Quantum Machine Learning for Conspicuity Detection in Production

Task1:

Pennylane Tutorial Codebook:

The Pennylane tutorial codebook for Learning Quantum Computing consist of the basic theory and theory formulations via Pennylane code for Understanding the basics of Quantum Computation which lays as the foundation for Quantum Machine learning.

A) The Introductory part gives us an understanding about the Single Qubit representations, Qubit operations and Normalization of Quantum states.
B) The single-qubit state part gives idea about state manipulation such as state preparation, qubit rotation, single qubit gate operations etc.
C) The third part gives us understanding about multi qubit quantum circuits, entanglement , circuit manipulation for desired quantum state and a multi qubit gate challenge.

Task2:

Quantum Machine Learning:

Classical Machine Learning Focuses on Statistical Analysis of Data to Predict patterns which can be used to classify new data samples. Quantum Machine Learning is a hybrid approach. The combination of Quantum Circuits with Classical Machine Learning models.

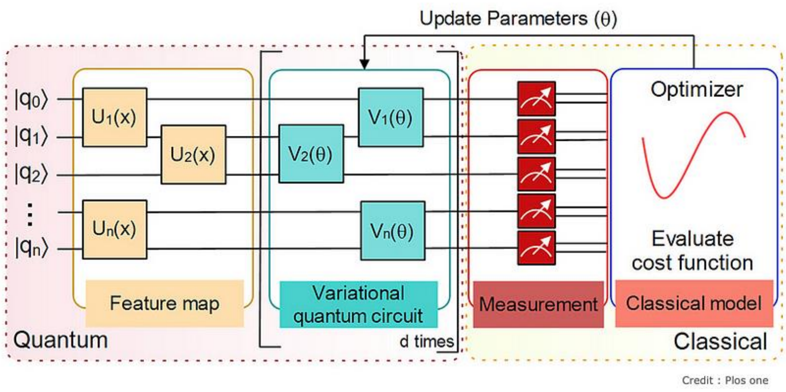
Concepts to Remember:

1) **Cost Function:** The cost function is generally an Optimization Parameter used in Classical as well as Quantum circuits. This cost function is Optimized according to an Optimization Strategy which will lead us towards the minimum value of the function. In a supervised learning model, the cost function is a combination of a loss function and a regularizer.
2) **Gradient:** Gradient points in the direction of the steepest Ascent. Basically, it acts as a useful constrain on the cost function. It gives us the idea to Move in a direction opposite to the Steep to get a lower value.
3) **Layer:** Layer forms the elementary building block for a VQ circuit which is replicated to form the entire circuit.
4) **QNode:** The parameterized quantum circuit in QML part is encoded as QNode in the QML model. We Optimize these parameters with the help of a classical ML model to minimize the cost function.
5) **Optimization:** Optimization is generally finding the best classical ML algorithm to optimize the parameters in VQC circuit. (Training using different models.)

Variational Quantum Classifiers:

VQC are quantum circuits that can be trained form labelled data to classify new data samples. The VQC consists of a sequence of parameter dependent unitary transformations which acts on an input quantum state.
The Input state is fed into the parameterized circuit and the resulting cost function is given to a classical optimizer which evaluates the cost function and give better parameter suggestions. We update the parameters according to the suggestions and repeat the process till maximum optimality is obtained.

Image from Medium Article



Our objective for the task is to represent Basis Encoding (Encoding the binary inputs into the basis states of Variational Circuits) and Amplitude Encoding (Real vectors as Amplitude vectors into quantum states).

Objective 1: Basis Encoding

Basis encoding shows us how to optimize Variational Quantum Circuit to emulate the Parity function. The parity function is a binary function defined form x to y where x is an n bit binary string such that

$$f : x \in \{0,1\}^{\otimes n} \rightarrow y = \begin{cases} 1 \text{ if uneven number of 1's in } x \\ 0 \text{ else.} \end{cases}$$

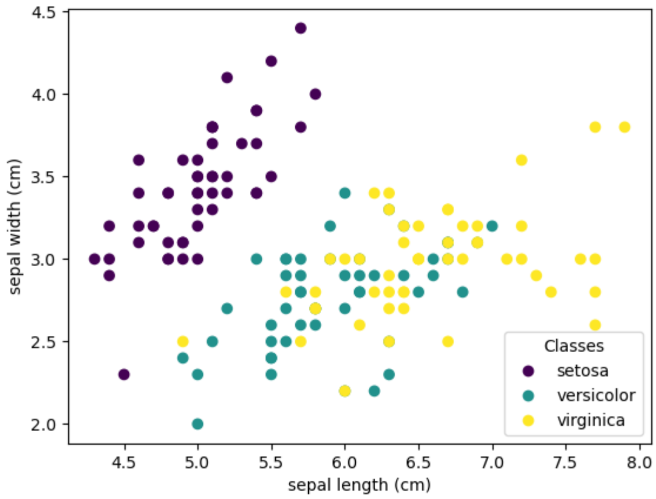
This is done via the standard steps followed in QML.

a) Importing the Necessary libraries and Optimizers. (Here the NesterovMomentumOptimizer is used which is a classical ML optimizer.)
b) Defining a Device – We need a device specified for running our quantum algorithm.
c) Defining Layer, Encoding the Classical bit string into quantum state and defining QNode.
d) Defining the cost function.
e) Optimization

Here the cost function is a standard square loss function which outputs the distance between the target labels and our model predictions.
We also define a function for getting the accuracy as a measure of the proportion of prediction that aggress with the actual label.

Objective 2: Amplitude Encoding (The IRIS Dataset Classification)

The Iris dataset consist of 3 different types of irises' (Sentosa, Versicolour, and Virginica) petal and sepal length, stored in a 150x4 NumPy array. The datapoints in the dataset are 2-dimensional vectors.



Here the state preparation is different from that of Basis Encoding. Each input data (2 dim vector) is converted to a set of angles which is fed into a routine for state preparation.
We need to update the layer function as we are working with two qubits and update the cost function as well.

Task 3: