Lab 2: Quantum Machine Learning on the Enron Dataset

**Objective:**

In this lab, we will explore the integration of classical data (from the Enron email dataset) with quantum machine learning (QML) techniques using PennyLane. Specifically, we will build a variational quantum classifier to classify email messages based on their categories.

**Quantum Machine Learning (QML):** QML combines the principles of quantum computing with machine learning. Variational quantum classifiers (VQC) are a popular method in QML, leveraging quantum circuits to optimize machine learning tasks. These classifiers can be particularly useful for binary classification, where the input data is encoded as quantum states, processed by quantum gates, and measured to yield predictions.

**PennyLane**: PennyLane is a Python library that supports the development of quantum algorithms using gradient-based optimization. It allows easy integration with existing machine learning frameworks and supports simulation of quantum circuits.

**Scenario:** In a large multinational corporation, the cybersecurity team faces constant threats from phishing emails designed to trick employees into sharing sensitive information. Traditional machine learning-based spam filters struggle to keep up with evolving tactics, leading the company to explore advanced solutions. They implement a variational quantum classifier (VQC) into their email filtering system, leveraging quantum machine learning (QML) to enhance detection capabilities. One day, a wave of sophisticated phishing attacks targets the finance department, disguised as internal emails. Thanks to the advanced QML filter, the system detects anomalies in message patterns, flagging the emails as potential threats before they reach employees’ inboxes. This proactive approach allows the cybersecurity team to mitigate risks and safeguard sensitive data, demonstrating the significant potential of quantum computing in defending against cyber threats.

Dataset: **Enron Dataset:** The Enron dataset is a large collection of emails generated by employees of the Enron Corporation, which became publicly available as part of the investigation following the company's collapse. This dataset is widely used in NLP tasks, such as spam classification, as it contains both spam and non-spam messages categorized into "ham" (non-spam) and "spam." https://www.kaggle.com/datasets/mohinurabdurahimova/maildataset

Tasks:

**Task 0: Installation of Pennylane**

We will review some sample examples below to help us become comfortable with the syntax needed to build circuits and simulate them in pennylane. Pennylane is a Python library that requires version 3.7 or higher to function.

1. Install Pennylane and Jupyter notebook

open google colab and do “pip install pennylane”

To check the installation of pennylane is working properly, type “ import pennylane as qml” in the jupyter notebook

**Task 1:** Quantum Circuit Design for Classification

In this task, you will design a **variational quantum circuit** that will be used for classifying emails in the Enron dataset. Variational quantum circuits are a hybrid approach that combines classical and quantum computing, enabling efficient training of quantum models for specific tasks like classification.

#### **Step 1: Implement a Variational Quantum Circuit**

Using the **PennyLane library**, you will create a quantum circuit that consists of several layers of quantum gates. A variational quantum circuit typically involves the following components:

* **Parameters:** The circuit parameters that will be optimized during training, often represented by angles for rotation gates.
* **Quantum Gates:** These gates manipulate the quantum state of qubits. Common gates include:
  + **Rotations (e.g., Rot):** Apply rotation around different axes on the Bloch sphere, helping to prepare the state based on input features.
  + **CNOT Gates:** Create entanglement between qubits, which is essential for capturing correlations in the data.

#### **Step 2: Define Quantum Layers and Gates**

* **State Preparation:** This is where you encode the classical input data into the quantum state. Use basis embedding or other methods to represent the input features in a way that can be processed by the quantum circuit.
* **Quantum Layers:** Define multiple layers of quantum gates to evolve the quantum state. The more layers you add, the more complex the circuit becomes, potentially improving the classifier's ability to learn from the data.
* **Circuit Function:** Create a function that will run the entire quantum circuit, applying the defined layers and returning the measurement outcomes, which will be used for classification.

|  |
| --- |
| Code 1-4 |

**Task 2: Data Preprocessing**

In this task, you will work with the **Enron dataset**, which contains a collection of emails from Enron Corporation executives. This dataset is particularly useful for exploring email classification tasks, such as spam detection or categorization based on content.

#### **Step 1: Load the Enron Dataset**

Begin by loading the Enron dataset using pandas. Explore the dataset’s structure to understand the columns, data types, and general content of the emails. This initial exploration helps you identify relevant features for classification.

#### **Step 2: Preprocess the Email Data**

Since machine learning algorithms, including quantum classifiers, require numerical inputs, you need to convert the email text into numerical features. Common techniques include:

* **TF-IDF Vectorization:** Transform the email content into a matrix of TF-IDF features, which represent the importance of words in the context of the entire dataset.
* **Bag-of-Words:** An alternative approach that counts the frequency of each word in the text.

Choose the method that best suits your classification needs.

#### **Step 3: Encode Labels for Quantum Classification**

For quantum classification, you will need to encode the target labels. If your classification task is binary (e.g., spam vs. ham), map the labels to numerical values:

* **Spam:** Encode as 1
* **Ham (not spam):** Encode as 0

|  |
| --- |
| Code 5-9 |

Show a screen shot of the email distrubution

|  |
| --- |
|  |

**Task 3: Training the Variational Classifier**

In this task, you will train a variational quantum classifier designed to classify emails in the Enron dataset. The training process involves preparing the data, initializing the model, and optimizing the parameters to improve classification performance.

#### **Step 1: Split the Data into Training and Testing Sets**

* **Data Preparation:** First, load and preprocess the Enron dataset. Ensure that the email messages are converted into numerical features (e.g., using TF-IDF).
* **Train-Test Split:** Use a method like train\_test\_split from the sklearn library to divide the data into training and testing sets. This helps in evaluating the classifier's performance on unseen data. A common practice is to allocate 70-80% of the data for training and the remaining 20-30% for testing.

#### **Step 2: Initialize the Quantum Classifier and Set Up an Optimizer**

* **Model Initialization:** Define the variational quantum circuit using the PennyLane library. Set the number of qubits and quantum layers based on your earlier design.
* **Optimizer Setup:** Choose an optimizer (e.g., AdamOptimizer) to adjust the model parameters during training. This optimizer will help minimize the loss function by updating the weights and bias based on the computed gradients.

#### **Step 3: Train the Quantum Classifier and Monitor Performance**

* **Training Loop:** Implement a loop to iteratively train the model for a specified number of epochs or iterations. In each iteration:
  + Select a batch of data from the training set.
  + Compute the predictions of the quantum classifier for the current batch.
  + Calculate the loss based on the predictions and actual labels.
  + Update the model parameters using the optimizer.
* **Performance Monitoring:** After each iteration, monitor and log the classifier's performance metrics, such as accuracy and loss. This will help in understanding how well the model is learning over time and whether any adjustments are needed in terms of hyperparameters or model structure.

|  |
| --- |
| Code 10-13 |

**Task 4: Evaluating the Quantum Classifier**

In this task, you will assess the performance of the trained variational quantum classifier using various evaluation metrics. This evaluation will help determine how well the classifier performs on the test data and its effectiveness in distinguishing between different categories of emails (e.g., spam vs. ham).

You will evaluate the performance of the trained classifier:

Use accuracy, precision, recall, and F1 score as evaluation metrics.

Analyze the classifier's predictions on the test data and compare them with the actual labels.

|  |
| --- |
| Code 14 |

Show the result of the evaluation

|  |
| --- |
|  |

Question:

Here are three simple questions for the lab on Quantum Machine Learning using the Enron dataset:

1. **What is the primary objective of using a variational quantum classifier (VQC) in the context of the Enron dataset?**

|  |
| --- |
| The primary objective of using a variational quantum classifier (VQC) with the Enron dataset is to enhance the classification of emails as either spam or non-spam (ham) by leveraging quantum machine learning techniques. The VQC aims to improve detection accuracy and efficiency in identifying sophisticated phishing attempts that traditional machine learning models might miss. |

1. **Which features from the Enron dataset are important for preprocessing before feeding them into the quantum classifier?**

|  |
| --- |
| Important features for preprocessing include email content (text), metadata (such as sender and recipient information), and specific derived attributes like the presence of certain keywords or patterns indicative of phishing. Additionally, any categorical features (e.g., email type) should be encoded numerically for input into the quantum classifier. |

1. **What metrics will be used to evaluate the performance of the quantum classifier after training, and why are they important?**

|  |
| --- |
| The metrics used to evaluate the performance of the quantum classifier will include accuracy, precision, recall, and F1 score. These metrics are important because they provide a comprehensive understanding of the classifier's performance. Accuracy indicates overall correctness, precision measures the proportion of true positives among predicted positives, recall evaluates the model's ability to identify actual positive instances, and the F1 score balances precision and recall, making it especially useful in scenarios with imbalanced classes, like spam detection. |