LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

MNM TEAM

Focused Tutorial on Quantum Computing

Korbinian Staudacher staudacher@nm.ifi.lmu.de

Xiao-Ting Michelle To to@nm.ifi.lmu.de

MNM-Team, Ludwig-Maximilians-Universität München

June 16, 2023

MNM-Team


Technische Universität München
Leibniz-Rechenzentrum
der Bayerischen Akademie der Wissenschaften
Garching
München
der Bundeswehr
Universität München
Neubiberg

Outlook of today's tutorial

- First part: Introduction to mathematics of QC

Outlook of today's tutorial

- First part: Introduction to mathematics of QC
  - QuBit states

Outlook of today's tutorial

- First part: Introduction to mathematics of QC
  - QuBit states
  - State manipulation via unitary matrices

Outlook of today's tutorial

- First part: Introduction to mathematics of QC
  - QuBit states
  - State manipulation via unitary matrices
  - Quantum circuits

Outlook of today's tutorial

- First part: Introduction to mathematics of QC
  - QuBit states
  - State manipulation via unitary matrices
  - Quantum circuits
- Second part (after LRZ site visit): Classification with quantum computing

Outlook of today's tutorial

- First part: Introduction to mathematics of QC
  - QuBit states
  - State manipulation via unitary matrices
  - Quantum circuits
- Second part (after LRZ site visit): Classification with quantum computing
  - Quantum programming in PennyLane framework

Outlook of today's tutorial

- First part: Introduction to mathematics of QC
  - QuBit states
  - State manipulation via unitary matrices
  - Quantum circuits
- Second part (after LRZ site visit): Classification with quantum computing
  - Quantum programming in PennyLane framework
  - Variational quantum circuits

Outlook of today's tutorial

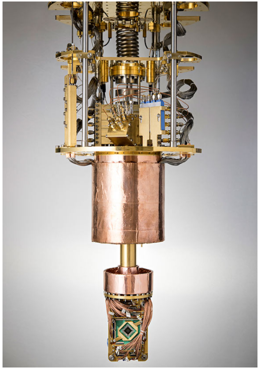
- First part: Introduction to mathematics of QC
  - QuBit states
  - State manipulation via unitary matrices
  - Quantum circuits
- Second part (after LRZ site visit): Classification with quantum computing
  - Quantum programming in PennyLane framework
  - Variational quantum circuits
  - Binary classification

Exercises

- Course Material: https://github.com/mnm-team/qc-focused-tutorial-23/
- Sheet 1: Pen and Paper
- Sheet 2: Jupyter notebooks

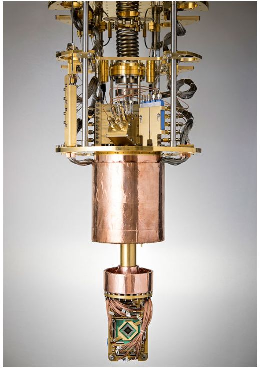
What is Quantum Computing?

- Exploiting quantum mechanical effects for computations



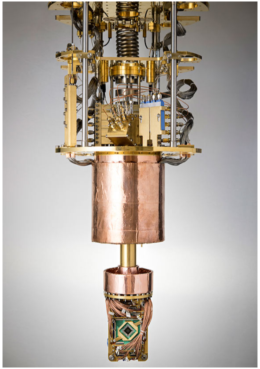
What is Quantum Computing?

- Exploiting quantum mechanical effects for computations
  - Superposition



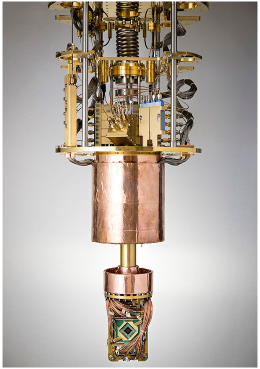
What is Quantum Computing?

- Exploiting quantum mechanical effects for computations
  - Superposition
  - Interference



What is Quantum Computing?

- Exploiting quantum mechanical effects for computations
  - Superposition
  - Interference
- Optimal hardware yet to be determined



Why do we need Quantum Computing?

- Speed up classical computation

Why do we need Quantum Computing?

- Speed up classical computation
  - Prime factorization

Why do we need Quantum Computing?

- Speed up classical computation
  - Prime factorization
  - Optimization/ Search algorithms

Why do we need Quantum Computing?

- Speed up classical computation
  - Prime factorization
  - Optimization/ Search algorithms
  - Simulation of quantum systems

Why do we need Quantum Computing?

- Speed up classical computation
  - Prime factorization
  - Optimization/ Search algorithms
  - Simulation of quantum systems
- Cryptography

Why do we need Quantum Computing?

- Speed up classical computation
  - Prime factorization
  - Optimization/ Search algorithms
  - Simulation of quantum systems
- Cryptography
  - Bug-proof communication

Why do we need Quantum Computing?

- Speed up classical computation
  - Prime factorization
  - Optimization/ Search algorithms
  - Simulation of quantum systems
- Cryptography
  - Bug-proof communication
  - True randomness

Single QuBit States

- QuBit state: $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ or a *superposition* of $|0\rangle$ and $|1\rangle$

Single QuBit States

- QuBit state: $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ or a *superposition* of $|0\rangle$ and $|1\rangle$
- Represented by a two-dimensional *normalized* vector:

$$|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \alpha |0\rangle + \beta |1\rangle \qquad \alpha, \beta \in \mathbb{C}$$

where

$$|\alpha|^2 + |\beta|^2 = 1$$

Measurement

- We cannot read the state of a QuBit, we can only measure the QuBit which then collapses to one of the basis states

Measurement

- We cannot read the state of a QuBit, we can only measure the QuBit which then collapses to one of the basis states
- Given the QuBit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is measured. Then...

Measurement

- We cannot read the state of a QuBit, we can only measure the QuBit which then collapses to one of the basis states
- Given the QuBit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is measured. Then...

  ... the probability of measuring $|0\rangle$ is $|\alpha|^2$

Measurement

- We cannot read the state of a QuBit, we can only measure the QuBit which then collapses to one of the basis states
- Given the QuBit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is measured. Then...

  ... the probability of measuring $|0\rangle$ is $|\alpha|^2$

  ... the probability of measuring $|1\rangle$ is $|\beta|^2$

Measurement

- We cannot read the state of a QuBit, we can only measure the QuBit which then collapses to one of the basis states
- Given the QuBit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is measured. Then...

  ... the probability of measuring $|0\rangle$ is $|\alpha|^2$

  ... the probability of measuring $|1\rangle$ is $|\beta|^2$
- The original quantum state cannot be restored anymore

Measurement

- We cannot read the state of a QuBit, we can only measure the QuBit which then collapses to one of the basis states
- Given the QuBit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is measured. Then...

  ... the probability of measuring $|0\rangle$ is $|\alpha|^2$

  ... the probability of measuring $|1\rangle$ is $|\beta|^2$
- The original quantum state cannot be restored anymore

$\rightarrow$ We have to measure multiple times to approximate the original quantum state

Dirac notation (bra-ket notation)

- "ket" is complex column vector in Hilbert space: $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$

Dirac notation (bra-ket notation)

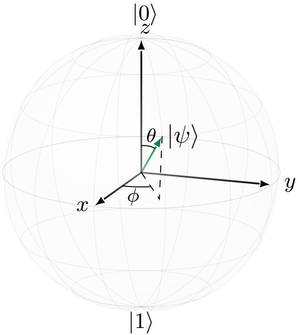
- "ket" is complex column vector in Hilbert space: $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$
- "bra" is complex conjugate transpose of ket, i.e. a row vector: $\langle\psi| = \begin{pmatrix} \alpha^* & \beta^* \end{pmatrix}$

Dirac notation (bra-ket notation)

- “ket” is complex column vector in Hilbert space: $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$
- “bra” is complex conjugate transpose of ket, i.e. a row vector: $\langle\psi| = \begin{pmatrix} \alpha^* & \beta^* \end{pmatrix}$
- “bra-ket” is an inner product yielding a scalar:
  $\langle\psi, \psi'\rangle = \begin{pmatrix} \alpha^* & \beta^* \end{pmatrix} \cdot \begin{pmatrix} \alpha' \\ \beta' \end{pmatrix} = \alpha^*\alpha' + \beta^*\beta' \in \mathbb{C}$

Bloch Sphere

We can interpret a single QuBit vector as a point on a three dimensional plane:



$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)e^{i\phi}|1\rangle$$

Unitary Operations

- Quantum operations represented by quantum gates

Unitary Operations

- Quantum operations represented by quantum gates
- The length of the state vector must remain $|\alpha|^2 + |\beta|^2 = 1$

Unitary Operations

- Quantum operations represented by quantum gates
- The length of the state vector must remain $|\alpha|^2 + |\beta|^2 = 1$

$\rightarrow$ Requires unitary matrix operations on the state vector

$$\begin{pmatrix} u_{00} & u_{10} \\ u_{01} & u_{11} \end{pmatrix} \cdot \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} u_{00} \cdot \alpha + u_{10} \cdot \beta \\ u_{01} \cdot \alpha + u_{11} \cdot \beta \end{pmatrix}$$

Unitary Operations

- Quantum operations represented by quantum gates
- The length of the state vector must remain $|\alpha|^2 + |\beta|^2 = 1$

$\rightarrow$ Requires unitary matrix operations on the state vector

$$\begin{pmatrix} u_{00} & u_{10} \\ u_{01} & u_{11} \end{pmatrix} \cdot \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} u_{00} \cdot \alpha + u_{10} \cdot \beta \\ u_{01} \cdot \alpha + u_{11} \cdot \beta \end{pmatrix}$$

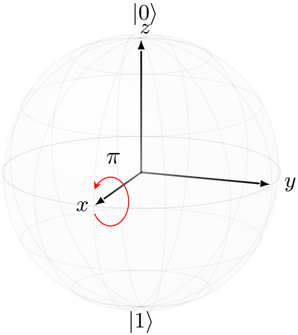
- We can represent operations on QuBits in the quantum circuit model:

$$|\psi\rangle \;\text{—}\boxed{U}\text{—}\; |\psi'\rangle$$

Single QuBit Gates I

Pauli matrices:

- $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ 

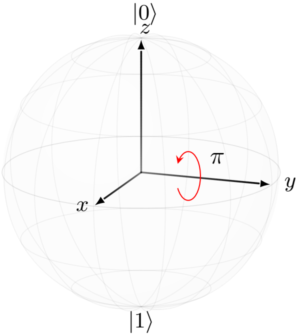


Single QuBit Gates I

Pauli matrices:

- $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ $\quad -\boxed{X}-$

- $Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$ $\quad -\boxed{Y}-$



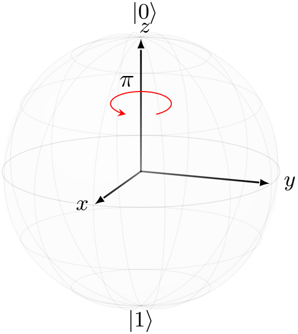
Single QuBit Gates I

Pauli matrices:

- $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ $\quad -\boxed{X}-$

- $Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$ $\quad -\boxed{Y}-$

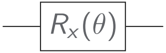
- $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ $\quad -\boxed{Z}-$

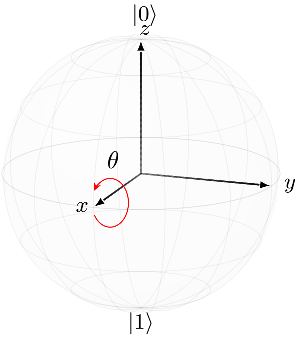


Arbitrary Rotations

$R_?(\theta)$-Gate: Rotation by θ around the X/Y/Z axis (?: X, Y, or Z)

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

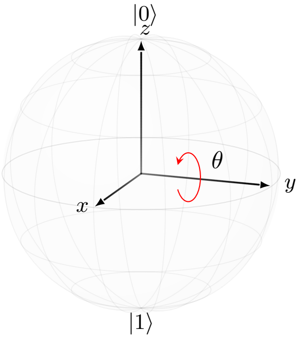




Arbitrary Rotations

R?(θ)-Gate: Rotation by θ around the X/Y/Z axis (?: X, Y, or Z)

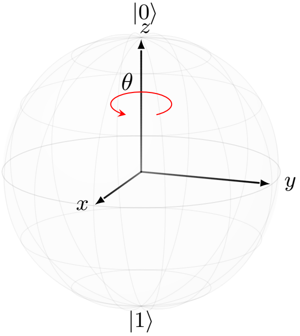
- $R_x(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -i\sin(\frac{\theta}{2}) \\ -i\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$ $-\boxed{R_x(\theta)}-$
- $R_y(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$ $-\boxed{R_y(\theta)}-$



Arbitrary Rotations

$R_?(\theta)$-Gate: Rotation by θ around the X/Y/Z axis (?: X, Y, or Z)

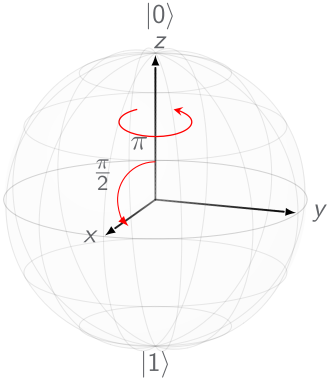
- $R_x(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -i\sin(\frac{\theta}{2}) \\ -i\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$ —$\boxed{R_x(\theta)}$—
- $R_y(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix}$ —$\boxed{R_y(\theta)}$—
- $R_z(\theta) = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix}$ —$\boxed{R_z(\theta)}$—



Single QuBit Gates II

- Hadamard-Gate (H-Gate)

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

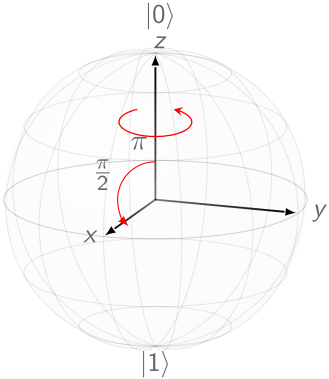


Single QuBit Gates II

- Hadamard-Gate (H-Gate)

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

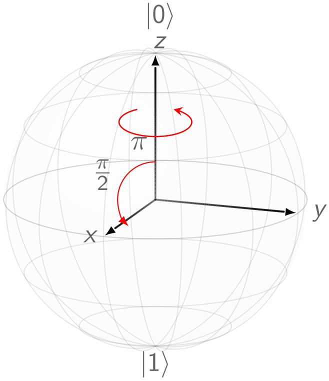
- Applying the H-Gate on a QuBit with state $|0\rangle$ or $|1\rangle$ introduces *equal* superposition:
  - $H\,|0\rangle = \frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle$
  - $H\,|1\rangle = \frac{1}{\sqrt{2}}\,|0\rangle - \frac{1}{\sqrt{2}}\,|1\rangle$



Single QuBit Gates II

- Hadamard-Gate (H-Gate)

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



- Applying the H-Gate on a QuBit with state $|0\rangle$ or $|1\rangle$ introduces *equal* superposition:
  - $H|0\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$
  - $H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$

- Probability of measuring $|0\rangle$ for state $H|0\rangle$ or $H|1\rangle = \frac{1}{2}$

Quantum Registers

- Multiple QuBits can be combined to a *quantum register* by calculating their tensorproduct:

$$\begin{pmatrix} \alpha_{n-1} \\ \beta_{n-1} \end{pmatrix} \otimes \begin{pmatrix} \alpha_{n-2} \\ \beta_{n-2} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix}$$

Quantum Registers

- Multiple QuBits can be combined to a *quantum register* by calculating their tensorproduct:

$$\begin{pmatrix} \alpha_{n-1} \\ \beta_{n-1} \end{pmatrix} \otimes \begin{pmatrix} \alpha_{n-2} \\ \beta_{n-2} \end{pmatrix} \otimes \cdots \otimes \begin{pmatrix} \alpha_0 \\ \beta_0 \end{pmatrix}$$

- Quantum operations on quantum registers are represented by the tensorproduct of the corresponding matrices (dimension: $2^n \times 2^n$)

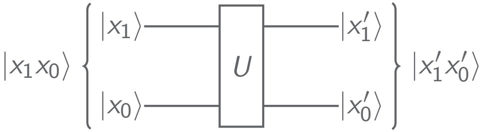
Two QuBit Register

$$|\psi\rangle = a_{00}\,|00\rangle + a_{01}\,|01\rangle + a_{10}\,|10\rangle + a_{11}\,|11\rangle$$

$$|00\rangle = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, |01\rangle = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}, |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, |11\rangle = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

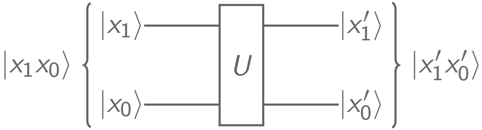
Multi-QuBit Gates

We can also apply gates on multiple QuBits:



Multi-QuBit Gates

We can also apply gates on multiple QuBits:



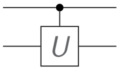
For instance, we can combine the existing single QuBit gates using the tensor product:

$$X \otimes X =$$

Controlled QuBit Operations

- An operation on a QuBit may depend on the state of another QuBit $\rightarrow$ Any single-QuBit gate can have a "controlled version"

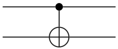
$$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{10} \\ 0 & 0 & u_{01} & u_{11} \end{pmatrix}$$



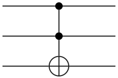
Controlled QuBit Operations

- An operation on a QuBit may depend on the state of another QuBit $\rightarrow$ Any single-QuBit gate can have a “controlled version”
- Most common: CX/CNOT and CCX/CCNOT/Toffoli Gate

$$CX = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$



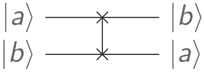
$$CCX = \begin{pmatrix} & & & 0 & 0 \\ & \mathbb{I}_6 & & \vdots & \vdots \\ & & & 0 & 0 \\ 0 & \cdots & 0 & 0 & 1 \\ 0 & \cdots & 0 & 1 & 0 \end{pmatrix}$$



The SWAP Operation

- The SWAP operation “swaps” the states of two qubits:

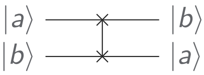
$$\text{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



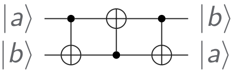
The SWAP Operation

- The SWAP operation “swaps” the states of two qubits:

$$\mathrm{SWAP} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

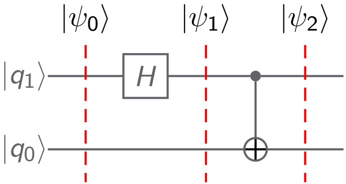


- It can be implemented with three CNOT gates:



Entanglement I

By applying a controlled gate (e.g., CNOT) on QuBits in superposition we can create entanglement:



e.g., $|q_1 q_0\rangle = |00\rangle$:

$$|\psi_2\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\1\end{pmatrix} = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

$\Rightarrow$ We either measure both QuBits in $|0\rangle$ **or** in $|1\rangle$.

Entanglement II

Definition

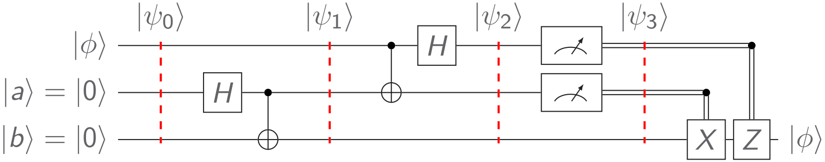
QuBits are *entangled* with each other iff the QuBit state cannot be represented as a tensor product of single QuBit states.

$$\text{e.g., } \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\1\end{pmatrix} = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

No-Cloning Theorem

- Entanglement is **not** equivalent to cloning a QuBit state
- It is not possible to clone or copy an arbitrary QuBit state
- With entanglement, it is possible to teleport an arbitrary state from one QuBit to another

Teleportation



Observables

- Hermitian operators that describe quantum-physical characteristics of a system

Observables

- Hermitian operators that describe quantum-physical characteristics of a system
- With an observable O, the expectation value of a measurement on QuBit $|\psi\rangle$ can be determined:

$$\langle\psi| O |\psi\rangle$$

Observables

- Hermitian operators that describe quantum-physical characteristics of a system
- With an observable O, the expectation value of a measurement on QuBit $|\psi\rangle$ can be determined:

$$\langle\psi| O |\psi\rangle$$

- The expectation value is calculated with respect to the eigenvalues of the observable, not the measurement bases

Pauli-Z Matrix as Observable

- Z matrix: $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$
- The expectation value can be calculated with $\langle\psi| Z |\psi\rangle$
- Examples:
  1. $\langle 0| Z |0\rangle =$
  2. $\langle 1| Z |1\rangle =$
  3. $\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|) \cdot Z \cdot \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) =$

Summary Classical vs. Quantum Computing

	Classical Computing	Quantum Computing
Basic States	Either 0 or 1	Superposition of 0 and 1
Operations	Logic (boolean) gates	Unitary reversible gates
Copying States	Yes	No
Readout	No information loss	Superposition is destroyed; information loss

Noise in Quantum Computers

- Current quantum computers are still very noisy

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary
- There are not that many QuBits at the moment (IBM: around 400 QuBits)

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary
- There are not that many QuBits at the moment (IBM: around 400 QuBits)
- Quantum algorithms like Grover's or Shor's algorithm assume ideal quantum computers with error corrected QuBits

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary
- There are not that many QuBits at the moment (IBM: around 400 QuBits)
- Quantum algorithms like Grover's or Shor's algorithm assume ideal quantum computers with error corrected QuBits
- Near-term goal: design quantum circuits/algorithms that...

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary
- There are not that many QuBits at the moment (IBM: around 400 QuBits)
- Quantum algorithms like Grover's or Shor's algorithm assume ideal quantum computers with error corrected QuBits
- Near-term goal: design quantum circuits/algorithms that...

  ... can adapt to noise

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary
- There are not that many QuBits at the moment (IBM: around 400 QuBits)
- Quantum algorithms like Grover's or Shor's algorithm assume ideal quantum computers with error corrected QuBits
- Near-term goal: design quantum circuits/algorithms that...

  ... can adapt to noise

  ... do not use many QuBits and/or gates

Noise in Quantum Computers

- Current quantum computers are still very noisy
- Error correction and mitigation methods are necessary
- There are not that many QuBits at the moment (IBM: around 400 QuBits)
- Quantum algorithms like Grover's or Shor's algorithm assume ideal quantum computers with error corrected QuBits
- Near-term goal: design quantum circuits/algorithms that...

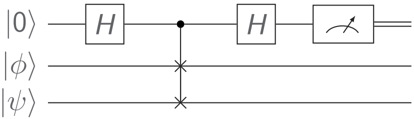
  ... can adapt to noise

  ... do not use many QuBits and/or gates
- Near-term quantum algorithms: Variational Quantum Algorithms (VQAs)

Swap Test I

Means to measure the distance d of two quantum states $|\phi\rangle$ and $|\psi\rangle$

$d = |\langle\psi|\phi\rangle|^2$ (angle^2 between $|\psi\rangle$ and $|\phi\rangle$)



Swap Test II

- At the end of the circuit the probability of measuring $|0\rangle$ is the following

$$P(\text{“Measure } |0\rangle \text{”}) = \frac{1}{2} + \frac{1}{2} |\langle \psi | \phi \rangle|^2$$

Swap Test II

- At the end of the circuit the probability of measuring $|0\rangle$ is the following

$$P(\text{“Measure } |0\rangle \text{”}) = \frac{1}{2} + \frac{1}{2}|\langle\psi|\phi\rangle|^2$$

  - If $|\psi\rangle$ and $|\phi\rangle$ are orthogonal ($|\langle\psi|\phi\rangle|^2 = 0$), then $P(\text{“Measure } |0\rangle \text{”}) = \frac{1}{2}$
  - If $|\psi\rangle$ and $|\phi\rangle$ are equal ($|\langle\psi|\phi\rangle|^2 = 1$), then $P(\text{“Measure } |0\rangle \text{”}) = 1$

Swap Test II

- At the end of the circuit the probability of measuring $|0\rangle$ is the following

$$P(\text{“Measure } |0\rangle\text{”}) = \frac{1}{2} + \frac{1}{2}|\langle\psi|\phi\rangle|^2$$

  - If $|\psi\rangle$ and $|\phi\rangle$ are orthogonal ($|\langle\psi|\phi\rangle|^2 = 0$), then $P(\text{“Measure } |0\rangle\text{”}) = \frac{1}{2}$
  - If $|\psi\rangle$ and $|\phi\rangle$ are equal ($|\langle\psi|\phi\rangle|^2 = 1$), then $P(\text{“Measure } |0\rangle\text{”}) = 1$
- Then $d = |\langle\psi|\phi\rangle|^2 = 2P(\text{“Measure } |0\rangle\text{”}) - 1$

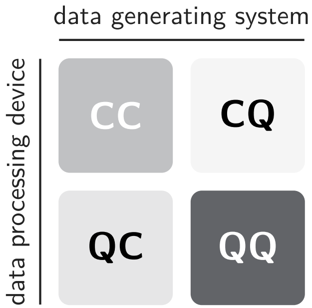
Swap Test II

- At the end of the circuit the probability of measuring $|0\rangle$ is the following

$$P(\text{“Measure } |0\rangle\text{”}) = \frac{1}{2} + \frac{1}{2}|\langle\psi|\phi\rangle|^2$$

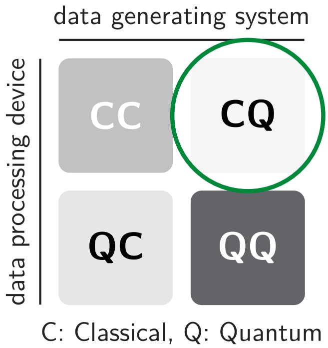
  - If $|\psi\rangle$ and $|\phi\rangle$ are orthogonal ($|\langle\psi|\phi\rangle|^2 = 0$), then $P(\text{“Measure } |0\rangle\text{”}) = \frac{1}{2}$
  - If $|\psi\rangle$ and $|\phi\rangle$ are equal ($|\langle\psi|\phi\rangle|^2 = 1$), then $P(\text{“Measure } |0\rangle\text{”}) = 1$
- Then $d = |\langle\psi|\phi\rangle|^2 = 2P(\text{“Measure } |0\rangle\text{”}) - 1$
- With multiple repetitions of the SWAP test, the distance d can be approximated with any precision

Overview



C: Classical, Q: Quantum

Overview


data generating system
data processing device
CC
CQ
QC
QQ
C: Classical, Q: Quantum

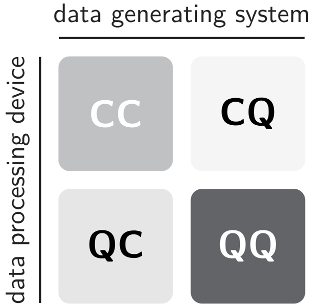
Overview – Classical Machine Learning

data processing device \ data generating system		
	CC	**CQ**
	QC	**QQ**

C: Classical, Q: Quantum

1. Input
   - Training data X_{train}
   - Initial parameter vector θ

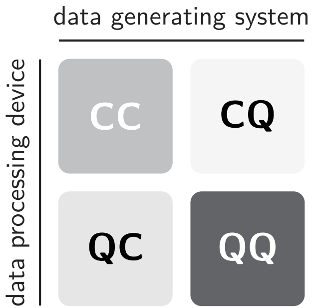
Overview – Classical Machine Learning



C: Classical, Q: Quantum

1. Input
   - Training data X_{train}
   - Initial parameter vector θ
2. Model: $f(X_{\text{train}}, \theta) = \hat{y}$

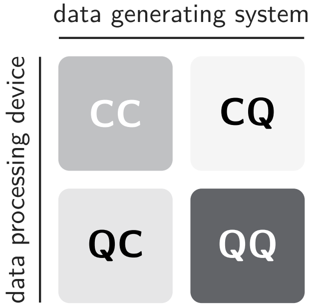
Overview – Classical Machine Learning



C: Classical, Q: Quantum

1. Input
   - Training data X_{train}
   - Initial parameter vector θ
2. Model: $f(X_{\text{train}}, \theta) = \hat{y}$
3. Prediction: Score with cost function $C(\hat{y}, y)$

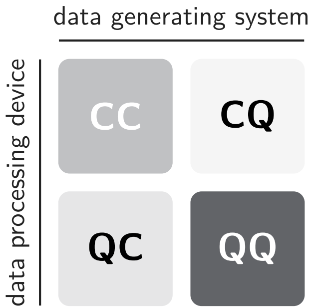
Overview – Classical Machine Learning



C: Classical, Q: Quantum

1. Input
   - Training data X_{train}
   - Initial parameter vector θ
2. Model: $f(X_{\text{train}}, \theta) = \hat{y}$
3. Prediction: Score with cost function $C(\hat{y}, y)$
4. Update θ based on gradient-based techniques

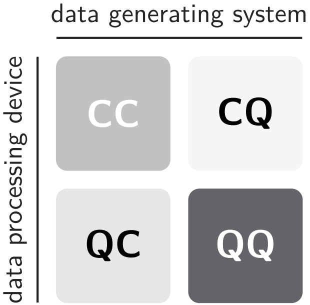
Overview – Classical Machine Learning



C: Classical, Q: Quantum

1. Input
   - Training data X_{train}
   - Initial parameter vector θ
2. Model: $f(X_{\text{train}}, \theta) = \hat{y}$
3. Prediction: Score with cost function $C(\hat{y}, y)$
4. Update θ based on gradient-based techniques
5. Repeat starting from step 2

Overview – Classical Machine Learning



C: Classical, Q: Quantum

1. Input
   - Training data X_{train}
   - Initial parameter vector θ
2. Model: $f(X_{\text{train}}, \theta) = \hat{y}$
3. Prediction: Score with cost function $C(\hat{y}, y)$
4. Update θ based on gradient-based techniques
5. Repeat starting from step 2

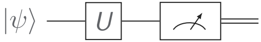
Classical Machine Learning → Quantum Machine Learning

- The calculation of the model (step 2) can also be done on a quantum computer

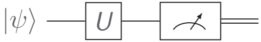
Classical Machine Learning → Quantum Machine Learning

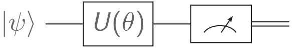
- The calculation of the model (step 2) can also be done on a quantum computer
- For this, we use variational quantum circuits with parametrized operations:

Classical Machine Learning → Quantum Machine Learning

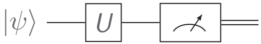
- The calculation of the model (step 2) can also be done on a quantum computer
- For this, we use variational quantum circuits with parametrized operations:
  - "Standard" quantum circuit: 

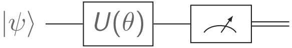
Classical Machine Learning → Quantum Machine Learning

- The calculation of the model (step 2) can also be done on a quantum computer
- For this, we use variational quantum circuits with parametrized operations:
  - "Standard" quantum circuit: 

  - Variational quantum circuit: 

Classical Machine Learning → Quantum Machine Learning

- The calculation of the model (step 2) can also be done on a quantum computer
- For this, we use variational quantum circuits with parametrized operations:
  - "Standard" quantum circuit: 

  - Variational quantum circuit: 

→ Goal: optimize the parameter value(s) for the application

Steps

Input: Training data (already labelled data samples)

Steps

Input: Training data (already labelled data samples)

Task: Predict the labels for new data

Steps

Input: Training data (already labelled data samples)

Task: Predict the labels for new data

1. Encode the classical data into a quantum state

Steps

Input: Training data (already labelled data samples)

Task: Predict the labels for new data

1. Encode the classical data into a quantum state
2. Encode the model into a parameterized quantum circuit

Steps

Input: Training data (already labelled data samples)

Task: Predict the labels for new data

1. Encode the classical data into a quantum state
2. Encode the model into a parameterized quantum circuit
3. Measure the circuit to determine the labels

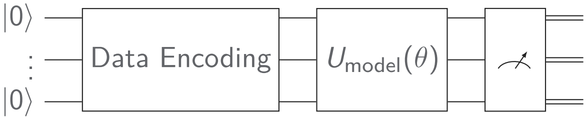
Steps

Input: Training data (already labelled data samples)

Task: Predict the labels for new data

1. Encode the classical data into a quantum state
2. Encode the model into a parameterized quantum circuit
3. Measure the circuit to determine the labels
4. Use (classical) optimization techniques to update the parameters of the quantum circuit

Quantum Circuit



Basis Encoding

- Encode data in computational basis states.

Basis Encoding

- Encode data in computational basis states.
- E.g. encode the number $N = 3$ in 2-QuBit state:

Basis Encoding

- Encode data in computational basis states.
- E.g. encode the number $N = 3$ in 2-QuBit state:
  - Possible basis states for 2 QuBits: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$.

Basis Encoding

- Encode data in computational basis states.
- E.g. encode the number $N = 3$ in 2-QuBit state:
  - Possible basis states for 2 QuBits: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$.
  - Encode 3 as $|\psi\rangle = |11\rangle$.

Basis Encoding

- Encode data in computational basis states.
- E.g. encode the number $N = 3$ in 2-QuBit state:
  - Possible basis states for 2 QuBits: $|00\rangle, |01\rangle, |10\rangle, |11\rangle$.
  - Encode 3 as $|\psi\rangle = |11\rangle$.
- Similar to classical computing.

Amplitude Encoding

- Encode data in 2^n amplitudes of n-QuBit state

Amplitude Encoding

- Encode data in 2^n amplitudes of n-QuBit state
- E.g. encode feature vector $(1.2, 0.6, -2, 1.5)$ into 2-QuBit state

Amplitude Encoding

- Encode data in 2^n amplitudes of n-QuBit state
- E.g. encode feature vector $(1.2, 0.6, -2, 1.5)$ into 2-QuBit state
  - Normalization to valid state vector: $\sim (0.423, 0.211, -0.705, 0.529)$

Amplitude Encoding

- Encode data in 2^n amplitudes of n-QuBit state
- E.g. encode feature vector $(1.2, 0.6, -2, 1.5)$ into 2-QuBit state
  - Normalization to valid state vector: $\sim (0.423, 0.211, -0.705, 0.529)$
  - Encode feature vector as $|\psi\rangle = 0.423\,|00\rangle + 0.211\,|01\rangle - 0.705\,|10\rangle + 0.529\,|11\rangle$

Angle Encoding

- Encode data in X, Y, Z rotation angles of n-QuBits.

Angle Encoding

- Encode data in X, Y, Z rotation angles of n-QuBits.
- Data needs to be normalized in interval $[0, 2\pi]$

Angle Encoding

- Encode data in X, Y, Z rotation angles of n-QuBits.
- Data needs to be normalized in interval $[0, 2\pi]$
- E.g., encode feature vector $(1.2, 0.6, -2, 1.5)$ into the X rotations of 4-QuBit state:

Angle Encoding

- Encode data in X, Y, Z rotation angles of n-QuBits.
- Data needs to be normalized in interval $[0, 2\pi]$
- E.g., encode feature vector $(1.2, 0.6, -2, 1.5)$ into the X rotations of 4-QuBit state:

$$|0\rangle \; — \boxed{R_x(1.2)} —$$
$$|0\rangle \; — \boxed{R_x(0.6)} —$$
$$|0\rangle \; — \boxed{R_x(-2)} —$$
$$|0\rangle \; — \boxed{R_x(1.5} —$$

Variational Model

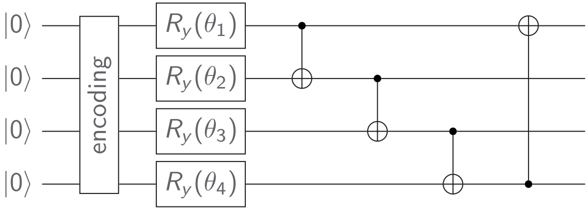
- Parameterized model containing (repeated) layers of rotation and entanglement gates (e.g. $R_y + CNOT$)

Variational Model

- Parameterized model containing (repeated) layers of rotation and entanglement gates (e.g. $R_y + CNOT$)
- Map data to higher-dimensional feature space

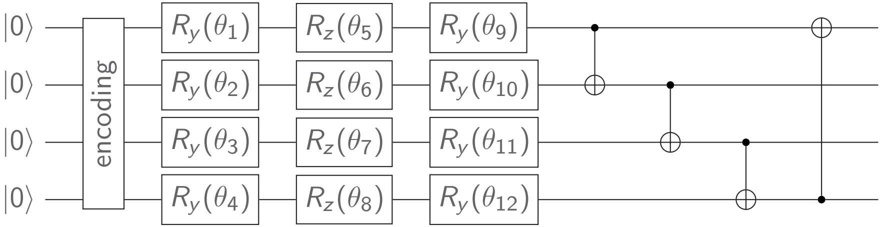
Variational Model

- Parameterized model containing (repeated) layers of rotation and entanglement gates (e.g. $R_y + CNOT$)
- Map data to higher-dimensional feature space
- Common layer structures:



Variational Model

- Parameterized model containing (repeated) layers of rotation and entanglement gates (e.g. $R_y + CNOT$)
- Map data to higher-dimensional feature space
- Common layer structures:



Classification

- Repeated measurements yield expectation value of **Pauli Z operator** in $[-1, 1]$.

Classification

- Repeated measurements yield expectation value of **Pauli Z operator** in $[-1, 1]$.
- Extract labels from **expectation value**.

Classification

- Repeated measurements yield expectation value of **Pauli Z operator** in $[-1, 1]$.
- Extract labels from **expectation value**.
- E.g., for binary classification we could take the **sign function** as classification function.

Summary: Variational Quantum Classification

- Optimal data encoding and variational model is still an **open question**.

Summary: Variational Quantum Classification

- Optimal data encoding and variational model is still an **open question**.
- Variational quantum classifiers are **kernel methods** with measurement as **linear classifier**.

Summary: Variational Quantum Classification

- Optimal data encoding and variational model is still an **open question**.
- Variational quantum classifiers are **kernel methods** with measurement as **linear classifier**.
- Quantum advantage may be mainly in more diverse **data encoding**.

Outlook

- Scientists and big companies, like IBM, Intel, and Google, are already working on quantum computing and building quantum computers

Outlook

- Scientists and big companies, like IBM, Intel, and Google, are already working on quantum computing and building quantum computers
- Quantum computer with the most QuBits (2023): IBM Osprey (433 QuBits)

Outlook

- Scientists and big companies, like IBM, Intel, and Google, are already working on quantum computing and building quantum computers
- Quantum computer with the most QuBits (2023): IBM Osprey (433 QuBits)
- Other approaches: measurement-based quantum computing (MBQC), quantum annealing

Outlook

- Scientists and big companies, like IBM, Intel, and Google, are already working on quantum computing and building quantum computers
- Quantum computer with the most QuBits (2023): IBM Osprey (433 QuBits)
- Other approaches: measurement-based quantum computing (MBQC), quantum annealing
- Other variational algorithms: quantum approximate optimization algorithm (QAOA), variational quantum eigensolver (VQE), quantum random walks (QRW)

Outlook

- Scientists and big companies, like IBM, Intel, and Google, are already working on quantum computing and building quantum computers
- Quantum computer with the most QuBits (2023): IBM Osprey (433 QuBits)
- Other approaches: measurement-based quantum computing (MBQC), quantum annealing
- Other variational algorithms: quantum approximate optimization algorithm (QAOA), variational quantum eigensolver (VQE), quantum random walks (QRW)
- Other algorithms for fault-tolerant QC: Grover's algorithm, Shor's algorithm

Outlook

- Scientists and big companies, like IBM, Intel, and Google, are already working on quantum computing and building quantum computers
- Quantum computer with the most QuBits (2023): IBM Osprey (433 QuBits)
- Other approaches: measurement-based quantum computing (MBQC), quantum annealing
- Other variational algorithms: quantum approximate optimization algorithm (QAOA), variational quantum eigensolver (VQE), quantum random walks (QRW)
- Other algorithms for fault-tolerant QC: Grover's algorithm, Shor's algorithm
- Quantum projects in Munich: Munich Quantum Valley, MuniQC Atoms/SC

References I

[1] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, en, 10th anniversary ed. Cambridge; New York: Cambridge University Press, 2010, ISBN: 978-1-107-00217-3.

[2] N. S. Yanofsky and M. A. Mannucci, *Quantum Computing for Computer Scientists*. Cambridge: Cambridge University Press, 2008, ISBN: 978-0-521-87996-5. DOI: 10.1017/CBO9780511813887. [Online]. Available: https://www.cambridge.org/core/books/quantum-computing-for-computer-scientists/8AEA723BEE5CC9F5C03FDD4BA850C711.

References II

[3] M. Homeister, *Quantum Computing verstehen: Grundlagen – Anwendungen – Perspektiven* (Computational Intelligence), de. Wiesbaden: Springer Fachmedien, 2022, ISBN: 978-3-658-36433-5. DOI: 10.1007/978-3-658-36434-2. [Online]. Available: https://link.springer.com/10.1007/978-3-658-36434-2.

[4] M. Schuld and F. Petruccione, *Machine Learning with Quantum Computers* (Quantum Science and Technology), en. Cham: Springer International Publishing, 2021, ISBN: 978-3-030-83097-7. DOI: 10.1007/978-3-030-83098-4. [Online]. Available: https://link.springer.com/10.1007/978-3-030-83098-4.

References III

[5] M. Schuld, I. Sinayskiy, and F. Petruccione, "An introduction to quantum machine learning," en, *Contemporary Physics*, vol. 56, no. 2, pp. 172–185, Apr. 2015, arXiv:1409.3097 [quant-ph], ISSN: 0010-7514, 1366-5812. DOI: 10.1080/00107514.2014.964942.

[6] M. Schuld, "Supervised quantum machine learning models are kernel methods,", no. arXiv:2101.11020, Apr. 2021, arXiv:2101.11020 [quant-ph, stat]. DOI: 10.48550/arXiv.2101.11020. [Online]. Available: http://arxiv.org/abs/2101.11020.