Quantum Machine Learning

Shubhashree Dash 4328734481

General introduction to Quantum Machine Learning

- Machine learning : “a set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty”

- Quantum Machine Learning includes many algorithms which apply the concept of machine learning depending on quantum computing laws. QML models have shown improvement of classical models. They have produced better results in shorter time.

- There have been various models developed, like quantum neural networks, quantum support vector machines, quantum k-nearest neighbours.

- Quantum classification algorithms can be divided into 3 categories: Quantum machine learning, quantum inspired machine learning, and hybrid quantum classical machine learning.

- **Quantum machine learning algorithms**
    - Algorithms that are quantum versions from conventional ML, as well as algorithms that can be executed on the real quantum device
- **Quantum Inspired Machine Learning**
    - Algorithms that apply quantum computing to improve classical methods of machine learning
- **Hybrid Quantum Classical Machine Learning**
    - Algorithms that combine quantum algorithms and classical to obtain higher performance and decrease in the learning cost

Quantum Classification Scheme

- In classical machine learning,
  - Data Set : $\{(x_1, y_1), \ldots, (x_i, y_i), \ldots, (x_n, y_n)\}$
  - Model : $f(x)$, where $f(x_i) = y_i$
- In quantum machine learning, the above is represented as
  - Data Set : $\{(|\psi_1\rangle, y_1), \ldots, (|\psi_i\rangle, y_i), \ldots, (|\psi_n\rangle, y_n)\}$
  - Model : $f(|\psi\rangle)$, where $f(|\psi_i\rangle) = y_i$

Encoding Methods

- Basis Encoding

Encode n-bit classical input into n-qubit quantum input, for each feature

For a dataset D, with M examples, and N features. The quantum dataset is

$$|D\rangle = \frac{1}{\sqrt{M}} \sum_{i=1}^{M} |x^{(i)}\rangle$$

where $x^{(i)}$ is a N-bit binary string defining the features of a single entry

- Amplitude Encoding

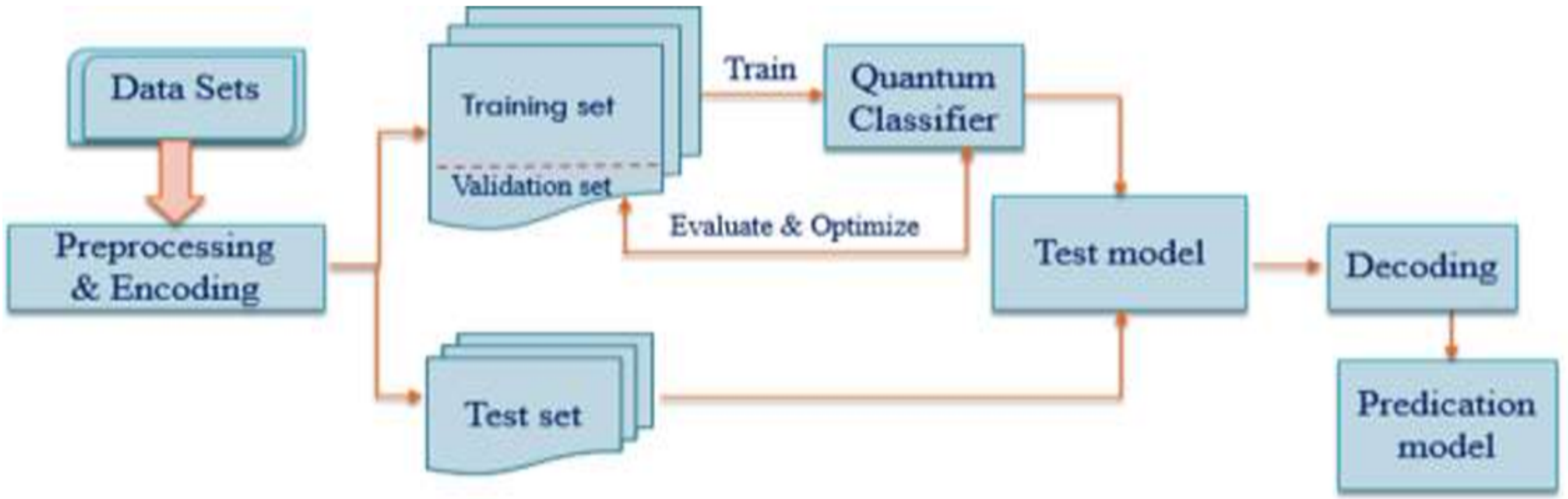
Encode the data into the amplitudes of the quantum state

For a dataset D, with M examples, and N features

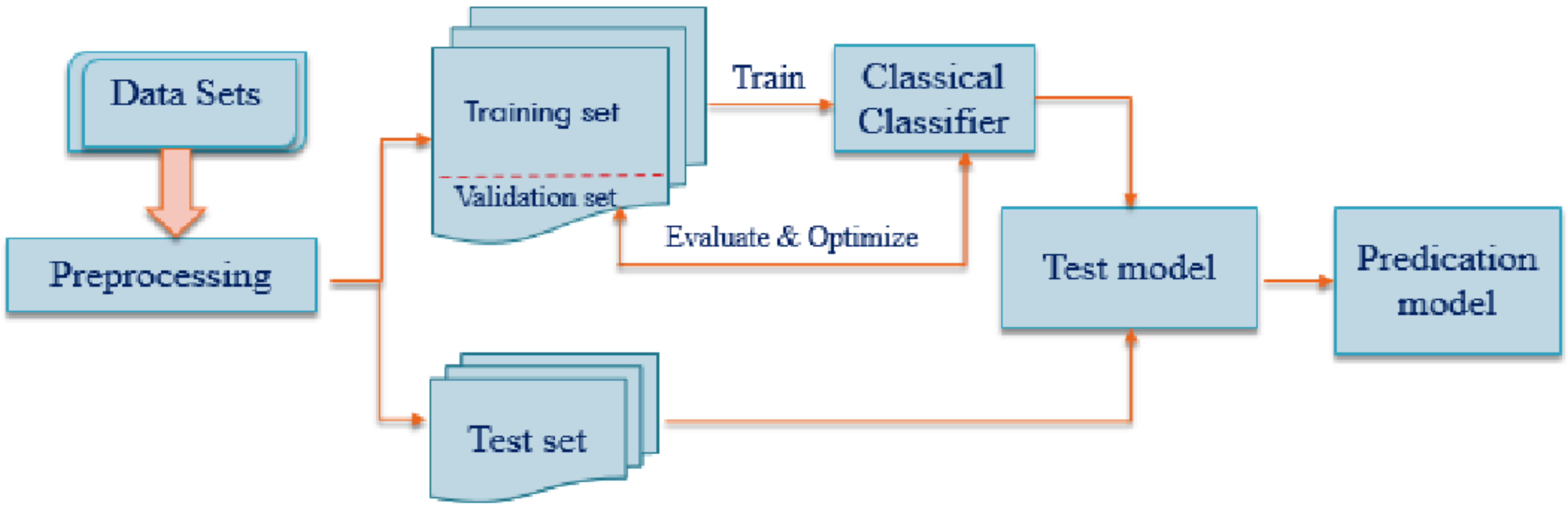
$$|\psi_x\rangle = \sum_{i=1}^{N} x_i |i\rangle$$

where x is a single entry in the dataset, x_i is i^{th} element of x, and $|i\rangle$ is the i^{th} computational basis state

Model Training Pipeline



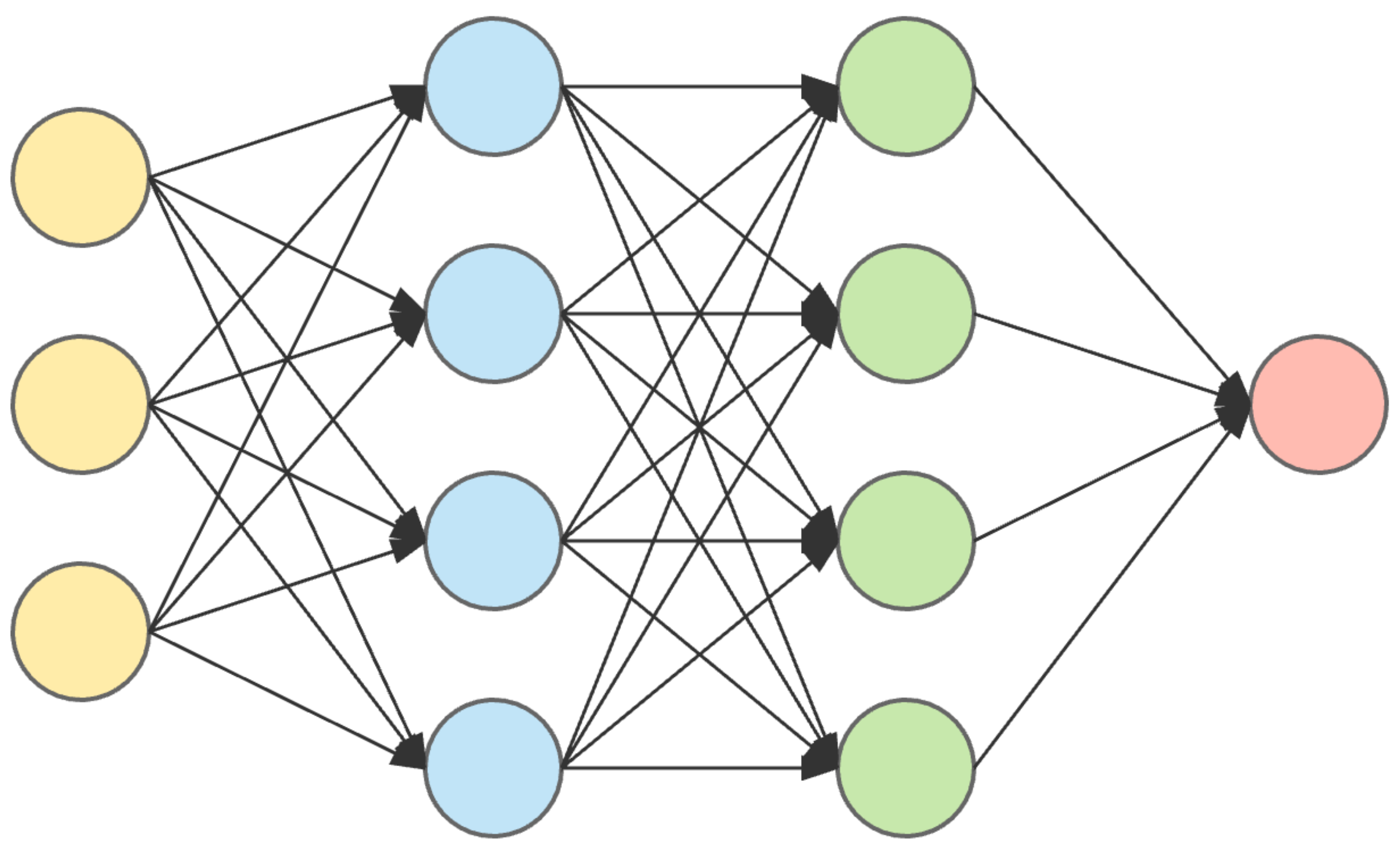
A

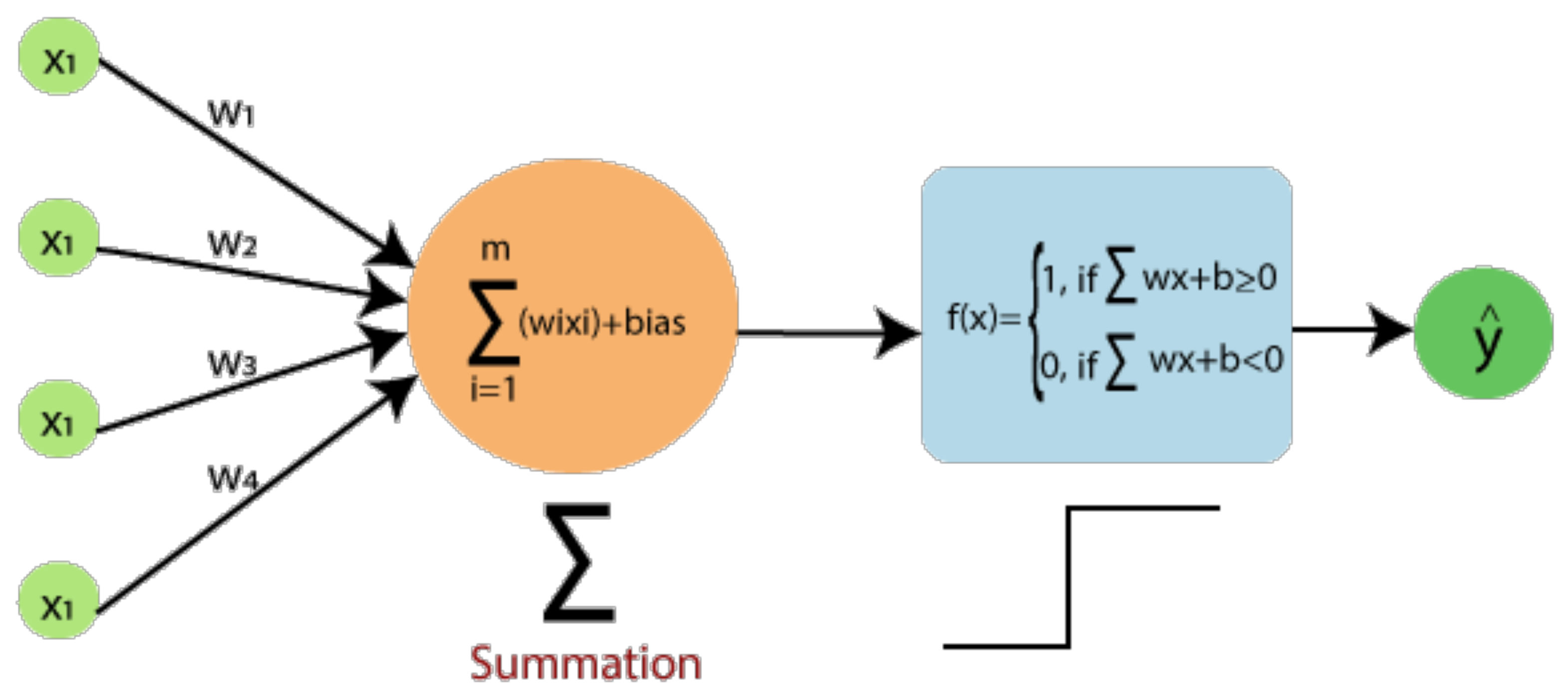


B

Classical Neural Networks

- Neural Networks are computing systems inspired by the biological neural networks that constitute animal brains
- They consist of multiple neurons, where the calculations happens. It takes inputs, and gives an output which is calculated by summing over the inputs and applying an activation function
- The network also contains weights, which are the parameters which are modified throughout the training process. The outputs from a layer are multiple by the weights and passed to the next layer as input.
- The network may have a backpropogation step which calculates the gradient of the cost function to give a better approximation of the optimal function. Some network just use the difference between the actual and the calculated output and use this difference to modify the weights.
- The networks also use a learning rate which controls the size of corrective steps.





Quantum Neural Network

Quantum perceptron

- Consider a quantum system with n inputs $|x_1\rangle, |x_2\rangle, \ldots, |x_n\rangle$
- The output from the perceptron is derived by, where F is the activation function

$$|y\rangle = F \sum_{i=1}^{n} w_i |x_i\rangle$$

- For each epoch t

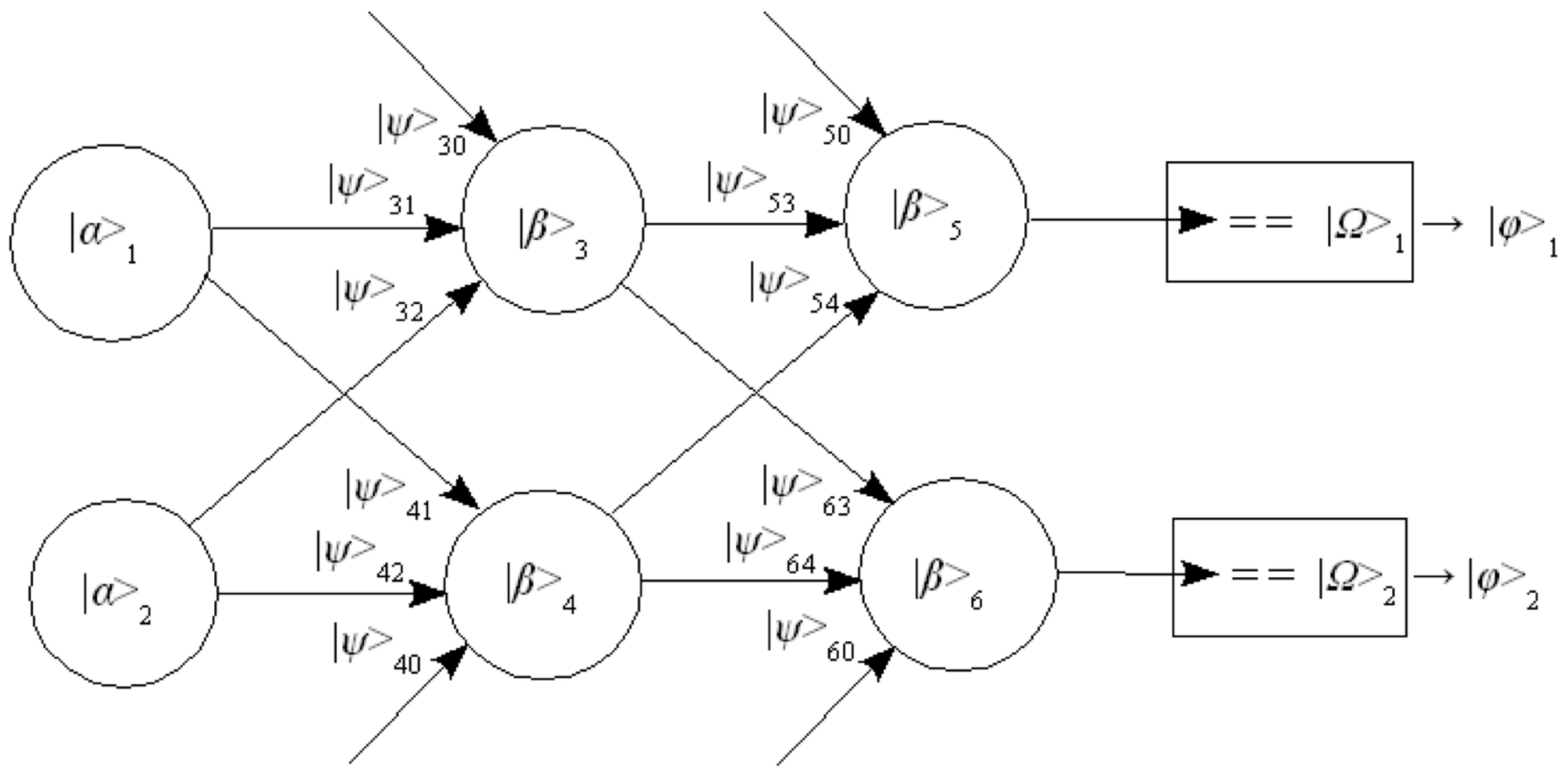
$$|y(t)\rangle = F \sum_{i=1}^{n} w_i(t) |x_i\rangle$$

- Update weights using the formula

$$w_i(t+1) = w_i(t) + \eta(|d\rangle - |y(t)\rangle)\langle x_i|$$

Quantum Network

- Input node i denoted by $|\alpha\rangle_i$
- Input for a data point is $|\alpha\rangle_{n1}, \ldots, |\alpha\rangle_{ni}$ where n is is the data point.
- Target outputs are denoted by $|\Omega\rangle_{n1}, \ldots, |\Omega\rangle_{ni}$
- Weights are denoted by $|\psi\rangle_i$
- The internal calculations are stored in the registers $|\beta\rangle_k$
- The network's ability to classify is stored in $|\phi\rangle_i$. The sum of all $|\phi\rangle_i$ is stored in $|\rho\rangle$
- Once all the inputs are processed, $|\rho\rangle$ should have a value between zero and the total number of input samples



Training the weights

- Need to find a solution which will give $|\rho\rangle = n * m$ where n is the number of samples and m is number of output nodes

- Put all possible weight vectors into a superposition $|\psi\rangle$ and initialise $|\beta\rangle, |\rho\rangle,$ and $|\phi\rangle$ to the state $|0\rangle$

- By superposition, each training sample is classified with respect to all the weight vectors simultaneously

- Use $|\rho\rangle$ as the oracle for quantum search, thus finding the weights which will classify the data correctly

- However to do this, the entire network will be un-computed, the registers will be unentangled and set back to their initial values. This will require the entire network to be re-computed for each epoch

- This can lead to underfitting or overfitting of the data.

- Underfitting such that there won't be any solution network which will correctly classify all training samples, so there would be equal chance of measuring the weight vector

- Overfitting as the weight vector measured will perfectly fit the data, but might fail at classifying new data

- This can be prevented by modifying the search oracle as $|\rho\rangle = n * m * p$ where p is some acceptable percentage of training sample to classify correctly

Questions?

Thank You