{\frac{2}{l(l+1)}} (\sum_{j=1}^l E_{j,j} - lE_{l+1,l+1})$ for $1 \leq l \leq n - 1$

single-qubit: $\rho = \frac{1}{2}(I_2 + a \cdot \sigma) \Rightarrow$

$$\begin{cases} \lambda_i = \{\sigma_x, \sigma_y, \sigma_z\} \\ \text{feature vector: } \mathbf{x}: (a_x, a_y, a_z) \end{cases}$$

- Generalized Gell-Mann Matrices: $\{\lambda_i\} = \{s_{j,k}\} \cup \{a_{j,k}\} \cup \{d_l\}$

$$\rho = \frac{1}{n} \left(\mathbb{I} + \sqrt{\frac{n(n-1)}{2}} \mathbf{x} \cdot \vec{\lambda} \right)$$



$$\begin{cases} \mathbf{x} = (x_1, x_2, \dots, x_{n^2-1}) \in \mathbb{R}^{n^2-1} \\ x_i = \sqrt{\frac{n}{2(n-1)}} \text{tr}(\rho \lambda_i) \end{cases}$$

Feature vector representation

Training Dataset

- A dataset of training examples is produced, with the form

$$\mathcal{D}_{\text{train}} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$$

which

n : the size of the set
 x_i : the i -th sample
 y_i : the i -th label



$y_i = 1$, if x_i is entangled,
 $y_i = -1$, if x_i is separable.

Inferring a Classifier

The aim of supervised learning is to infer a classifier $h: \mathcal{H} \rightarrow \{-1,1\}$, where h is expected to be close to the true decision function.

To evaluate how well h fits the training data \mathcal{D}_{train} , a loss function is defined as

$$\mathcal{L}(h, \mathcal{D}_{train}) = \frac{1}{|\mathcal{D}_{train}|} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{train}} \mathbf{1}(y_i \neq h(\mathbf{x}_i))$$

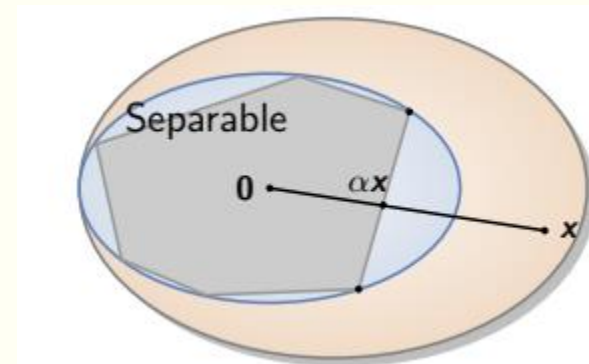
Method: Convex Hull Approximation (CHA)

- We randomly sample m separable pure states $\mathbf{c}_1, \dots, \mathbf{c}_m \in \mathcal{H}$ to form a convex hull $\mathcal{C} = \text{conv}(\{\mathbf{c}_1, \dots, \mathbf{c}_m\})$.
- With \mathcal{C} , we can approximately tell whether a state ρ is separable or not by testing if it is in \mathcal{C} .

The error rate of CHA \mathcal{C}_m for two-qubit separable set for some critical m .

m	1000	2000	3000	4000	5000
Error of CHA (%)	8.55	6.01	4.85	4.05	3.60
m	6000	7000	8000	9000	10 000
Error of CHA (%)	3.25	2.95	2.76	2.64	2.55

$$\begin{aligned} \max \quad & \alpha \quad \text{s.t.} \quad \alpha \mathbf{x} \in \mathcal{C}, \\ \text{i.e.} \quad & \alpha \mathbf{x} = \sum_{i=1}^m \lambda_i \mathbf{c}_i, \quad \lambda_i \geq 0, \quad \sum_i \lambda_i = 1. \end{aligned}$$



Bell Diagonal Example

$$\rho = \sum_{i=1}^4 p_i |\psi_i\rangle \langle \psi_i|, \quad 0 \leq p_i \leq 1, \quad \sum_{i=1}^4 p_i = 1,$$

$$|\psi_1\rangle = |\phi^+\rangle = \frac{1}{\sqrt{2}}(|\uparrow\uparrow\rangle + |\downarrow\downarrow\rangle),$$

$$|\psi_2\rangle = |\phi^-\rangle = \frac{1}{\sqrt{2}}(|\uparrow\uparrow\rangle - |\downarrow\downarrow\rangle),$$

$$|\psi_3\rangle = |\psi^+\rangle = \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle + |\downarrow\uparrow\rangle),$$

$$|\psi_4\rangle = |\psi^-\rangle = \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle).$$

Bell Diagonal Example

Step1

Creating the feature vector of density matrix by writing the Gell-Mann form of it

$$\rho = \frac{1}{4} \left(I \otimes I + \sum_{i=1}^3 t_i \sigma_i \otimes \sigma_i \right), \quad \begin{aligned} t_1 &= p_1 - p_2 + p_3 - p_4, \\ t_2 &= -p_1 + p_2 + p_3 - p_4, \\ t_3 &= p_1 + p_2 - p_3 - p_4. \end{aligned}$$

Feature Vector of ρ

$$\mathbf{x} = (t_1, t_2, t_3)$$

Positivity conditions of eigenvalues

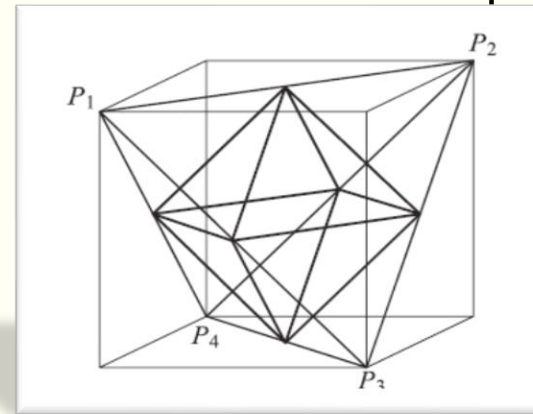
$$\begin{aligned} 1 + t_1 - t_2 + t_3 &\geq 0, \\ 1 - t_1 + t_2 + t_3 &\geq 0, \\ 1 + t_1 + t_2 - t_3 &\geq 0, \\ 1 - t_1 - t_2 - t_3 &\geq 0. \end{aligned}$$

PPT conditions

$$\begin{aligned} 1 + t_1 + t_2 + t_3 &\geq 0, \\ 1 - t_1 - t_2 + t_3 &\geq 0, \\ 1 + t_1 - t_2 - t_3 &\geq 0, \\ 1 - t_1 + t_2 - t_3 &\geq 0. \end{aligned}$$

Step2

Producing a dataset of training examples



$$\begin{aligned} c_1 : \{1, 0, 0\} &\equiv \frac{1}{2} (|\psi_1\rangle\langle\psi_1| + |\psi_3\rangle\langle\psi_3|) \\ c_2 : \{-1, 0, 0\} &\equiv \frac{1}{2} (|\psi_2\rangle\langle\psi_2| + |\psi_4\rangle\langle\psi_4|) \\ c_3 : \{0, 1, 0\} &\equiv \frac{1}{2} (|\psi_2\rangle\langle\psi_2| + |\psi_3\rangle\langle\psi_3|) \\ c_4 : \{0, -1, 0\} &\equiv \frac{1}{2} (|\psi_1\rangle\langle\psi_1| + |\psi_4\rangle\langle\psi_4|) \\ c_5 : \{0, 0, 1\} &\equiv \frac{1}{2} (|\psi_1\rangle\langle\psi_1| + |\psi_2\rangle\langle\psi_2|) \\ c_6 : \{0, 0, -1\} &\equiv \frac{1}{2} (|\psi_3\rangle\langle\psi_3| + |\psi_4\rangle\langle\psi_4|) \end{aligned}$$

$$O_1^\pm = (\pm 1, 0, 0), O_2^\pm = (0, \pm 1, 0) \text{ and } O_3^\pm = (0, 0, \pm 1).$$

$$\begin{aligned} \max \alpha \\ \text{s. t. } \alpha \mathbf{x} &= \sum_{i=0}^m \lambda_i \mathbf{c}_i, \\ \lambda_i &\geq 0, \quad \sum_i \lambda_i = 1. \end{aligned}$$

Convex hull

$$C = \text{conv}(\{c_1, c_2, \dots, c_6\})$$

Werner State

$$\rho \equiv (-\tau, -\tau, -\tau) \quad \rho \geq 0 \Rightarrow -\frac{1}{3} \leq \tau \leq 1$$

$$\text{Positivity condition: } \begin{cases} 1 - \alpha\tau \geq 0 \\ 1 + 3\alpha\tau \geq 0 \end{cases} \Rightarrow \begin{cases} \text{If } \tau \leq 0 \Rightarrow 1 + 3\alpha\tau \geq 0 \Rightarrow \alpha \geq -\frac{1}{3\tau} \\ \text{If } \tau \geq 0 \Rightarrow 1 - \alpha\tau \geq 0 \Rightarrow \alpha \leq \frac{1}{\tau} \end{cases}$$

$$\text{PPT condition: } \begin{cases} 1 - 3\alpha\tau \geq 0 \\ 1 + \alpha\tau \geq 0 \end{cases} \Rightarrow \begin{cases} \text{If } \tau \leq 0 \Rightarrow 1 + \alpha\tau \geq 0 \Rightarrow \alpha \geq -\frac{1}{\tau} \\ \text{If } \tau \geq 0 \Rightarrow 1 - 3\alpha\tau \geq 0 \Rightarrow \alpha \leq \frac{1}{3\tau} \end{cases}$$

$$\text{Result: } \begin{cases} \text{If } \tau \leq 0 \Rightarrow \max \alpha = -\frac{1}{\tau} \geq 1 \Rightarrow \rho \text{ is separable} \\ \text{If } \tau \geq 0 \Rightarrow \max \alpha = \frac{1}{\tau} \geq 1 \Rightarrow \rho \text{ is separable} \end{cases} \Rightarrow \rho \text{ is separable}$$



SOME CQ CASES

CQ: Quantum Computing Help Machine Learning

- **Core idea:** Inputs to learning problem are often high-dimensional vectors of numbers (texts, images, ...).
- Required number of qubits is only logarithmic in dimension!
- Vector $v \in \mathbb{R}^d \Rightarrow \log_2(d)$ -qubit state $|v\rangle = \frac{1}{\|v\|} \sum_{i=1}^d v_i |i\rangle$
- So we want to efficiently represent our data as quantum states, and apply quantum algorithms on them to learn.



QUANTUM ASSOCIATIVE MEMORY

An example of CQ

What Is Associative Memory?

- Store a set of fundamental memories $\{\xi_1, \xi_2, \dots, \xi_M\}$, so that, when presented a new pattern x , the system outputs one of the stored memories that is most similar to x .

Purpose of Quantum Associative Memory

- The main purpose of the quantum associative memory built by Ventura and Martinez is **pattern completion**, where the full pattern can be restored from partial pattern.
- The memory use a storage algorithm and Grover's quantum search algorithm for retrieving the patterns.

Two Tasks of Quantum Associative Memory

Task 1

- **Preparing Superpositions of Inputs**

Task 2

- **Retrieves the patterns.**

Algorithm of Quantum Associative Memory

1. Generate the initial state $|\psi\rangle = |\bar{0}\rangle$
2. For $m \geq p \geq 1$
3. $FLIP|\psi\rangle$
4. $\hat{S}^p|\psi\rangle$
5. $SAVE|\psi\rangle$
6. $|\psi\rangle = \hat{I}_\tau|\psi\rangle$
7. $|\psi\rangle = \hat{G}|\psi\rangle$
8. $|\psi\rangle = \hat{I}_p|\psi\rangle$
9. $|\psi\rangle = \hat{G}|\psi\rangle$
10. Repeat T times
11. $|\psi\rangle = \hat{I}_\tau|\psi\rangle$
12. $|\psi\rangle = \hat{G}|\psi\rangle$

- With $2n + 1$ qubits, the QuAM can store up to $N = 2^n$ patterns, where N is the total number of basis states, in $O(mn)$ steps and requires $O(\sqrt{N})$ time to recall a pattern.

$$\hat{I}_\phi = \text{identity matrix except for } \phi\phi = -1, \quad \mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

$$|\bar{0}\rangle = |0, \dots, 0\rangle$$

$$\hat{G} = -\mathbf{H}\hat{I}_0\mathbf{H}$$

$$\text{Times} \begin{cases} T_{Grover} = \frac{\pi}{4} \sqrt{N} \\ T_{\text{complete pattern}} = T_{Grover} - 2 \\ T_{\text{incomplete pattern}} = T \end{cases}$$

Time-Step for Incomplete Pattern Retrieving

- r_1 : the number of marked states that correspond to stored patterns
- r_0 : the number of marked states that do not correspond to stored patterns
- N : the total number of basis states
- p : the number of patterns stored in the QuAM

$$\left\{ \begin{array}{l} a = \frac{2(p - 2r_1)}{N} \\ b = \frac{4(p + r_0)}{N} \end{array} \right. \longrightarrow \left\{ \begin{array}{l} \bar{k} = 4a - ab + \frac{r_1}{r_0 + r_1} \\ \bar{l} = -ab + \frac{2a(N + p - r_0 - 2r_1)}{N - r_0 - r_1} - \frac{(p - r_1)}{N - r_0 - r_1} \end{array} \right.$$

$$T = \frac{\frac{\pi}{2} - \arctan \left((\bar{k}/\bar{l}) \sqrt{(r_0 + r_1)/(N - r_0 - r_1)} \right)}{\arccos \left(1 - 2((r_0 + r_1)/N) \right)}$$



Task 1

- **Preparing Superpositions of Inputs**

Basis Encoding

- Assume we are given a binary dataset D where each pattern $x^m \in D$ is a binary string of the form $x^m = (b_1^m, \dots, b_N^m)$ with $b_i^m \in \{0,1\}$ for $i = 1, \dots, N$.
- We can prepare a superposition of basis states $|x^m\rangle$ that qubit-wise correspond to the binary input patterns,

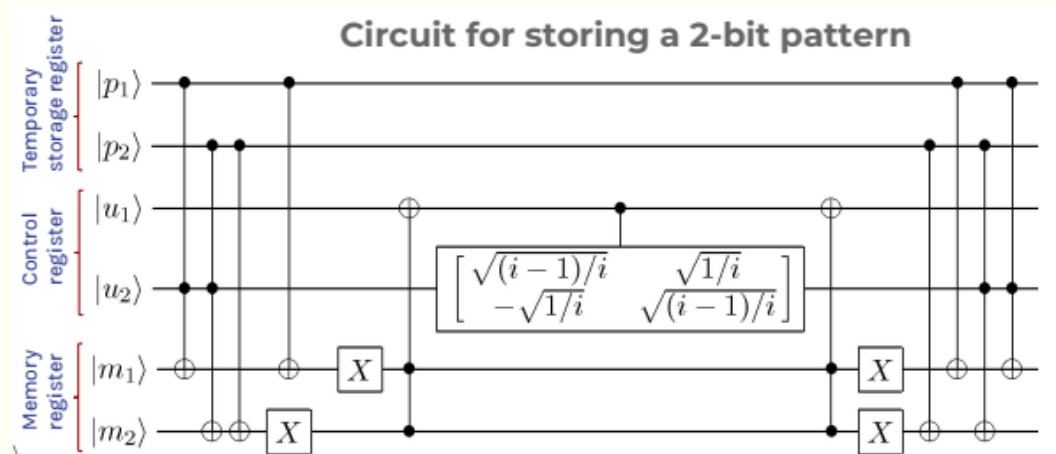
$$|\mathcal{D}\rangle = \frac{1}{\sqrt{M}} \sum_{m=1}^M |x^m\rangle$$

Example: binary inputs $\begin{cases} x^1 = (01, 01)^T \\ x^2 = (11, 10)^T \end{cases} \xrightarrow{\text{Corresponding binary patterns}} \begin{cases} x^1 = (0110) \\ x^2 = (1110) \end{cases}$

$$|\mathcal{D}\rangle = \frac{1}{\sqrt{2}} |0101\rangle + \frac{1}{\sqrt{2}} |1110\rangle$$

Circuit for One Step of Ventura and Martinez's State Preparation

Summary of an elegant way to construct **data superpositions** introduced by Ventura, Martinez and others



$$1. |\psi_0^1\rangle = |p_1^1, \dots, p_d^1; 01; 0_1, \dots, 0_d\rangle$$

$$2. |\psi_1^i\rangle = \prod_{j=1}^d {}^{2c} \hat{X}_{p_j^i u_2 m_j} |\psi_0^i\rangle$$

$$3. |\psi_2^i\rangle = \prod_{j=1}^d \hat{X}_{m_j} {}^{1c} \hat{X}_{p_j^i m_j} |\psi_1^i\rangle$$

$$4. |\psi_3^i\rangle = {}^{dc} \hat{X}_{m_1 \dots m_d u_1} |\psi_2^i\rangle$$

$$5. |\psi_4^i\rangle = {}^{1c} \hat{S}_{u_1 u_2} (p+1-i) |\psi_3^i\rangle$$

$$6. |\psi_5^i\rangle = {}^{dc} \hat{X}_{m_1 \dots m_d u_1} |\psi_4^i\rangle$$

$$7. |\psi_6^i\rangle = \prod_{j=d}^1 {}^{1c} \hat{X}_{p_j^i m_j} \hat{X}_{m_j} |\psi_5^i\rangle$$

$$= \frac{1}{\sqrt{p}} \sum_{k=1}^i |p^i; 00; p^k\rangle + \sqrt{\frac{p-i}{p}} |p^i; 01; p^i\rangle$$

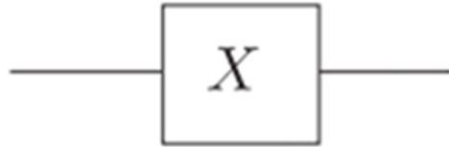
$$8. |\psi_7^i\rangle = \prod_{j=d}^1 {}^{2c} \hat{X}_{p_j^i u_2 m_j} |\psi_6^i\rangle$$

Ventura, D., Martinez, T.: Quantum associative memory. *Inf. Sci.* 124(1), 273–296 (2000).

Trugenberger, Carlo A. "Probabilistic quantum memories." *Physical Review Letters* 87.6 (2001): 067901.

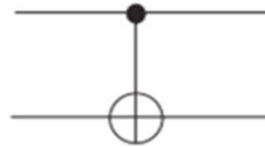
Quantum Gates

Not gate



$$a_0|0\rangle + a_1|1\rangle \xrightarrow{X} a_0|1\rangle + a_1|0\rangle$$

CNOT gate



$$\underbrace{(a_0|0\rangle + a_1|1\rangle)}_{\text{control}} \underbrace{(b_0|0\rangle + b_1|1\rangle)}_{\text{target}} \xrightarrow{\text{CNOT}} a_0b_0|0\rangle|0\rangle + a_0b_1|0\rangle|1\rangle + a_1b_0|1\rangle|1\rangle + a_1b_1|1\rangle|0\rangle$$

Venture and Martinez gate operator

$$S^J = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \sqrt{\frac{I-1}{J}} & \frac{1}{\sqrt{J}} \\ 0 & 0 & \frac{-1}{\sqrt{J}} & \sqrt{\frac{I-1}{J}} \end{bmatrix}$$

Tofolli gate.



$$T^2|uvw\rangle = |uv, (w + (u \cdot v)) \bmod 2\rangle$$

$$T^j|u_1u_2\dots u_ju_{j+1}\rangle = |u_1u_2\dots u_j, (u_{j+1} + \prod_{n=1}^j u_n) \bmod 2\rangle$$



A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Example of State Preparation

patterns: {000,011,100,110}

Storage the fourth pattern **110**

$$\begin{aligned} \text{step1: } |\psi_0\rangle &= |110, 000, 01\rangle \\ \text{step2: } |\psi_1\rangle &= T^2_{p,f_2,q_1} |\psi_0\rangle = |110, 110, 01\rangle \\ \text{step3: } |\psi_2\rangle &= NOT_{q_1} XOR_{p,q_1} |\psi_1\rangle = |110, 111, 01\rangle \\ \text{step4: } |\psi_3\rangle &= T^3_{q_2,q_3,f_1} |\psi_2\rangle = |110, 111, 11\rangle \\ \text{step5: } |\psi_4\rangle &= S^4_{c_1,c_2} |\psi_3\rangle = \frac{1}{2} |110, 111, 10\rangle + \frac{\sqrt{3}}{2} |110, 111, 11\rangle \\ \text{step6: } |\psi_5\rangle &= T^3_{q_2,q_3,f_1} |\psi_4\rangle = \frac{1}{2} |110, 111, 00\rangle + \frac{\sqrt{3}}{2} |110, 111, 01\rangle \\ \text{step7: } |\psi_6\rangle &= XOR_{p,q_1} NOT_{q_1} |\psi_5\rangle = \frac{1}{2} |110, 110, 00\rangle + \frac{\sqrt{3}}{2} |110, 110, 01\rangle \\ \text{step8: } |\psi_7\rangle &= T^2_{p,f_2,q_1} |\psi_6\rangle = \frac{1}{2} |110, 110, 00\rangle + \frac{\sqrt{3}}{2} |110, 000, 01\rangle \end{aligned}$$

Storage Result

$$|\psi\rangle = \frac{1}{2} (|000\rangle + |011\rangle + |100\rangle + |110\rangle)$$



Task 2

- **Pattern Retrieving**

Example 1: Complete Pattern Retrieving by Ventura-Martinez Version of Grover Search

patterns: $\{p^1 = 110, p^2 = 011, p^3 = 010\}$ test pattern: 110

$$|s\rangle = \frac{1}{\sqrt{8}} \sum_{i,j,k=0}^1 |ijk\rangle$$

$$G = 2|s\rangle\langle s| - I$$

$$\tau = |110\rangle$$

$$|\psi\rangle = \frac{1}{\sqrt{3}} (0, 0, 1, 1, 0, 0, 1, 0)$$

$$|\psi\rangle \xrightarrow{I_\tau} |\psi\rangle = \frac{1}{\sqrt{3}} (0, 0, 1, 1, 0, 0, -1, 0)$$

$$|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{4\sqrt{3}} (1, 1, -3, -3, 1, 1, 5, 1)$$

$$|\psi\rangle \xrightarrow{I_P} |\psi\rangle = \frac{1}{4\sqrt{3}} (1, 1, 3, 3, 1, 1, -5, 1)$$

$$|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{8\sqrt{3}} (1, 1, -3, -3, 1, 1, 13, 1)$$

Result

The probability of collapsing into the desired $|110\rangle$ is **88%**.

Example 2: Incomplete Pattern Retrieving by Common Grover Search

- We want to amplify the amplitude of search string **10?** in a sparse uniform superposition,

$$|\psi\rangle = \frac{1}{2}(|000\rangle + |011\rangle + |100\rangle + |110\rangle)$$

which in vector notation corresponds to the amplitude vector: $|\psi\rangle = \frac{1}{2}(1, 0, 0, 1, 1, 0, 1, 0)^T$

- In conventional Grover search, the first step is to mark the target state by a negative phase,

$$|\psi\rangle \xrightarrow{I_\tau} |\psi\rangle = \frac{1}{2}(1, 0, 0, 1, -1, 0, 1, 0)^T$$

and then apply G : $|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{4}(-1, 1, 1, -1, 3, 1, -1, 1)^T$

- In the second iteration we mark again the target state: $|\psi\rangle \xrightarrow{I_\tau} |\psi\rangle = \frac{1}{4}(-1, 1, 1, -1, -3, -1, -1, 1)^T$

and apply again G : $|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{2}(0, -1, -1, 0, 1, 0, 0, -1)^T$

The probability of collapsing into the desired 100 is 25%.

Example 2: Incomplete Pattern Retrieving by Ventura-Martinez Version of Grover Search

- Getting back to the previous example and applying the Ventura-Martinez trick, starting with the same initial state, marking the target, and ‘rotating’ the amplitudes for the first time, we have

$$|\psi\rangle = \frac{1}{2}(1,0,0,1,1,0,1,0)^T \Rightarrow |\psi\rangle \xrightarrow{I_\tau} |\psi\rangle = \frac{1}{2}(1,0,0,1,-1,0,1,0)^T \Rightarrow |\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{4}(-1,1,1,-1,3,1,-1,1)^T$$

- The adapted routine ‘marks’ all amplitude that correspond to states in the data superposition,

$$|\psi\rangle \xrightarrow{I_P} |\psi\rangle = \frac{1}{4}(1,1,1,1,-3,-1,1,1)$$

- and applying G , we have: $|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{8}(-1,3,3,-1,-5,3,-1,3)$

At this point, there is a 53% probability of observing the system and finding the state 10?. Of course there are two states that match and state 100 has a 39% chance while state 101 has a 14%.

Example 3: Incomplete Pattern Retrieving by Ventura-Martinez Version of Grover Search

patterns : 000,011,100,110,101

test pattern: 10?

Preparing

- Similar to the previous example, using the algorithm for storing patterns, we have:

$$|\psi\rangle = \frac{1}{\sqrt{5}}(000+011+100+110+101)$$

At this point, there is a **24%** probability of observing the system and finding the state **10?**. Of course there are two states that match and states **100** and **101** have the same chance equal to **38%**.

Pattern Retrieving

$$|s\rangle = \frac{1}{\sqrt{8}} \sum_{i,j,k=0}^1 |ijk\rangle$$

$$G = 2|s\rangle\langle s| - I$$

$$\tau = |10?\rangle$$

$$|\psi\rangle = \frac{1}{2}(1,0,0,1,1,1,0)$$

$$|\psi\rangle \xrightarrow{I_\tau} |\psi\rangle = \frac{1}{\sqrt{5}}(1,0,0,1,-1,-1,0)$$

$$|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{4\sqrt{5}}(-3,1,1,-3,5,5,-3,1)$$

$$|\psi\rangle \xrightarrow{I_p} |\psi\rangle = \frac{1}{4\sqrt{5}}(3,1,1,3,-5,-5,3,1)$$

$$|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{8\sqrt{5}}(-5,-1,-1,-5,11,11,-5,-1)$$

Example 3: Incomplete Pattern Retrieving by Ventura-Martinez Version of Grover Search

patterns : 000,011,100,110,101

test pattern: 10?

Preparing

- Similar to the previous example, using the algorithm for storing patterns, we have:

$$|\psi\rangle = \frac{1}{\sqrt{5}}(000+011+100+110+101)$$

At this point, there is a **76%** probability of observing the system and finding the state **10?**. Of course there are two states that match and states **100** and **101** have the same chance equal to **38%**.

Pattern Retrieving

$$|s\rangle = \frac{1}{\sqrt{8}} \sum_{i,j,k=0}^1 |ijk\rangle$$

$$G = 2|s\rangle\langle s| - I$$

$$\tau = |10?\rangle$$

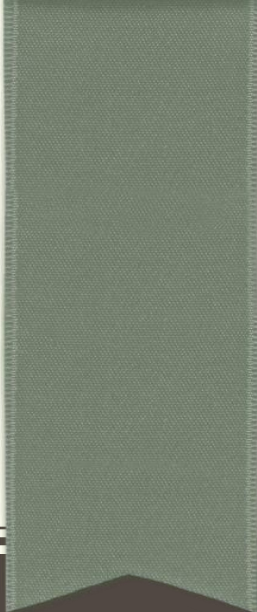
$$|\psi\rangle = \frac{1}{2}(1,0,0,1,1,1,0)$$

$$|\psi\rangle \xrightarrow{I_\tau} |\psi\rangle = \frac{1}{\sqrt{5}}(1,0,0,1,-1,-1,0)$$

$$|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{4\sqrt{5}}(-3,1,1,-3,5,5,-3,1)$$

$$|\psi\rangle \xrightarrow{I_P} |\psi\rangle = \frac{1}{4\sqrt{5}}(3,1,1,3,-5,-5,3,1)$$

$$|\psi\rangle \xrightarrow{G} |\psi\rangle = \frac{1}{8\sqrt{5}}(-5,-1,-1,-5,11,11,-5,-1)$$



QUANTUM PATTERN CLASSIFICATION

Another example of CQ case



SCHULD-SINAYSKIY-PETRUCCIONE (SSP) ALGORITHM

Quantum Nearest Neighbour Algorithms

Schuld-Sinayskiy-Petruccione (SSP) Algorithm

The algorithm is a quantum version of the classical weighted nearest neighbor algorithm.

Properties:

$$\text{if } \begin{cases} \mathbf{n} : \text{size of an input vector} \\ \mathbf{m} : \text{the number of input vectors in a training set} \\ \mathbf{D} : \text{a number of classes} \end{cases}$$

$$\xrightarrow{\substack{\text{properties of the algorithm SSP-kNN} \\ \text{for classifying a test vector}}} \begin{cases} \text{size(SSP-kNN)} = 2n + \log D + 1; \\ \text{time(SSP-kNN preliminary stage)} = O(mn) \\ \text{time(SSP-kNN main stage)} = O(n) \end{cases}$$

SSP Algorithm

Step 1

- Preparing a superposition of all of the training vectors into one quantum state: $|T\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^m |x_1^j, \dots, x_n^j, y^j\rangle$

Step 2

- Prepare the following initial state: $|\psi_0\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^m |x_1, \dots, x_n; x_1^j, \dots, x_n^j, y^j; 0\rangle$

Step 3

- Apply the Hadamard gate to the utility qubit: $|\psi_1\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^m |x_1, \dots, x_n; x_1^j, \dots, x_n^j, y^j\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

Step 4

- Apply the *CNOT* gate and *NOT* gate as follows: $|\psi_2\rangle = \prod_{s=1}^n \text{NOT}(x_s^j) \text{CNOT}(x_s, x_s^j) |\psi_1\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^m |x_1, \dots, x_n; d_1^j, \dots, d_n^j, y^j\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$

SSP Algorithm

Step 5

- Apply the unitary operator U: $U = e^{i\frac{\pi}{2n}\eta}$, $\eta = I \otimes D_{\text{Ham}}^j \otimes I \otimes \sigma_z$, $D_{\text{Ham}}^j = \sum_{s=1}^n \left(\frac{\sigma_z + I}{2} \right)$

Step 6

- Apply a Hadamard gate on the utility qubit,
- And write the phase information of the j -th state into amplitudes:

$$\left\{ \begin{array}{l} U_2 = I \otimes I \otimes I \otimes H; \\ |\psi_4\rangle = U_2 |\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{j=1}^m \\ \left(\cos \frac{\pi}{2n} d_{\text{Ham}}(x, x^j) |x_1, \dots, x_n; d_1^j, \dots, d_n^j, y^j; 0\rangle + \right. \\ \left. \sin \frac{\pi}{2n} d_{\text{Ham}}(x, x^j) |x_1, \dots, x_n; d_1^j, \dots, d_n^j, y^j; 1\rangle \right) \end{array} \right.$$

Step 7

- Measure the utility qubit. If the test vector is close to the training vector,
- Then the probability of a 0 result is high;
- Otherwise, the probability of a 1 result is high.

$$Pr_0 = \frac{1}{\sqrt{m}} \sum_{j=1}^m \cos^2 \left[\frac{\pi}{2n} d_{\text{Ham}}(x, x^j) \right]$$

Step 8

- If we obtain a 0 result, then the next step is to measure the “class register”.
- The probability of obtaining c -result from the “class register” measurement is:

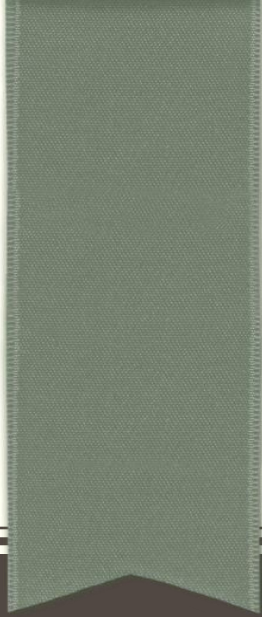
$$Pr(c) = \frac{1}{m \cdot Pr_0} \sum_{l: v^l=c} \cos^2 \left[\frac{\pi}{2n} d_{\text{Ham}}(x, x^l) \right]$$

SSP Algorithm

Algorithm 4 SPP- k NN(X, m, n, x). Schuld-Sinayskiy-Petruccione algorithm.

```
 $|T\rangle \leftarrow \text{Construct\_Superposition}(X, m, n)$   $\triangleright$  Constructing the  
superposition of the training set  $X: \frac{1}{\sqrt{m}} \sum_{j=1}^m |x_1^j, \dots, x_n^j,$   
 $y^j\rangle$   
 $|x\rangle \leftarrow |x_1, \dots, x_n\rangle$   $\triangleright$  test vector  
 $|u\rangle \leftarrow |0\rangle$   $\triangleright$  utility qubit  
 $|\psi\rangle \leftarrow |x\rangle |T\rangle |u\rangle$   $\triangleright$  initial state  
 $|\psi\rangle \leftarrow |x\rangle |T\rangle \otimes H |u\rangle$   $\triangleright H$  gate to the utility qubit  
 $|\psi\rangle \leftarrow \prod_{s=1}^n X(x_s) \text{CNOT}(x_s, x_s^j) |\psi\rangle$   
 $|\psi\rangle \leftarrow U |\psi\rangle$   $\triangleright$  applying operator  $U$  for calculating the  
Hamming distance  
 $|\psi\rangle \leftarrow |x\rangle |T\rangle \otimes H |u\rangle$   
 $ut \leftarrow \text{Measurement}(|u\rangle)$   $\triangleright$  measuring the utility qubit  
if  $ut$  is  $|0\rangle$  then  
     $c \leftarrow \text{Measurement}(|y\rangle)$   $\triangleright$  measure register  $|y\rangle$   
end if  
if  $ut$  is  $|1\rangle$  then  
     $c \leftarrow -1$   
end if  
return  $c$ 
```

$$U = e^{i\frac{\pi}{2n}\eta}, \quad \eta = I \otimes D_{\text{Ham}}^j \otimes I \otimes \sigma_z$$
$$D_{\text{Ham}}^j = \sum_{s=1}^n \left(\frac{\sigma_z + I}{2} \right),$$



CQ APPLICATION

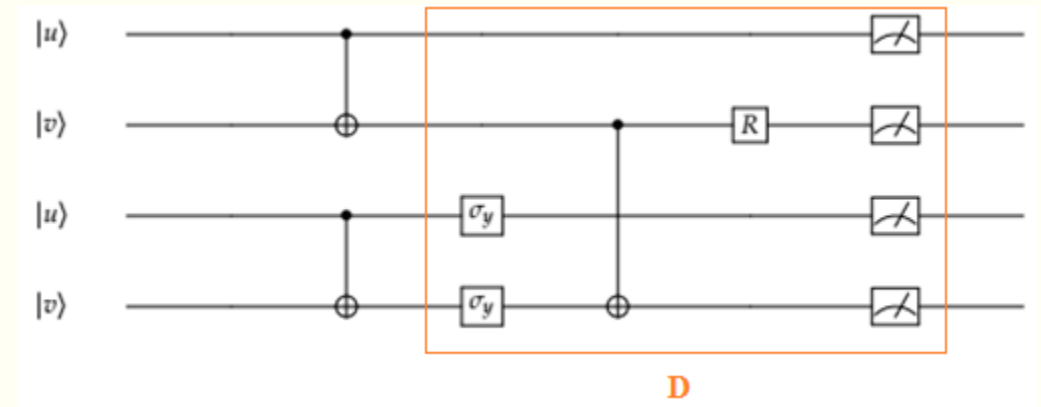


IMPLEMENTATION OF ENTANGLEMENT MEASURE

Application: Implementation of Entanglement Measure

- The M_z is a unitary operator that applies the $CNOT$ gate between the qubits $|u\rangle$ and $|v\rangle$ followed by measuring the degree of entanglement in between through the operator D .
- The state of the qubit $|u\rangle$ is arbitrary.
- The state of the qubit $|v\rangle$ is initialize in the vacuum state $|0\rangle$.
- For the state $|uv\rangle = \alpha|00\rangle + \delta|11\rangle$, the concurrence measure is as follows:

$$C = 2|\alpha\delta|$$



Algorithm

- The *CNOT* gate is applied on each replica of the two-qubit systems $|u\rangle$ and $|v\rangle$.

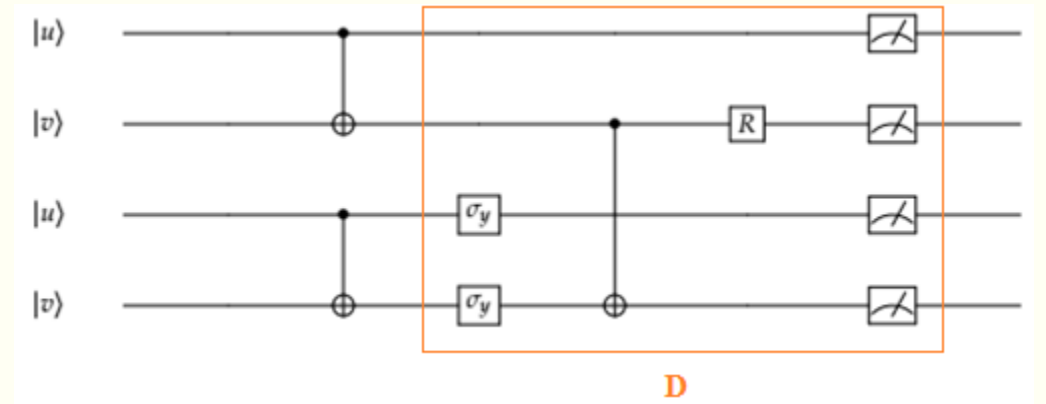
$$|\eta_0\rangle = CNOT_{1,2} CNOT_{3,4} |uv\rangle \otimes |uv\rangle$$

- The Pauli gate y is applied to the third and fourth qubits.

$$|\eta_1\rangle = (\sigma_{y_3} \otimes \sigma_{y_4}) |\eta_0\rangle$$

- CNOT* gate is applied between the second and the fourth qubits, respectively, followed by the rotation R gate as follows:

$$|\eta_2\rangle = (R_2 \otimes CNOT_{2,4}) |\eta_1\rangle$$



$$R|0\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}, \quad R|1\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}.$$

Example

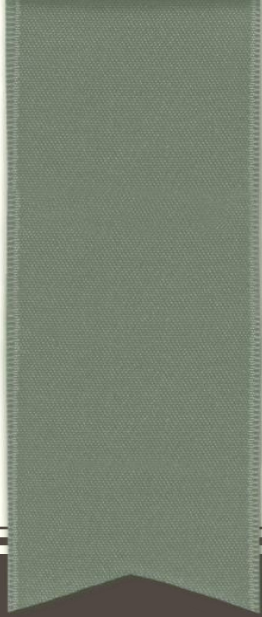
- Initial state u is $|u\rangle = \alpha|0\rangle + \delta|1\rangle$
- The state of the system after applying M_Z operator is as follows:

$$|\eta_2\rangle = \frac{1}{\sqrt{2}}\{-\alpha\delta|0000\rangle + \alpha\delta|0100\rangle - \alpha^2|0011\rangle - \alpha\delta|1010\rangle + \alpha^2|0111\rangle - \delta^2|1001\rangle - \delta^2|1101\rangle - \alpha\delta|1110\rangle\}$$

- Given that $C = 2|\alpha\delta|$, the relationship between the concurrence and the success probability for obtaining the states $|0000\rangle$, $|0100\rangle$, $|1010\rangle$, $|1110\rangle$ is:

$$C = 2\sqrt{2P_{0000}}, C = 2\sqrt{2P_{0100}}, C = 2\sqrt{2P_{1010}} \quad \text{or} \quad C = 2\sqrt{2P_{1110}}.$$

- In general, the measurement outputs are 0000, 0100, 0011, 1010, 0111, 1001, 1101, 1110. Now, if we see the four states 0000, 0100, 1010, 1110, then the desired state is entangled. Otherwise, nothing can be said about the entanglement or separability of the state.



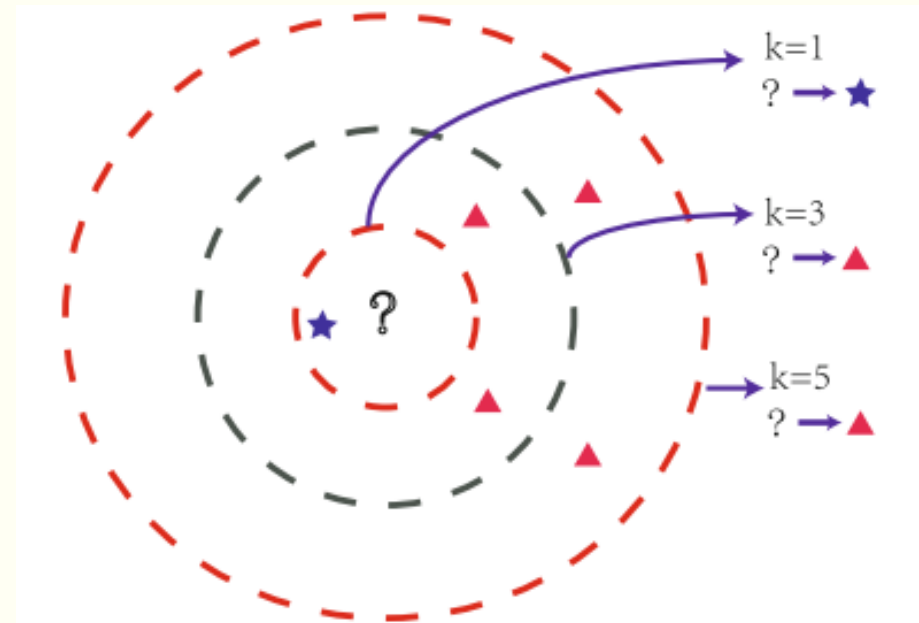
QQ CASE



K-NEAREST NEIGHBORS (KNN)

K -Nearest Neighbors (KNN) Algorithm

- Supervised ML algorithm
- Classification Based on measure of 'similarity'
- Computer is trained with a set of train states whose class labels are known.
- The test state with the unknown label is compared with the train states.
- k number of the nearest neighbors from the train states are identified for the given test state.
- The label of the test state is determined upon majority voting.



Classical & Quantum kNN Algorithm

Classical kNN

- **Aim:** Determine the distance between the test state and all the train states.
- Each state (train or test) is represented by a vector of complex numbers.
- Classical kNN algorithm has a complexity of $O(MN)$, with
 - N : dimension of vector to be classified
 - M : cardinality of set of train vectors

Quantum kNN

- Quantum k-nearest neighbor (QKNN), is a quantum analog of classical kNN algorithm.
- Distance between the test state and all the train states are simultaneously calculated by using:
 - superposition properties of the quantum states
 - collapse of the wavefunction upon measurement
- Swap test is used to calculate the fidelity simultaneously between the test state and all the train states:
 - This makes Quantum KNN algorithms much faster than its classical counterpart.

Classical kNN Algorithm

Steps

- For each test state (\mathbf{u}) (whose label is to be determined), compute its distance to the train states (\mathbf{v}) whose labels are known.
- Choose the k number of neighbors which are nearest to the test point.
- Conduct a majority voting and assign the label of the majority to the test point.

Different Types of Distance Measures

- **Euclidean distance:**

$$d(\mathbf{u}, \mathbf{v}) = \left(\sum_i^r |u_i - v_i|^2 \right)^{1/2}$$

- **Cosine similarity:**

$$(\mathbf{u}, \mathbf{v}) = \frac{\sum_i^r u_i^* v_i}{\sqrt{\sum_i^r u_i^2} \sqrt{\sum_i^r v_i^2}}$$

- **Fidelity:**

$$F(\mathbf{u}, \mathbf{v}) = |(\mathbf{u}, \mathbf{v})|^2 = |\langle \mathbf{u} | \mathbf{v} \rangle|^2$$

Limitations of Classical k NN

- As the number of train data points and the dimension of the state vectors grows, k NN can quickly turn intractable for classical computers.
- Classification of an N dimensional test state by comparing with M train states requires $O(MN)$ multiplication operations.
- Finding the nearest neighbors will require sorting of M number of distance which requires $O(M \log M)$ operations.
- The choice of the number k is also highly debated. There is no general way of choosing k and usually, hyperparameter tuning is done to choose the best possible k .

Swap Test

The swap test is a quantum algorithm that can be used to statistically estimate the fidelity of two pure states $|\psi\rangle$ and $|\phi\rangle$, i.e., $|\langle\psi|\phi\rangle|^2$.

1. Three registers prepared in states $|0\rangle, |\psi\rangle, |\phi\rangle$ are needed to implement the swap test. The initial combined state of the three registers is: $|R\rangle = |0\rangle \otimes |\psi\rangle \otimes |\phi\rangle$
2. Next we apply a Hadamard operation H on the first register followed by a control swap C_S on the other two registers where the first register serves as the control system.

effect of C_S :

$$C_S |0\rangle |a\rangle |b\rangle = |0\rangle |a\rangle |b\rangle,$$
$$C_S |1\rangle |a\rangle |b\rangle = |1\rangle |b\rangle |a\rangle.$$

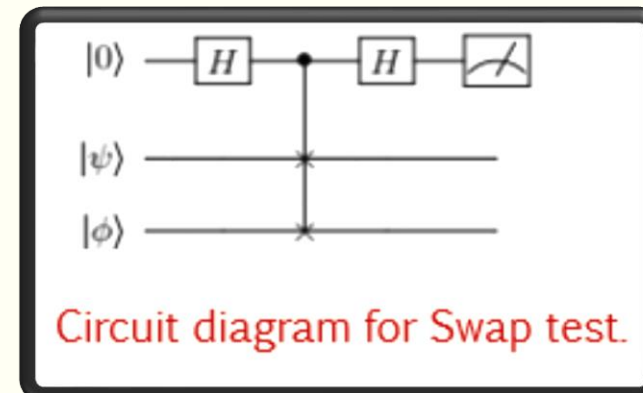
Result after two operations

$$|\bar{R}\rangle = \frac{1}{\sqrt{2}}(|0\rangle |\psi\rangle |\phi\rangle + |1\rangle |\phi\rangle |\psi\rangle)$$

3. Applying another Hadamard operation H on the first qubit followed by a measurement on the first qubit in the $\{|0\rangle, |1\rangle\}$ results in 0 and 1 with probabilities:

$$P(0) = \frac{1}{2} + \frac{1}{2}|\langle\psi|\phi\rangle|^2,$$
$$P(1) = \frac{1}{2} - \frac{1}{2}|\langle\psi|\phi\rangle|^2.$$

4. The quantity $P(0) - P(1)$ gives us the desired fidelity.



QKNN for Classifying Bipartite Entangled States

$|\psi\rangle$: n-qubit test state without label
 $\{|\phi_i\rangle\}$: train states with label i

regstrs: $\left\{ \begin{array}{l} r_1 : \text{single-qubit system} \\ r_2 \\ r_3 \end{array} \right\}$: n-qubit system
 r_4 : m-qubit system which 2^m is cardinality of the set $\{|\phi_i\rangle\}$

Step 1 : Initialization

· initialize the registers in the required state vectors $|R\rangle$

Step 2: State transformation

· transforming the initial state to arrive at the state $|\bar{R}\rangle$, which is suitable for fidelity estimation.

Step 3: Measurements

· performing measurements to estimate the fidelity.

Initialization

In this step of the algorithm, we prepare the four registers in a suitable state:

- r_1 is prepared in $|0\rangle$
- r_2 is prepared in the test state $|\psi\rangle$
- r_3 is prepared in the state $|0\rangle^{\otimes n}$ which $n = \log N$
- r_4 is prepared in the state $|0\rangle^{\otimes m}$ which $m = \log M$
- The initial state of the total system is: $|R\rangle = |0\rangle|\psi\rangle|0\rangle^{\otimes n}|0\rangle^{\otimes m}$

State Transformation

In the second step of the algorithm, we apply a set of quantum operations that are independent of the given test state.

- Apply a Hadamard gate H to the first register r_1 and $H^{\otimes m}$ to the r_4 :

$$|R'\rangle = \frac{1}{\sqrt{2M}} \sum_{i=1}^M (|0\rangle + |1\rangle) |\psi\rangle |0\rangle^{\otimes n} |i\rangle$$

- Apply a quantum oracle W of the form $W|0\rangle|i\rangle = |\phi_i\rangle|i\rangle$ to the third and fourth registers.

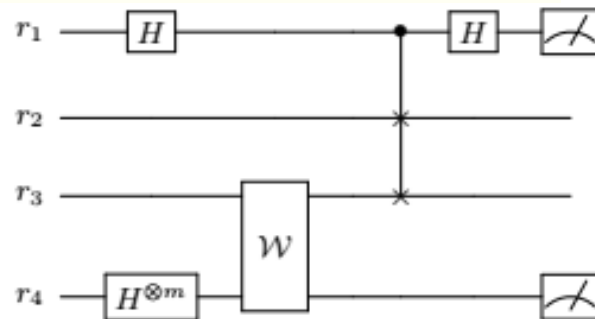
$$|R''\rangle = \frac{1}{\sqrt{M}} \sum_{i=1}^M |0\rangle |\psi\rangle |\phi_i\rangle |i\rangle$$

- Implement a control swap $CS(6)$ with r_1 as the control qubit and r_2 and r_3 as the target registers:

$$|R'''\rangle = \frac{1}{\sqrt{2M}} \sum_{i=1}^M (|0\rangle |\psi\rangle |\phi_i\rangle + |1\rangle |\phi_i\rangle |\psi\rangle) |i\rangle$$

- Apply the Hadamard operation on the r_1 register.

$$|\bar{R}\rangle = \frac{1}{2\sqrt{M}} \sum_{i=1}^M (|0\rangle [|\psi\rangle |\phi_i\rangle + |\phi_i\rangle |\psi\rangle] + |1\rangle [|\psi\rangle |\phi_i\rangle - |\phi_i\rangle |\psi\rangle]) |i\rangle$$



Measurements

In the final step we perform measurements on the four registers in the following order.

- First a measurement is performed on register r_i in basis $\{|0\rangle, |1\rangle\}$ resulting in 0 and 1 outcomes with probabilities:

$$p(0) = \frac{1}{2} + \frac{1}{2M} \sum_{i=1}^M |\langle \psi | \phi_i \rangle|^2,$$

$$p(1) = \frac{1}{2} - \frac{1}{2M} \sum_{i=1}^M |\langle \psi | \phi_i \rangle|^2.$$

- the state of the other three registers after the measurement is

$$|R_0\rangle = \frac{1}{\sqrt{2}} \frac{\sum_{i=1}^M (|\psi\rangle |\phi_i\rangle + |\phi_i\rangle |\psi\rangle) |i\rangle}{\sqrt{M + \sum_{j=1}^M |\langle \phi_j | \psi \rangle|^2}},$$

$$|R_1\rangle = \frac{1}{\sqrt{2}} \frac{\sum_{i=1}^M (|\psi\rangle |\phi_i\rangle - |\phi_i\rangle |\psi\rangle) |i\rangle}{\sqrt{M - \sum_{j=1}^M |\langle \phi_j | \psi \rangle|^2}}.$$

- Upon measurement on r_4 in the basis $\{|i\rangle\}$, the probability of the i -th outcome is:

$$p_0(i) = \frac{1 + |\langle \phi_i | \psi \rangle|^2}{M + \sum_{j=1}^M |\langle \phi_j | \psi \rangle|^2},$$

$$p_1(i) = \frac{1 - |\langle \phi_i | \psi \rangle|^2}{M - \sum_{j=1}^M |\langle \phi_j | \psi \rangle|^2}.$$

- Define:

$$\begin{aligned} q(i) &= p_0(i) - p_1(i) \\ &= \frac{1 + F_i}{M + \sum_{j=1}^M F_j} - \frac{1 - F_i}{M - \sum_{j=1}^M F_j} \\ &= \frac{2(F_i - \langle F \rangle)}{M(1 - \langle F \rangle^2)}. \end{aligned}$$

which $\langle F \rangle = \sum_{j=1}^M F_j / M$

Important Notes

- The quantity $q(i)$ is directly proportional to the desired fidelity and is the quantity of interest in QKNN algorithm.
- We need to initialize the system in the state $|R\rangle$ and transform it into the state $|\bar{R}\rangle$ and perform the measurement for a sufficiently large number of times.
- In each run of the algorithm, we acquire a click in the register r_1 and a click in the register r_4 .
- The larger values of the fidelity yields larger contrast $q(i)$; hence, running the QKNN algorithm a sufficient number of times, we can find the k states which are closest to $|\psi\rangle$, i.e., the k number of indices having highest $q(i)$.

Advantages of QKNN Over Its Classical Counterpart.

- It offers the capability to classify unknown states. This is advantageous when we deal with quantum data as we get to bypass the expensive process of quantum state tomography. Any classical kNN method will require the complete description of the quantum state.
- In classical kNN methods, one requires to compute the distance of the test state with every train state, even far off states, to obtain the k nearest neighbors. In our QKNN algorithm, through quantum parallelism and the probabilistic nature of quantum measurement, only those train states which have high Fidelity with the train states will have high probability of getting detected upon measurement. Therefore, in a limited number of trails only the states which are closer to the train state will appear in the measurement hence fewer resources are spent on them.

Entanglement Classification Using Classical kNN

- The results show that the classical kNN works perfectly for entanglement classification in two-qubit case. In the case of three-qubit case the accuracy we achieve is little over 82%. This accuracy can be increased by increasing the number of k and by increasing the size of the set of train states.

No. of Qubits	No. of classes	Entanglement classes	Accuracy	Class size	Algorithm type
2	2	Separable, Entangled	100%	10^5	Classical
2	2	Separable, Maximally entangled	100%	10^5	Classical
3	5	1-2-3, 12-3, 1-23, 13-2, 123	82.2%	10^5	Classical

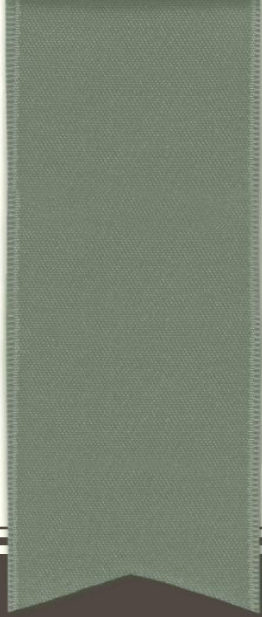
Entanglement classification using classical kNN classifier.

Entanglement classification using QKNN

- we simulate the QKNN algorithm and classify two-qubit states in two scenarios. First, when the classification is between separable states and maximally entangled states and next when the two classes are separable states and general entangled states.

No. of Qubits	No. of classes	Entanglement classes	Accuracy	Class size	Algorithm type
2	2	Separable, Maximally entangled	96.67%	16	Classical
2	2	Separable, Maximally entangled	95.67%	16	Quantum 10^4 shots
2	2	Separable, Entangled	80.1%	16	Classical
2	2	Separable, Entangled	80.67%	16	Quantum 10^4 shots

Entanglement classification using quantum kNN classifier compared with classical kNN classifier. Here, shots indicate the number of measurement shots performed over each quantum circuit simulation.



THANK YOU FOR YOUR ATTENTION!
QUESTIONS?
