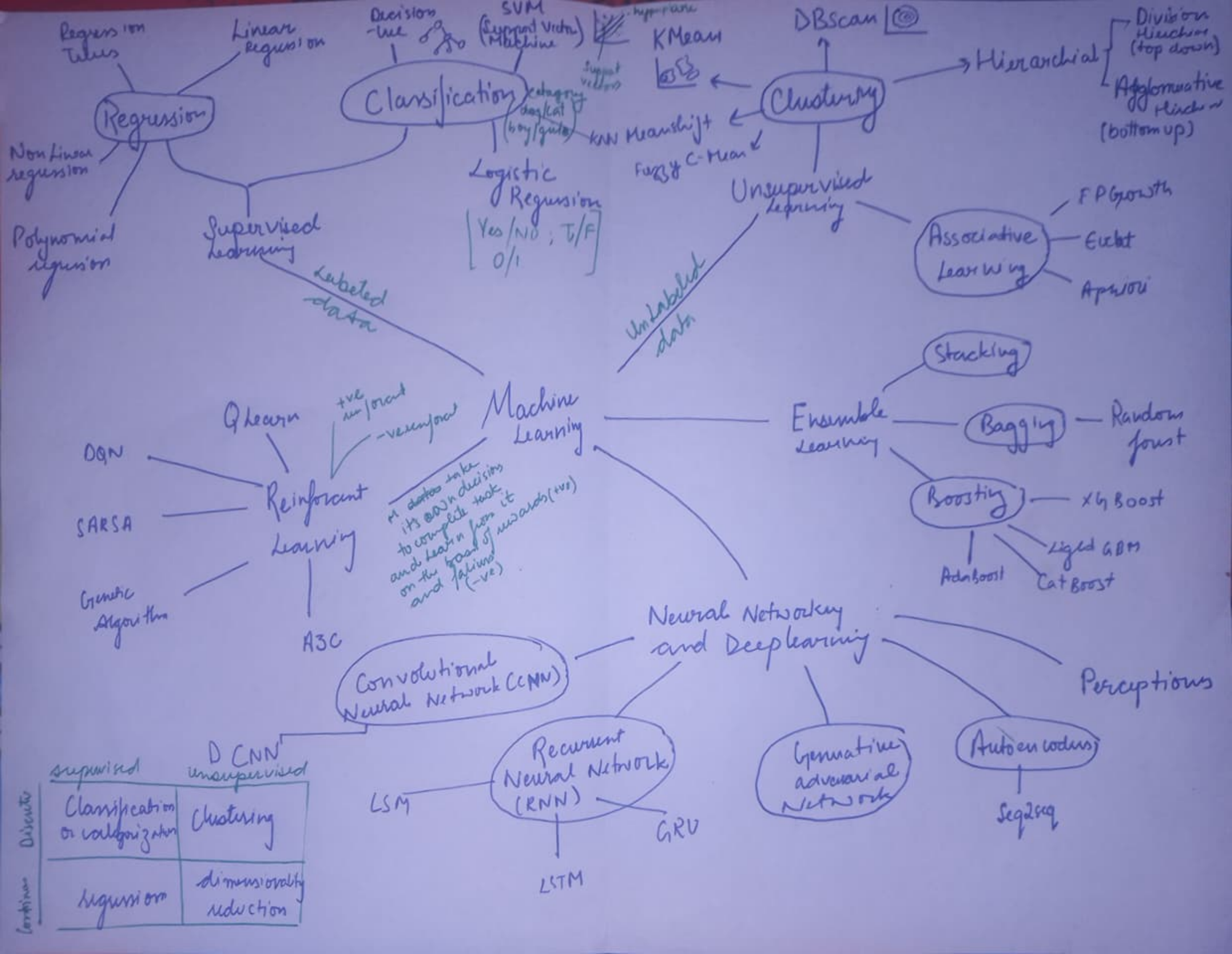


	supervised	unsupervised
Discrete	Classification or categorization	Clustering
Continuous	Regression	dimensionality reduction

Exploring Quantum Machine Learning

Machine Learning

"Learning and adapting without following explicit instructions by analysing and drawing inferences from patterns in data.

mathematical model

$g(x)$ true function $\xrightarrow{\text{approximate}}$ $f(\vec{x}, \vec{\theta})$ mathematical model

Machine Learning algo
- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
  - Dimensionality reduction
  - Clustering
- Reinforcement Learning

Supervised learning workflow

Training data : x_{train} → Model $f(x_{train}, \vec{\theta}) = \hat{y}$ → Cost $C(f(x_{train}, \vec{\theta}), y)$ → Compute gradient $\frac{\partial C(f(x_{train}, \vec{\theta}), y)}{\partial \theta}$ → Update $\vec{\theta}$ → Model

Model validation

- → Model should work well both on training and test data
- → The model should not overfit or underfit to training data

'bias - variance' trade-off

Quantum ML

Type of Algorithm →

Type of Data ↓

	classical	quantum
classical	CC	CQ
quantum	QC	QQ

Variational Circuit as a Classifier

Data x ⟶ Model: $f(x;\vec{\theta})$ ⟶ Get a prediction/answer label

$|\psi\rangle$ — $\boxed{U(\theta)}$ — $\boxed{M}$ $\langle z \rangle$

Variational circuit
parameterized quantum circuit (PQC)
Ansatz

Task - Supervised learning (suppose binary classification, $\{1, -1\}$)

Step 1 - Encode the classical data into a quantum state.

Step 2 - Apply a parameterized Model

Step 3 - Measure the circuit to extract labels

Step 4 - Use optimization techniques to update model parameters.

Data Encoding

Step 1

- Basis Encoding
- Amplitude Encoding
- Angle Encoding
- Higher order Encoding

$3 \rightarrow 011$

$|0\rangle$ ——————— $|0\rangle$

$|0\rangle$ —— $\boxed{X}$ —— $|1\rangle$

① Basis Encoding

Encode each n-bit feature into n qubits

3	1
0	3

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 1\,1 \\ 0\,1 \\ 0\,0 \\ 1\,1 \end{bmatrix}$$

$$= \begin{bmatrix} |11\rangle \\ |01\rangle \\ |00\rangle \\ |11\rangle \end{bmatrix}$$

one of the computational basis states of 8 qubits

② Amplitude Encoding

$|0\rangle$ —— $\boxed{U(x)}$ —— $|x\rangle$
$|0\rangle$ ——

Encoding into quantum state amplitude

$$|\psi_x\rangle = \sum_{i=1}^{N} x_i |i\rangle$$

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{19} \\ 1/\sqrt{19} \\ 0/\sqrt{19} \\ 3/\sqrt{19} \end{bmatrix}$$ — Amplitude of 2 qubits

③ Angle Encoding

3	1
0	3

Encode values into qubit rotation angles

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3/\sqrt{19} \\ 1/\sqrt{19} \\ 0/\sqrt{19} \\ 3/\sqrt{19} \end{bmatrix}$$

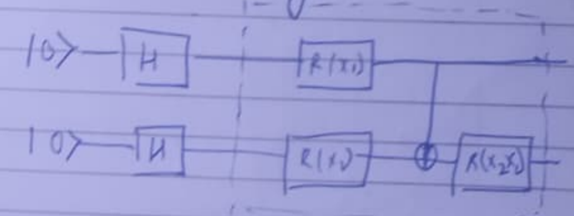
$|0\rangle$ — $\boxed{R_x(x_1)}$ —
$|0\rangle$ — $\boxed{R_x(x_2)}$ —
$|0\rangle$ — $\boxed{R_x(x_3)}$ —
$|0\rangle$ — $\boxed{R_x(x_4)}$ —

$$|x\rangle = \bigotimes_{i=1}^{N} \cos(x_i)|0\rangle + \sin(x_i)|1\rangle$$ angle encoding

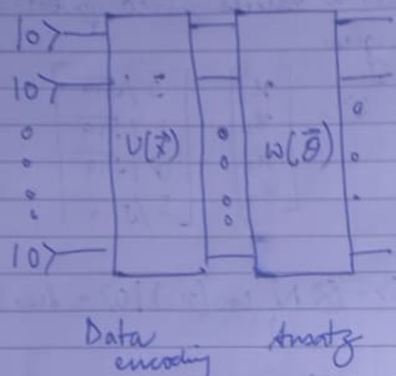
$$|x\rangle = \bigotimes_{i=1}^{n} \cos(x_{2i-1})|0\rangle + e^{x} \sin(x_{2i-1})|1\rangle$$

dense angle encoding

④ Higher Order Encoding

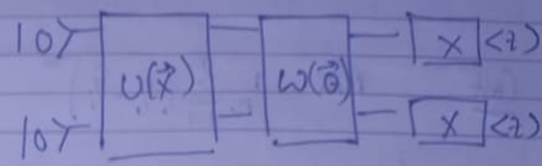


Step 2 - Applying a parametrized Model



Goal → designing a hardware-efficient ansatz

Step 3 - Measure the Extraction Labels



measurement outcomes → labels → cost function

↓

Classical optimizer

→ Binary classification {1, -1}: Qiski

1) Parity post-processing (00, 01, 10, 11) Sampl

2) Measure only 1 qubit (0 $\langle Z \rangle >= 0$, otherwise) estimat

Step 4 - Use optimization techniques to update parameters.

Cost : $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

If optimizer needs $\partial_\theta f(\theta)$

Parameter-shift rule

Gradient $|0\rangle^{\otimes n} - \boxed{U(\theta+s)} - \boxed{} = \hat{y}_{\theta+1}$

$-$

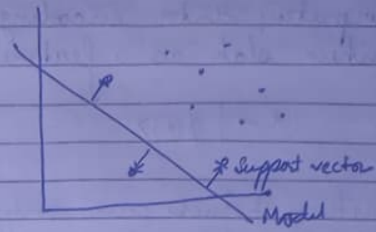
$|0\rangle^{\otimes n} - \boxed{U(\theta-s)} - \boxed{} = \hat{y}_{\theta-1}$

$\overline{}$

2

$s = \pi/2$

Support Vector machine



primal formulation

$f(x) = \vec{\theta}^T\vec{x} + b$

Dual formulation

$f(x) = \sum_{i=1}^{n} \alpha_i y_i (\vec{x}_i^T\vec{x}) + b$

When data is not linearly separated

original data set

data after feature map

$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$\phi(\vec{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{bmatrix}$

Quantum Kernels

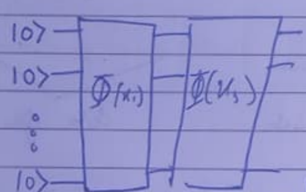
Interpreting data encoding to a quantum state as a feature map

$$x \to |\phi(x)\rangle$$

- Quantum kernels can only be expected to do better than classical kernels if they are hard to estimate classically
  . necessary but not sufficient

- It was shown that learning problems exist for which learners with access to quantum kernel methods have a quantum advantage over all classical learners

Quantum SVM

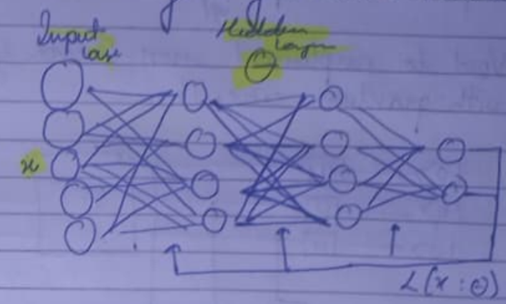


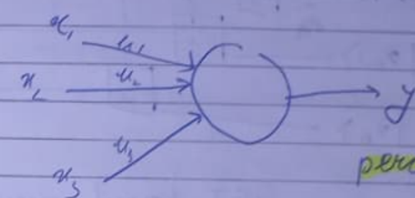
quantum kernel estimation

$$K_{i,j} = |\langle \Phi(x_j)|\Phi(x_i)\rangle|^2$$

$$pr[\text{measure } |0\rangle] = |\langle 0|\Phi(x_i)^\dagger \Phi(x_j)|0\rangle|^2$$

Classical feed-forward neural Network





perceptron

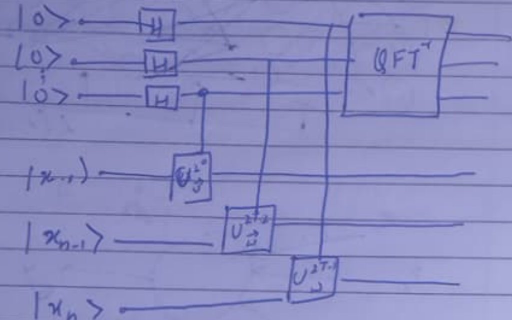
$$\boxed{f(\vec{x}) = \sigma(\vec{w}\cdot\vec{x} + b)}$$

non-linear activation function.

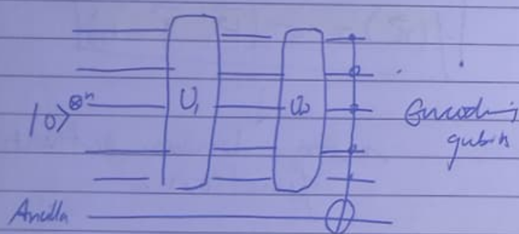
→ ~~would~~ wouldn't be able to capture complex patterns that deep neural networks can

Quantum perception

Need to implement non-linearity with quantum circuit



QFT based perception



Non linearity from measure

Convolutional neural network CNN

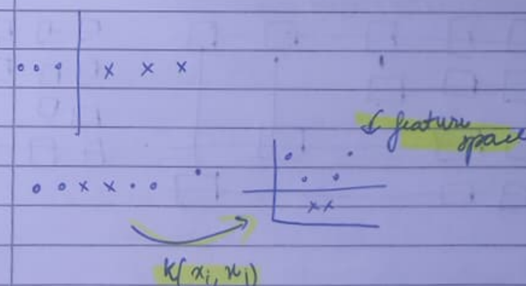
convolutional layer → Pooling → fully connected layer → Output layer

Quantum CNN

properties

- QCNNs have $O(\log N)$ layers and parameters
- They don't suffer from the problem of barren plateaus

Lec. 2



Lec 3 What is Quantum ML?

Classical Vs Quantum machine learning

Classical Generative Models
- Generative Adversarial Network (GAN)
- Restricted Boltzmann Machine (RBM)
- Variational Autoencoder (VAE)

Quantum Generative Models
- Quantum Generative Adversarial Network (QGAN)
- Quantum Boltzmann Machines (QBM)
- Quantum Circuit Born Machine (QCBM)

Star topology

Line topology

Training process (DDQCL)

Compare Loss function
- KL divergence
- Log-likelihood
- Maximum mean discrepancy

Update angles Gradient descent

Non-gradient method
- CMA-ES
- Particle swarm optimization

Lec 4

- Exponentially faster!
- 2^n bits
- 300 bits - not enough to store 1 image
- 300 qubits - number of particles in universe

Quantum ML

- QML uses quantum ckt to find pattern and relationship in data
- Quantum ckt do special operation on data to extract info. that classical computer cannot

- with these patterns quantum machine learning can make predictions about new data

Exploit Duality of QML

- **Quantum interference** :- allows QML algorithm to amplify certain states and surpress others, leading to more accurate predictions and better classification of data

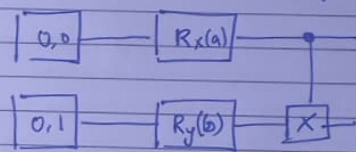
- **Superposition** - by using qubits in superposition, QML algorithms can process exponentially more information in || leading to more efficient calculations for ML

Cirq

- writing, manipulating and optimizing circuits made easier
- **simulations** (easier without a Quantum computer even)

Example

```
q0, q1 = cirq.GridQubit.rect(1,2)
circuit = cirq.Circuit(
 cirq.rx(a).on(q0),
 cirq.ry(b).on(q1)
 cirq.CNOT(control=q0, target=q1))
```



Tensorflow → **framework in python fom**