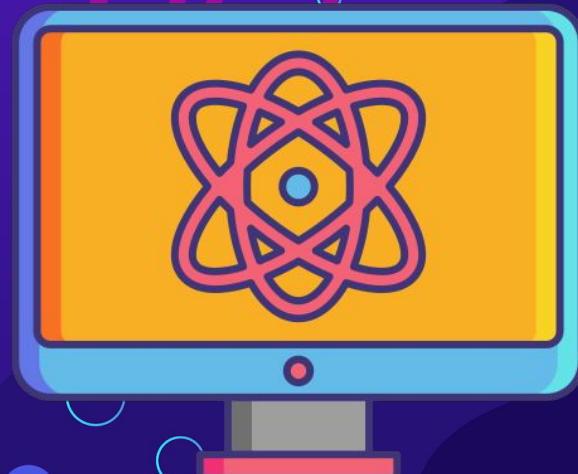


Quantum Machine Learning and Encoding

NEMO Lab Fall 2024: Ben Gerdes, Caleb Holmbeck, Andrew Maas, Ally Muellner, Reva Long, Hannah Pearce, Bishop Placke, and Thy Tran



Quantum Intro

General State of a qubit, Bra-ket notation, Kronecker Product- Ben Gerdes

Common orthonormal bases and Bell state- Ally

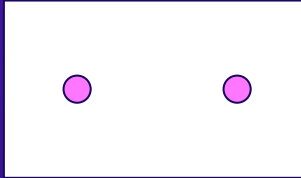
Quantum gates and circuits- Caleb

Measurement (including the measurement strat for bell state basis for 2 qubits)- Bishop Placke

An Introduction to Qubits

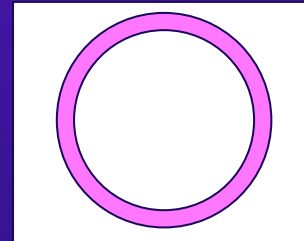
Normal Bits:

- Represented as 0 or 1
- Visually, can be represented by two dots



Qubits (Quantum bits)

- Represented by a unit vector with two entries
- Visually, can be represented by a unit circle*



$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

*For real entries. If we include complex entries, the qubit is a sphere called the Bloch sphere.

Bra-ket Notation

To simplify the computations that happen most frequently in quantum computing, we use a different kind of notation for quantum states.

Ket: $|\psi\rangle$ always denotes a column vector, e.g.

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_d \end{bmatrix}$$

Convention:

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Bra: $\langle\psi|$ always denotes a row vector that is the conjugate transpose of $|\psi\rangle$, e.g. $[\alpha_1^* \ \alpha_2^* \ \dots \ \alpha_d^*]$

Bracket: $\langle\phi|\psi\rangle$ denotes $\langle\phi|\cdot|\psi\rangle$, the inner product of $|\phi\rangle$ and $|\psi\rangle$

The Kronecker Product

To get the product of two quantum states, we use the Kronecker product, denoted by the symbol \otimes . If A is an $m \times n$ matrix and B is a $p \times q$ matrix, the Kronecker product $A \otimes B$ will be an $mp \times nq$ matrix such that each entry in A is multiplied by the entire matrix B.

For Example:

$$\begin{array}{c} A \\ \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \end{array} \otimes \begin{array}{c} B \\ \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix} \end{array} = \begin{bmatrix} 1 \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix} & 2 \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix} \\ 3 \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix} & 4 \begin{bmatrix} 0 & 5 \\ 6 & 7 \end{bmatrix} \end{bmatrix} = \left[\begin{array}{cc|cc} 1 \times 0 & 1 \times 5 & 2 \times 0 & 2 \times 5 \\ 1 \times 6 & 1 \times 7 & 2 \times 6 & 2 \times 7 \\ \hline 3 \times 0 & 3 \times 5 & 4 \times 0 & 4 \times 5 \\ 3 \times 6 & 3 \times 7 & 4 \times 6 & 4 \times 7 \end{array} \right] = \left[\begin{array}{cc|cc} 0 & 5 & 0 & 10 \\ 6 & 7 & 12 & 14 \\ \hline 0 & 15 & 0 & 20 \\ 18 & 21 & 24 & 28 \end{array} \right].$$

Common Orthonormal Bases

A set of vectors is orthonormal if:

1. They are pairwise orthogonal
2. vector u base i is a unit vector for $1 \leq i \leq n$

Two of the most relevant to quantum computing are the Standard basis: $\{|0\rangle, |1\rangle\}$

1-qubit forms a vector of 2 components, 2 qubits form a vector of 4 components. This 2^n pattern continues with every qubit added.

Hadamard basis: $\{|+\rangle, |-\rangle\}$

$$|+\rangle = (|0\rangle + |1\rangle) / \sqrt{2}$$

$$|-\rangle = (|0\rangle - |1\rangle) / \sqrt{2}$$

This basis is important because it establishes equal superposition

$ 0\rangle$	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	$\frac{1}{\sqrt{2}}(0\rangle + 1\rangle)$
$ 1\rangle$		$\frac{1}{\sqrt{2}}(0\rangle - 1\rangle)$
INPUT		OUTPUT

1-qubit examples

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

2-qubit example

$$|11\rangle = \left(\begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Bell State

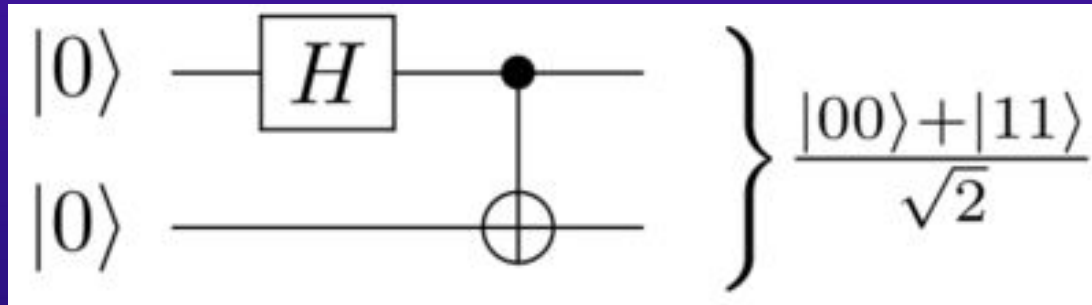
By using a Hadamard gate, we achieve equal superposition

Equal superposition is when there is an equal chance of measuring 0 or 1










Using CNOT gate, the two qubits are entangled

When qubits are entangled, quantum computers are able to manipulate all entangled qubits in one operation

Bell states are useful in many areas of quantum computing, such as quantum teleportation



Quantum Gates and Circuits

Operator	Gate(s)		Matrix
Pauli-X (X)			$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
Pauli-Y (Y)			$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$
Pauli-Z (Z)			$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Hadamard (H)			$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
Phase (S, P)			$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
Controlled Not (CNOT, CX)			$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
SWAP			$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Rotates vectors on bloch sphere by 180° over respective axis.

Also called Z90 gate since it represents a rotation of 90° over the z-axis

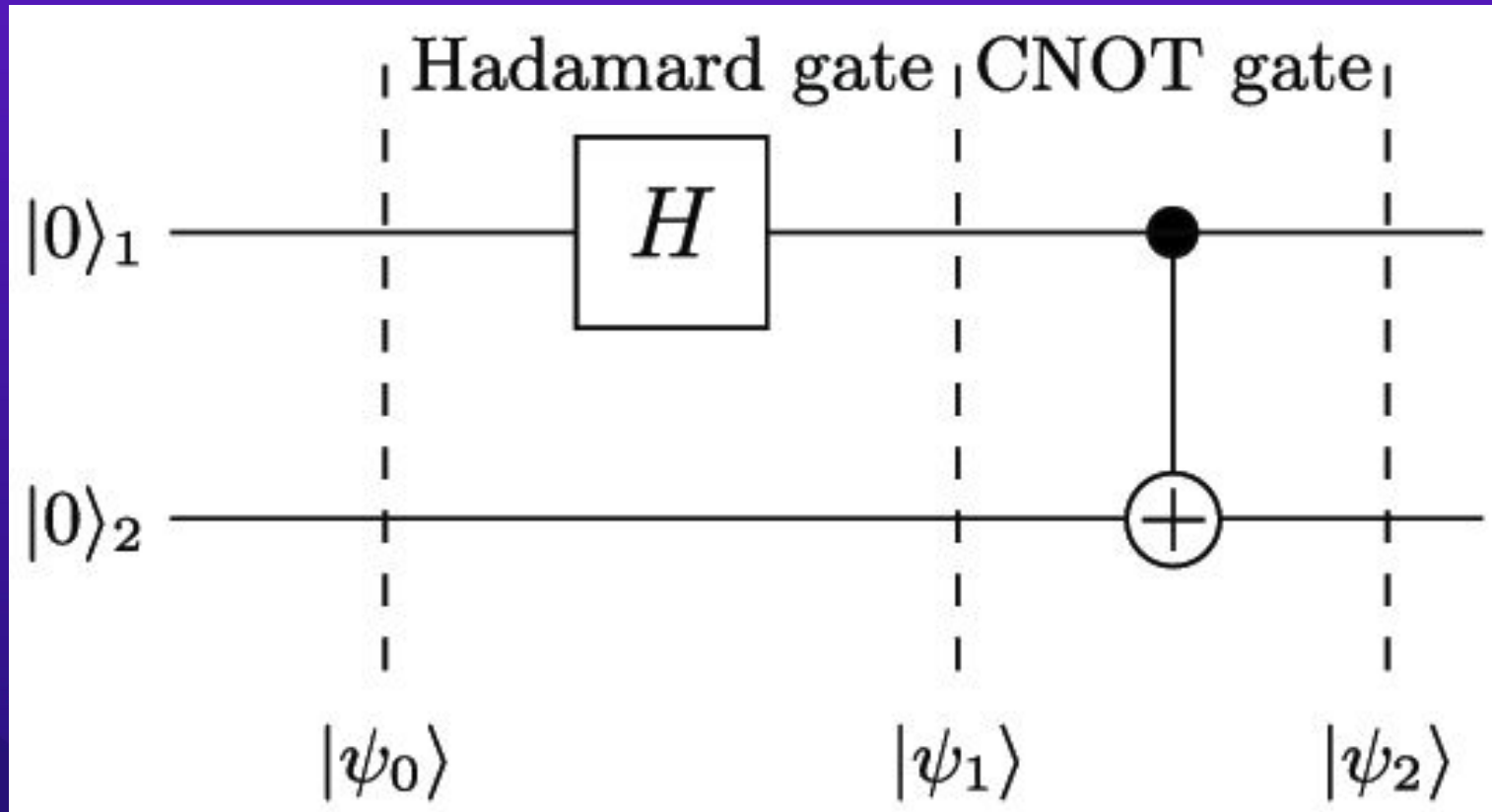
Used to swap the states of two qubits.

Pauli-X is similar to a NOT gate.

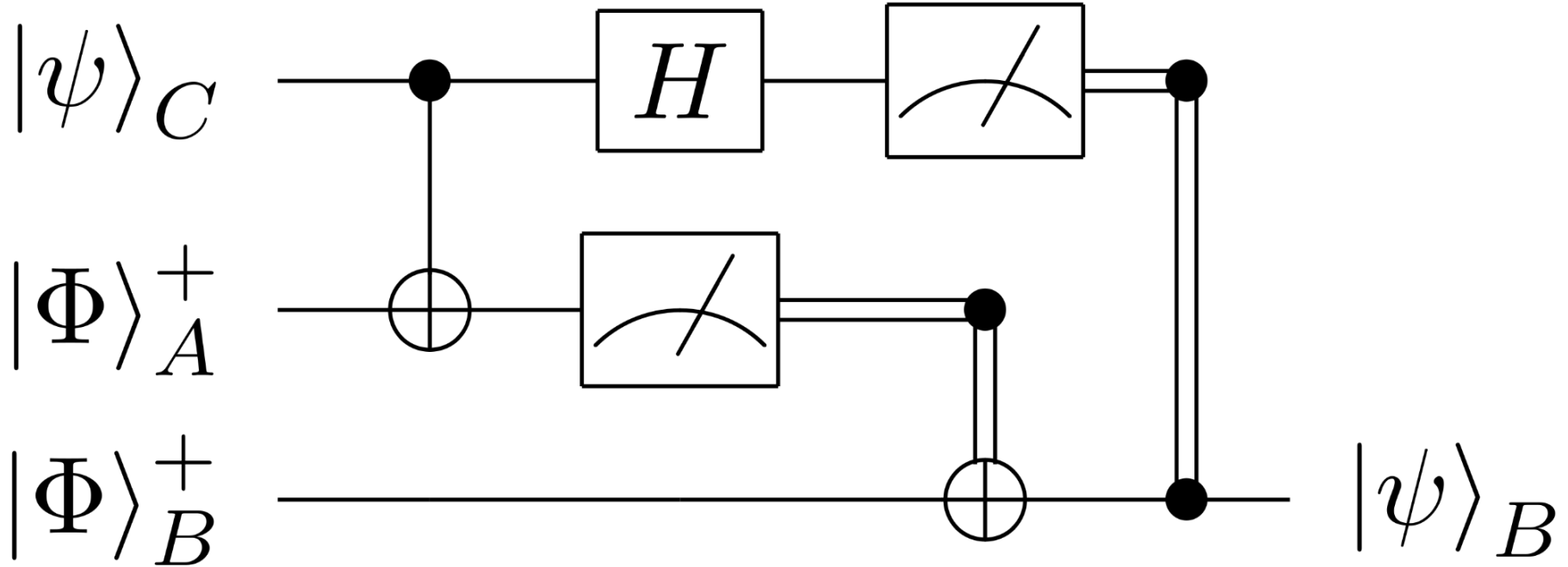
Used in making Bell states/entangled states.
Creates equal superpositions (existing in multiple states).

Used to entangle states.
Flips the values of the target qubit. (similar to R-XOR)

Circuit Diagram Example of Bell State β_{00}



Quantum Teleportation Circuit

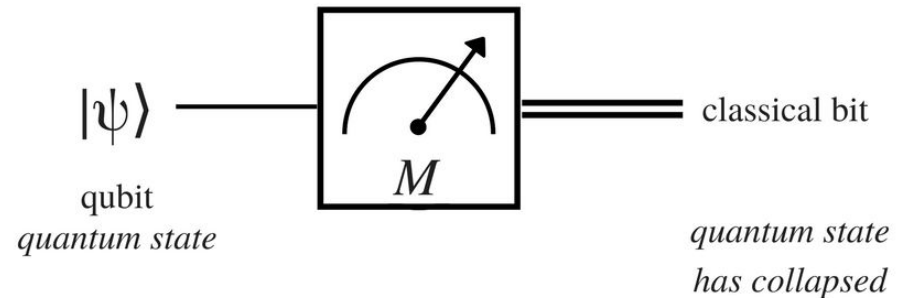


Measurement

Measuring takes a qubit in a state $\psi = \alpha|0\rangle + \beta|1\rangle$ and returns the states $|0\rangle$ or $|1\rangle$ with probabilities $|\alpha|^2$ and $|\beta|^2$ respectively.

We take $|0\rangle$ to mean the qubit is “off” or the classical bit 0, and $|1\rangle$ to mean “on” or the classical bit 1.

Measurement can also be done with respect to some other basis.



Measuring ψ with respect to a Non-Standard Basis

Sometimes, it is useful to measure a qubit in some other basis.

Hadamard Basis

$$\psi = 1/\sqrt{2}((\alpha + \beta)|+\rangle + (\alpha - \beta)|-\rangle)$$

If we measure and get +, we consider the qubit to be off (or a 0 classically). Likewise, measuring a - tells us the qubit is on (or a 1 classically)

Bell Basis

$$\psi = \alpha_1 |\beta_{00}\rangle + \alpha_2 |\beta_{01}\rangle + \alpha_3 |\beta_{10}\rangle + \alpha_4 |\beta_{11}\rangle$$

When we measure, we get β_{ab} , then we consider our two bits to be: ab

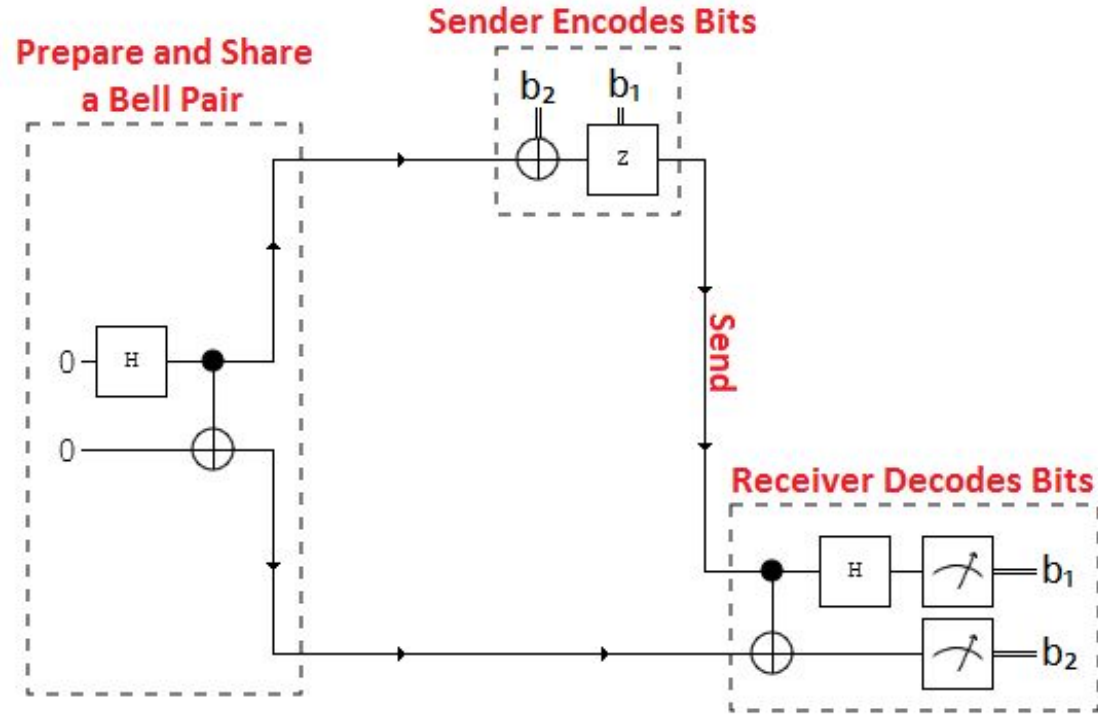
Guaranteed Measurements

Intended Message	Applied Gate	Resulting State ($\cdot \sqrt{2}$)
00	I	$ 00\rangle + 11\rangle$
10	X	$ 01\rangle + 10\rangle$
01	Z	$ 00\rangle - 11\rangle$
11	ZX	$- 01\rangle + 10\rangle$

Will receive the state

$$\psi = |\beta_{ab}\rangle$$

Measuring with Bell Basis then gives the bits ab with probability 1



Entanglement and Quantum Teleportation

A multi-qubit state is said to be entangled if it cannot be written as the Kronecker product of two other states.

Ex. Bell states are entangled states of 2 qubits

Say Alice and Bob share the Bell State,

$$|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$$

And Alice has an additional qubit in an unknown state,

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Using the gate shown earlier (CNOT on two of the qubits, followed by a Hadamard Gate on one, and then Measurement), we will transport the state $|\psi\rangle$ to Bob.

Output of Quantum Teleportation Circuit

$$\frac{1}{2}(\alpha \begin{bmatrix} |00\rangle \\ |10\rangle \\ |01\rangle \\ |11\rangle \end{bmatrix} + \beta \begin{bmatrix} |01\rangle \\ -|11\rangle \\ |00\rangle \\ -|10\rangle \end{bmatrix})$$

Alice may now measure her qubits. Below, we show Alice's measured states on the left and Bob's qubit's state resulting from the measurement.

$$|00\rangle \Rightarrow \alpha|0\rangle + \beta|1\rangle$$

$$|01\rangle \Rightarrow \alpha|1\rangle + \beta|0\rangle$$

$$|10\rangle \Rightarrow \alpha|0\rangle - \beta|1\rangle$$

$$|11\rangle \Rightarrow \alpha|1\rangle - \beta|0\rangle$$

Alice would then send the **classical** information of what she measured to Bob, which tells him what state his qubit is now in. He can then use gates to recover $|\psi\rangle$.

Machine Learning - Reva, Thy, Hannah

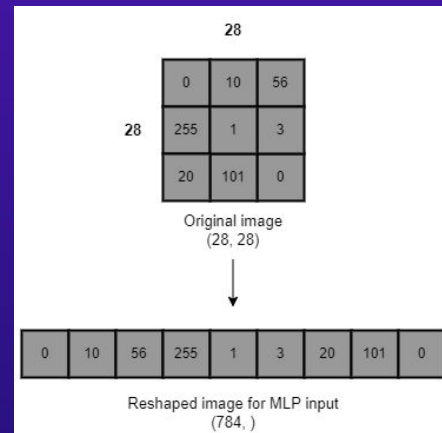
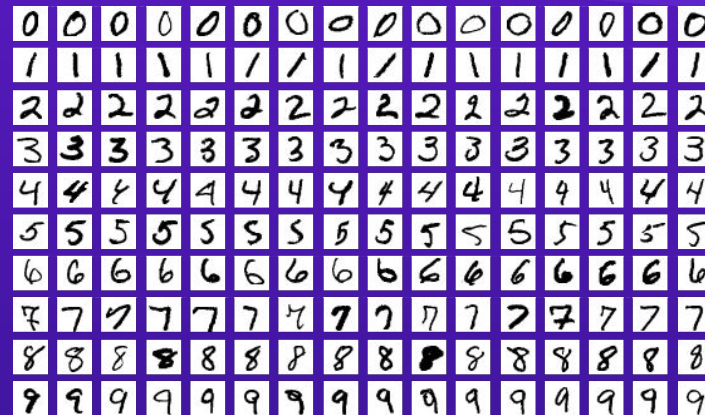
- Subset of AI that uses mathematical models and algorithms to analyse, identify patterns, and make predictions on data
- Process:
 - Data collection: gather and preprocess raw data
 - Feature extraction: identify key characteristics or patterns in the data
 - Model training: use algorithms (ex: neural networks) to learn patterns from training data
 - Evaluation: test the model to measure accuracy and performance
 - Prediction: apply the trained model to make decisions or predictions on new data
- Applications:
 - **Image Recognition:** Identifying handwritten digits (MNIST dataset).
 - **Personalized Recommendations:** Suggesting content on platforms like Netflix and Spotify.
 - **Language Processing:** Enabling chatbots to understand and respond to queries.

What is quantum machine learning?

- **Emerging field:** Researchers are actively developing methods to utilize the potential of quantum computing in machine learning.
- **Main approaches**
 - Hybrid models: combine classical machine learning with quantum computing
 - Purely quantum models: use quantum circuits to perform computations directly on quantum computers
- **Future potential:**
 - Quantum machine learning is not intended to replace classical machine learning but shows promise for solving problems involving complex, nonlinear data, such as weather where randomness is involved
 - Still in early stages as QML algorithms are limited by the inefficiency of quantum encoding techniques and limited qubits

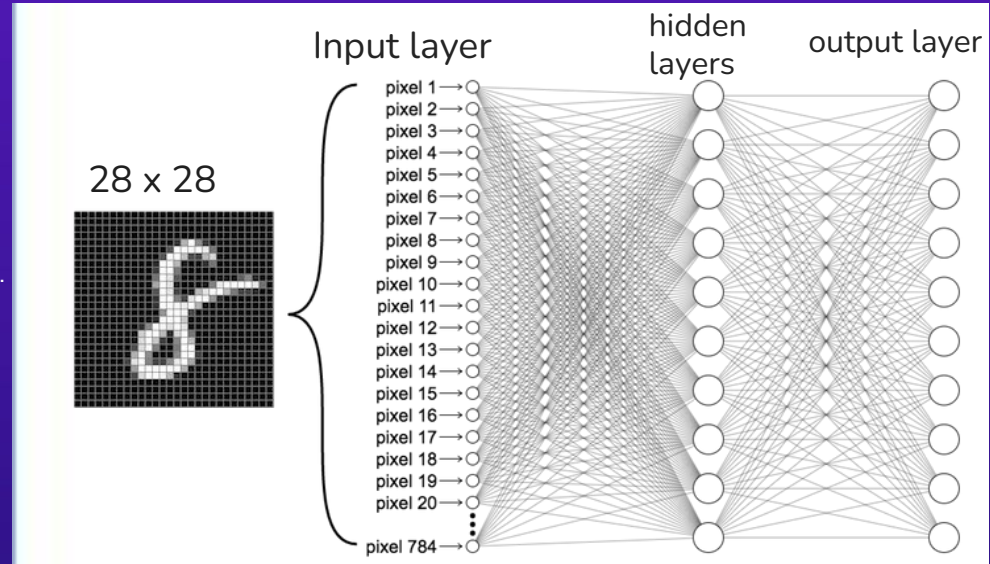
MNIST Dataset

- MNIST is one of the most popular datasets in machine learning, it is good for evaluating models as it captures different handwriting styles.
- Contains **70,000 images**: 60,000 for training and 10,000 for testing.
- Each image is **28x28 pixels**, grayscale, representing digits from **0 to 9**.
- Grayscale pixel values range from **0 (black)** to **255 (white)**.
- For machine learning, each 2D array is flattened into a 1D **784 pixel array** ($28 \times 28 = 784$).
- extensions of MNIST, such as Fashion MNIST (fashion articles) and EMNIST (letters and digits), which offer more complex and varied data.



ANN with MNIST example

- ANN architecture:
 - **Input Layer:**
 - $28 \times 28 = 784$ input neurons, each representing a pixel.
 - Pixel values (grayscale between 0 and 255) are normalized to the range $[0, 1]$.
 - **Hidden Layers:**
 - One or more layers with activation functions to capture patterns and features in the input data.
 - **Output Layer:**
 - 10 output neurons, each corresponding to a digit (0–9).
 - Each neuron outputs a probability indicating how likely the input corresponds to that digit.
 - Classical neural network
 - Achieves >98% accuracy by effectively learning from the MNIST training dataset.
 - Quantum neural network
 - Currently achieves ~89% accuracy with the MNIST dataset, showcasing potential but limited by current quantum hardware and encoding.



Artificial Neural Network for Classical Computer

E.g.

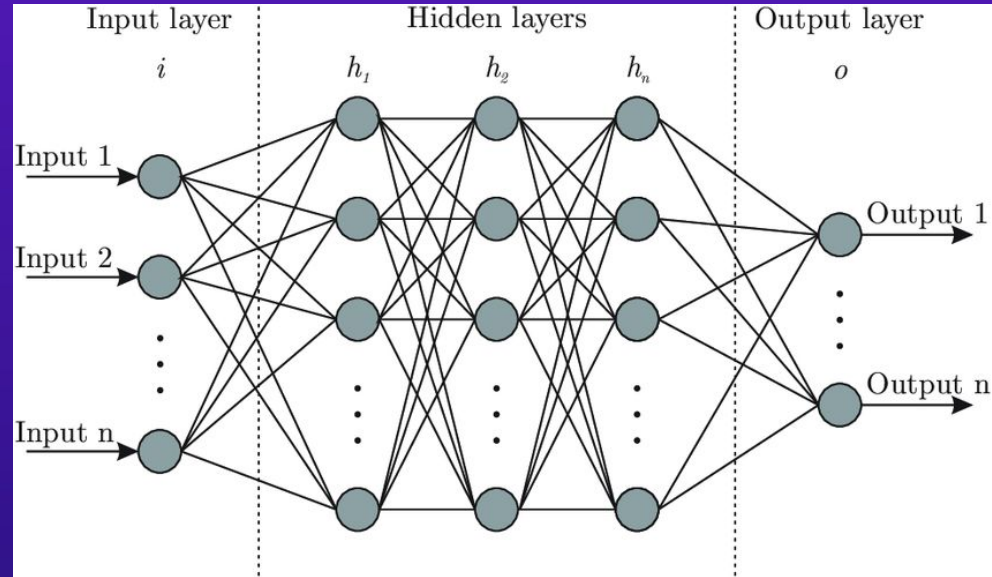
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \rightarrow \begin{bmatrix} 1 \\ 4 \\ 2 \\ 5 \\ 3 \\ 6 \end{bmatrix}$$

Input layer:

using $28 \times 28 = 784$ pixels and convert it into a 784×1 matrix as example.

Hidden layers: we can apply different function to increase weight or turn neurons on and off. Some common functions are: Sigmoid function, ReLu function, softmax function,....

Output layer: a 10×1 matrix with probability of 10 digits. The answer will be the digit with the highest probability.



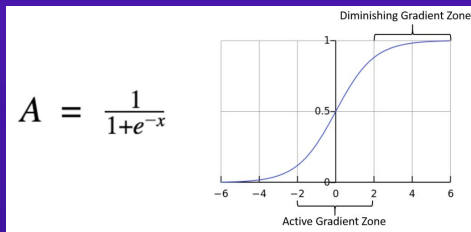
Artificial Neural Network for Classical Computer

To convert one layer to another, we apply some functions where:

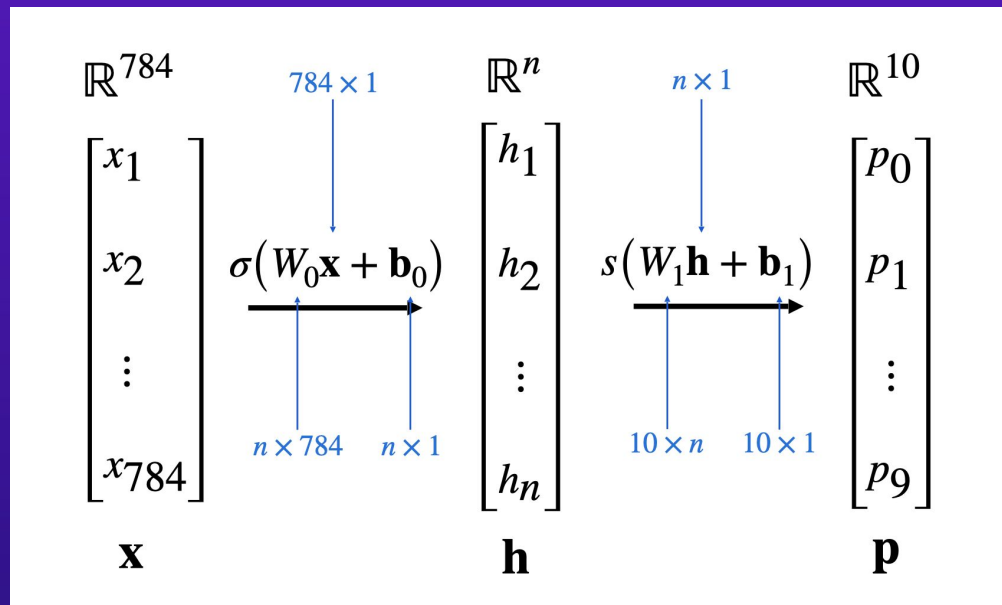
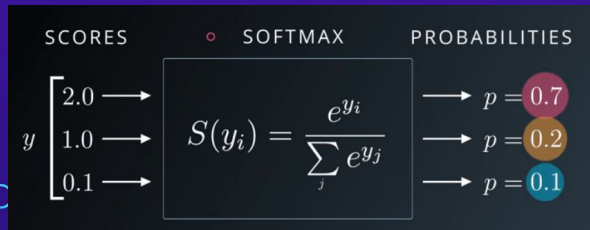
W: weight

b: bias

Sigmoid function:



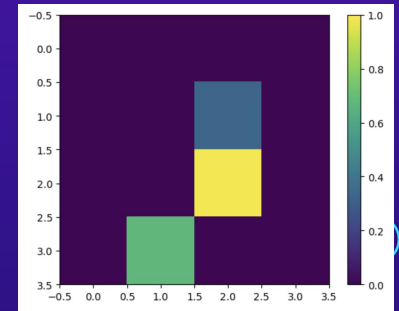
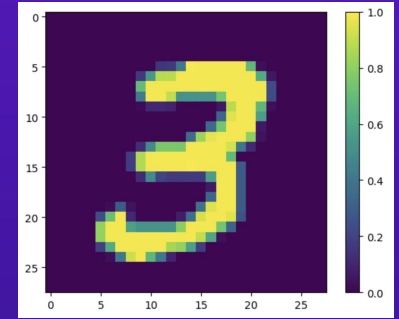
Softmax function:



Artificial Neural Network for Quantum Computer

Problem: Classify between digits 3 and 6

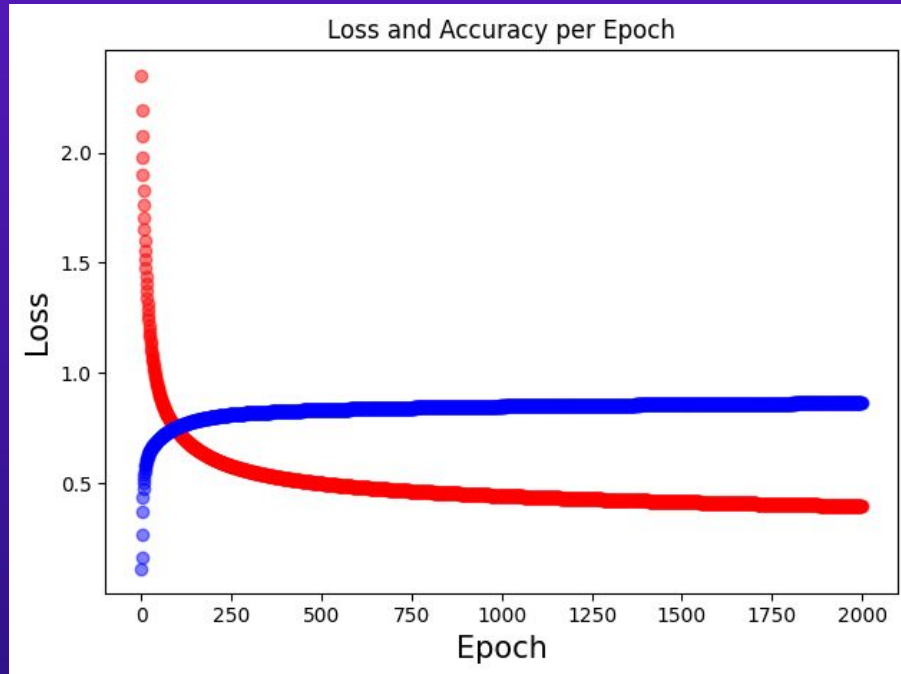
- The data came from MNIST dataset mention above but just keep 3's and 6's
 - Training samples = 12,049
 - Test samples = 1,968
 - Each sample is in 28x28 size
- Down-size to 4x4 (Quantum computers are limited to the number of qubits)
- Get rid of contradictory examples (remove images that belong to both classes)
- Convert: pixel -> binary -> qubits
- Build a quantum circuit to process the data



Classical Models

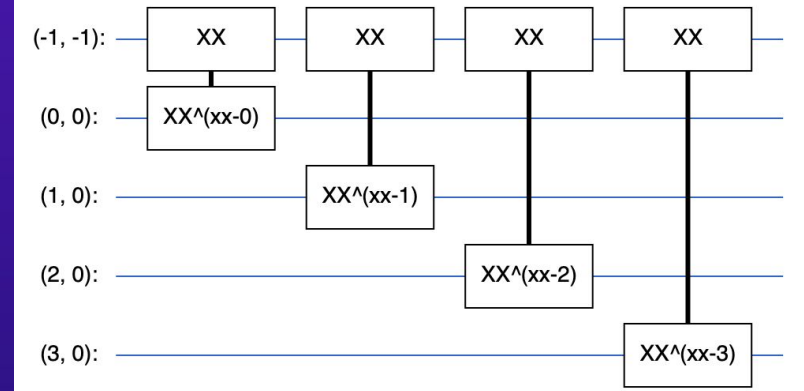
Accuracy = blue

Loss = red



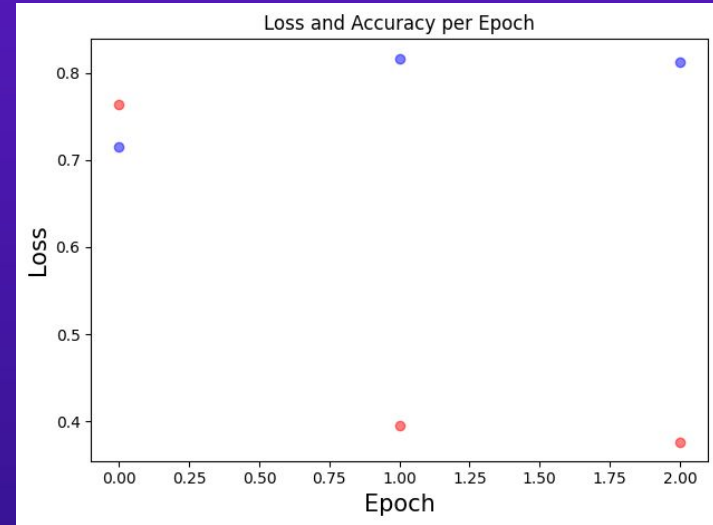
Quantum Neural Network

- Encode data
 - 2 qubit gates with a readout qubit
 - Each layer uses n instances of the same gate with each of the data qubits acting on the readout qubit.
- Train model
 - Parameterized Quantum Circuit
 - Threshold
- Classify images
 - Expectation of the readout qubit

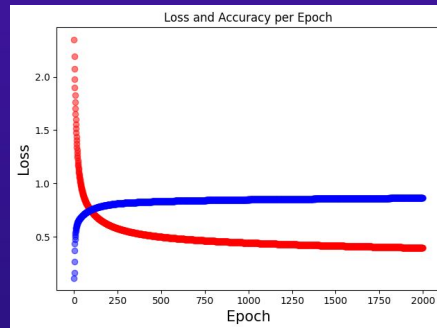


Quantum Model

Epoch	1	2	3
Loss	0.7642	0.3955	0.3764
Accuracy	0.7154	0.8170	0.8130



Classical Example



Classical vs Quantum

Classical:

- # of epochs = 1
- Time: 1 minute
- Accuracy = 0.9816
- Loss = 0.0429

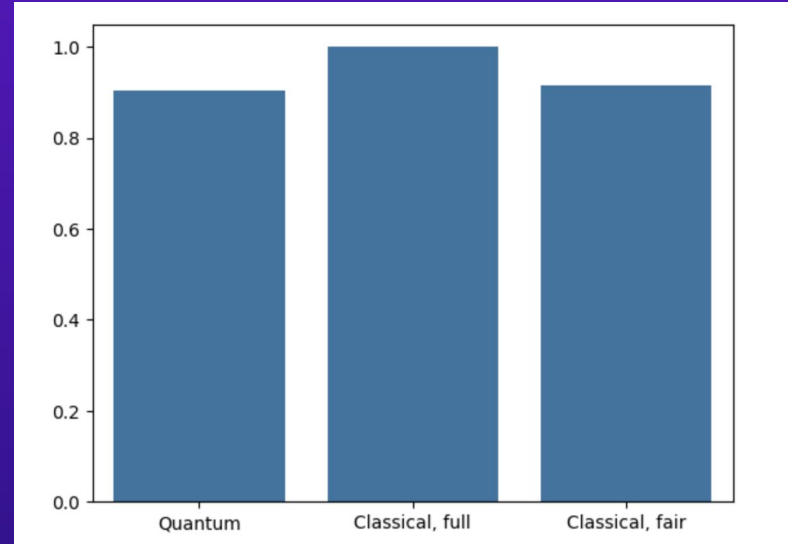
Quantum:

- # of epochs = 3
- Time: 45 minutes
- Accuracy = 0.8992
- Loss = 0.3464

Classical vs Quantum

A basic classical neural network
outperforms the quantum neural network!

What is the future of quantum machine
learning?



Basis Encoding

1. Convert number into binary
2. Rewrite as a quantum state

- Advantages:
 - Easy and intuitive
- Disadvantages
 - Can require LOTS of qubits
- Best for discrete data
 - Integers and binary



$$\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Rotation Encoding

- Rotates information in complex plane

$$56 \rightarrow \theta = 56^\circ$$

$$R_x(56^\circ)|0\rangle = \begin{bmatrix} 0.883 & -0.469i \\ -0.469i & 0.883 \end{bmatrix} |0\rangle = \begin{bmatrix} 0.883 \\ -0.469i \end{bmatrix}$$

- Rotating around different axes have different operations

56 with Basis Encoding: $|111000\rangle$

- Rotating around X-axis:

$$R_x(\theta) = \begin{bmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{bmatrix}$$

- Best for continuous data
 - Floating point

$$101 \rightarrow \theta_0 = 0, \theta_1 = \pi, \theta_2 = 0$$

$$R_{x2}(0) * R_{x1}(\pi) * R_{x0}(0) |0\rangle$$

Amplitude Encoding

1. Normalize each data point

$$X = [1.4, 2.6, 0.1, 1.2]$$

2. Take sum of x_i for each i^{th} term of data

$$(1.4)^2 + (2.6)^2 + (0.1)^2 + (1.2)^2 = 10.17$$

- Best for continuous data
 - Complex

$$\begin{aligned} X_{00} &= 1.4 / \sqrt{(10.17)} \\ X_{01} &= 2.6 / \sqrt{(10.17)} \\ X_{10} &= 0.1 / \sqrt{(10.17)} \\ X_{11} &= 1.2 / \sqrt{(10.17)} \end{aligned}$$

$$\begin{bmatrix} \frac{1.4 \sqrt{}}{(10.17)} \\ \frac{2.6 \sqrt{}}{(10.17)} \\ \frac{0.1 \sqrt{}}{(10.17)} \\ \frac{1.2 \sqrt{}}{(10.17)} \end{bmatrix}$$

$$\frac{1.4}{\sqrt{(10.17)}} |00\rangle + \frac{2.6}{\sqrt{(10.17)}} |01\rangle + \frac{0.1}{\sqrt{(10.17)}} |10\rangle + \frac{1.2}{\sqrt{(10.17)}} |11\rangle$$

Implications in Machine Learning

Method	Basis	Rotation	Amplitude
Pure Simulator	58.29%	94.65%	86.63%
Hardware, no Error Mitigation	55.61%	54.01%	52.94%
IBM_Torino, no Error Mitigation	52.68%	86.63%	86.63%
Hardware, Error Mitigation	57.21%	94.11%	83.95%
Hardware Simulator, no Error Mitigation	58.29%	94.11%	86.10%
Hardware Simulator, Error Mitigation	59.5%	95.19%	86.64%