

**AI Research Intern**

**Task-1 Report**

Dept. of Quantum Algorithm

Submitted by: Basa Gnana Chandrika

Date: 19/05/2025

In Cooperation with

Anantwave Pvt ltd

Table of Contents

[1 Introduction 3](#_Toc198557725)

[1.1 Objective: 3](#_Toc198557726)

[2 Method of Solving 3](#_Toc198557727)

[3 About Dataset 4](#_Toc198557728)

[4 Code 5](#_Toc198557729)

[4.1 Load Dataset: 5](#_Toc198557730)

[4.2 Classical Extraction: PCA 6](#_Toc198557731)

[4.3 Quantum Encoding: Amplitude Encoding Circuit 6](#_Toc198557732)

[4.4 Quantum Algorithm: Quantum Kernel Matrix 7](#_Toc198557733)

[4.5 Optimization: Balance Between Dimensionality and Feature Preservation 7](#_Toc198557734)

[4.6 Training & Evaluation 8](#_Toc198557735)

[4.7 Loss Curve & Latency Simulation 9](#_Toc198557736)

[5 Output 10](#_Toc198557737)

[5.1 Description of the Results Obtained 10](#_Toc198557738)

[6 Conclusion: 13](#_Toc198557739)

[7 What I have Learned: 13](#_Toc198557740)

**Task**: **Advanced Algorithm-Based Quantum Data Encoding with Quantum Algorithms.**

# Introduction

## Objective:

The objective of this project is to combine classical and quantum methods to classify images more effectively. It uses PCA to reduce the size of the image data, then converts that data into quantum states using amplitude encoding. A quantum kernel is created from these states and used by a classical SVM to improve classification accuracy. The project uses PennyLane for quantum tasks and Scikit-learn for classical processing, applied to an ADAS image dataset.

# Method of Solving

To address the challenge of classifying ADAS images using quantum computing, a hybrid classical-quantum machine learning approach is adopted. The pipeline begins with the collection and preprocessing of image data. Images are resized to a uniform dimension of 32x32 pixels, converted to grayscale to reduce complexity, and then flattened into one-dimensional arrays for processing.

Next, Principal Component Analysis (PCA) is used to extract relevant features while reducing the high-dimensional space of image data to a manageable size. PCA is a statistical technique that identifies the directions (principal components) in which the data varies the most, allowing the dataset to be represented with fewer variables while preserving essential information. In this implementation, the number of components is fixed at 8.

These reduced feature vectors are then encoded into quantum states using amplitude encoding. Amplitude encoding is efficient as it encodes the information into the amplitudes of quantum basis states, minimizing qubit usage and allowing for high-dimensional data representation.

Quantum Kernel Estimation (QKE) is performed using quantum circuits. These circuits compare quantum states of input data samples and measure their fidelity (overlap), constructing a quantum kernel matrix. This matrix captures nonlinear relationships among data points in a high-dimensional Hilbert space.

A classical Support Vector Machine (SVM) classifier uses the quantum kernel matrix to learn decision boundaries between classes. Evaluation is done using accuracy, F1-score, AUROC, and confusion matrix to assess classification performance.

Optimization is incorporated by balancing the dimensionality reduction (via PCA) and the number of qubits required for amplitude encoding. By carefully choosing the number of PCA components, the system ensures optimal resource usage while retaining important features for classification.

# About Dataset

The dataset used in this project is the **ADAS (Advanced Driver Assistant Systems)** dataset, designed for **object detection tasks** related to road safety and driver assistance. It consists of:

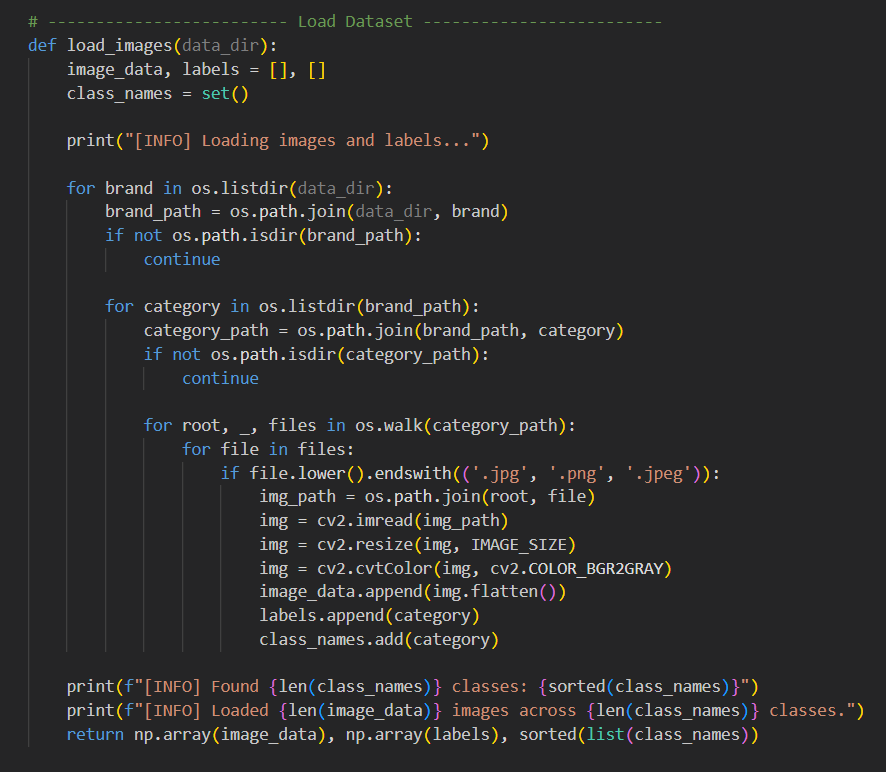
* 903 images
* 1143 labelled objects
* 9 object classes, including:
* Pothole
* Left-hand curve
* Right-hand curve
* Speed Breaker
* Bridge ahead
* Pedestrian
* Animal
* Name Board
* Vehicle

Key Features:

* Variability: Images were captured in a wide range of real-world scenarios:
* Different road types: highways, village roads, city streets.
* Various environmental conditions: sunny, rainy, foggy, cloudy, night-time.
* Different angles and distances for realistic object coverage.
* **Device Diversity:** Captured using **two smartphones** – *iPhone 12* and *VIVO Y51A* – to ensure device-based variance for robust evaluation.
* **Annotations:** Bounding box labels were manually created using the **LabelMe** tool.
* **Tags:** Each image may contain additional metadata such as id, camera, and category. Some also include road\_type.

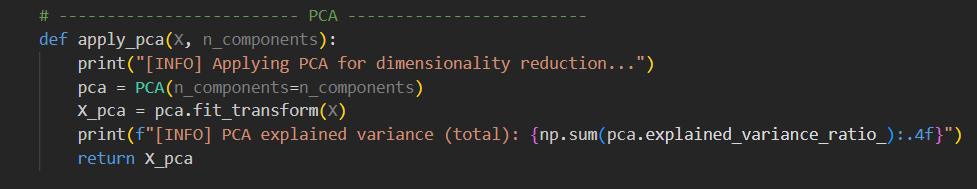
# Code

## Load Dataset:



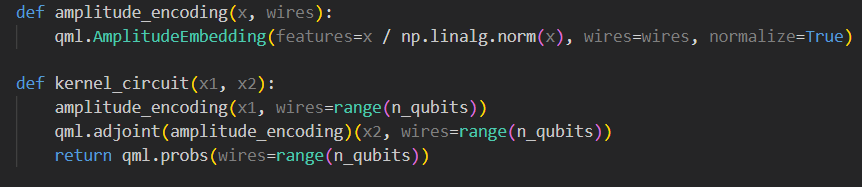
* Function: load\_images(data\_dir)
* Input: Path to the root image dataset directory.
* Output: Numpy arrays of image data, labels, and list of class names.
* Method: This function traverses a nested folder structure of image files, resizes each image to 32x32 pixels, converts it to grayscale, and flattens it into a vector. Labels are extracted from folder names. This uniform representation ensures compatibility with subsequent PCA transformation.

## Classical Extraction: PCA



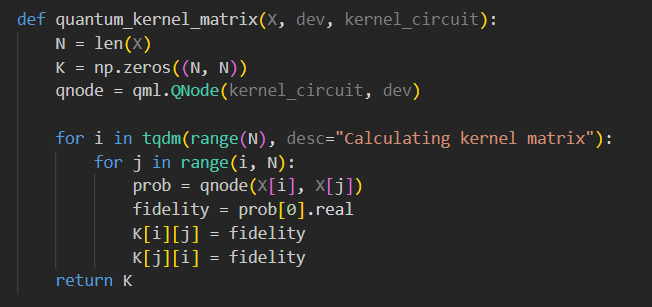
* Function: apply\_pca(X, n\_components)
* Input: Flattened image vectors, number of principal components.
* Output: PCA-transformed data matrix.
* Method: PCA reduces the input image data to 8 principal components. This step ensures that only the most informative features are retained, reducing computational load for quantum encoding.

## Quantum Encoding: Amplitude Encoding Circuit



* Function: amplitude\_encoding(x, wires)
* Input: Feature vector x of reduced dimensions (from PCA), qubit wire list.
* Output: Quantum state with data encoded into amplitude values.
* Method: This method normalizes the input vector and uses PennyLane's AmplitudeEmbedding to encode the vector onto a quantum circuit. This approach uses fewer qubits than basis or angle encoding methods.

## Quantum Algorithm: Quantum Kernel Matrix



* Function: quantum\_kernel\_matrix(X, dev, kernel\_circuit)
* Input: Dataset X, quantum device dev, and kernel circuit.
* Output: Symmetric quantum kernel matrix.
* Method: This function iterates over all sample pairs, computes their quantum fidelity using a QNode, and stores results in a symmetric matrix for use in SVM.

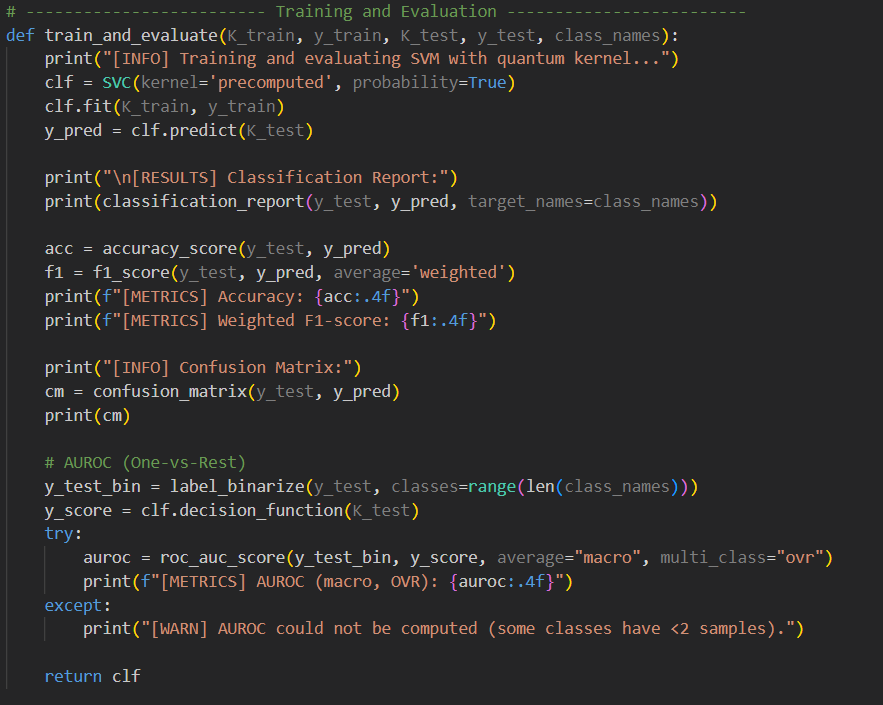
## Optimization: Balance Between Dimensionality and Feature Preservation





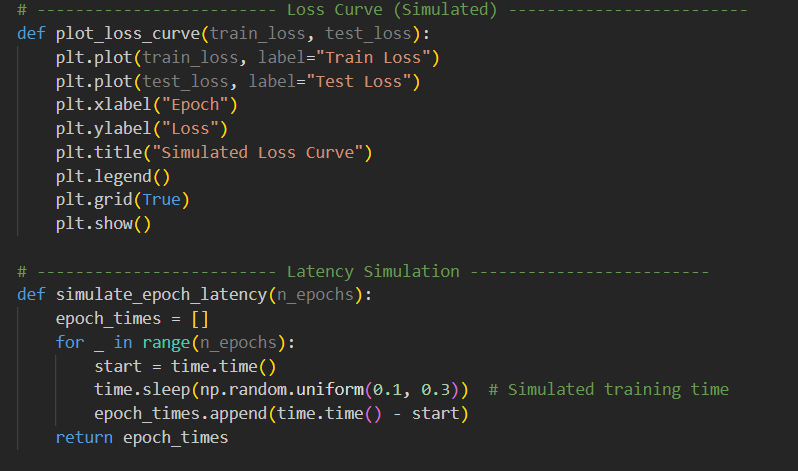
* Explanation: The number of PCA components (features) is fixed to 8 to ensure high variance retention and a manageable quantum encoding dimension. The number of qubits is derived from the number of features via logarithmic scaling, optimizing circuit complexity and resource allocation.

## Training & Evaluation



* Function: train\_and\_evaluate(K\_train, y\_train, K\_test, y\_test, class\_names)
* Input: Quantum kernel matrices and encoded class labels.
* Output: Trained SVM model and printed performance metrics.
* Method: An SVM classifier is trained using the precomputed kernel. Evaluation metrics such as accuracy, F1-score, confusion matrix, and AUROC are computed to assess performance.

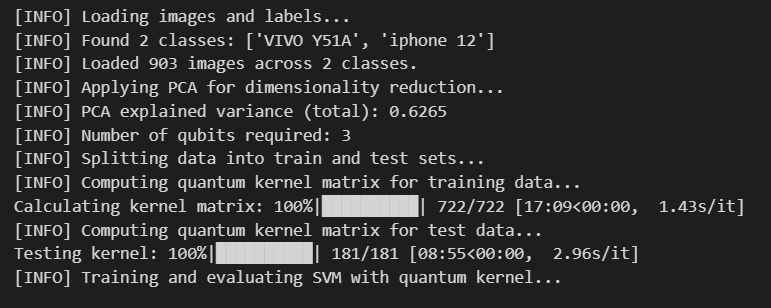
## Loss Curve & Latency Simulation



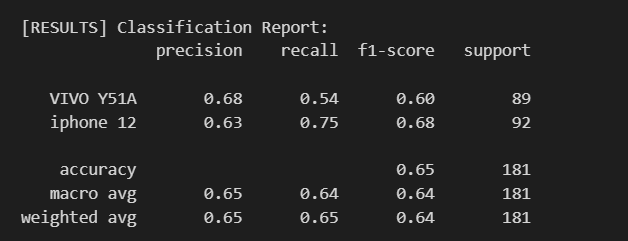
* Function: plot\_loss\_curve(train\_loss, test\_loss)
* Input: Simulated training and testing loss values.
* Output: Line plot of loss over epochs.
* Method: This function visualizes the simulated behavior of model performance over training epochs.
* Function: simulate\_epoch\_latency(n\_epochs)
* Input: Number of epochs.
* Output: List of simulated latency per epoch.
* Method: Uses random sleep durations to mimic quantum computation delays.

# Output

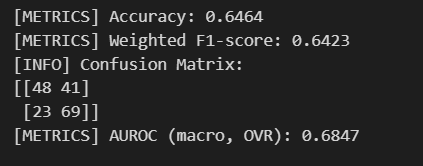
## Description of the Results Obtained



* The output confirms that the dataset was successfully loaded and processed.
* Total images: 903, distributed across 2 distinct classes — ‘VIVO Y51A’ and ‘iPhone 12’.
* Dimensionality reduction via PCA retained 62.65% of the explained variance and reduced input to 3 qubits, optimizing it for quantum processing.
* The dataset was split into training and testing subsets, with a final test set containing 181 samples.
* Each of the 722 training sample pairs was processed through a quantum circuit to compute similarity scores, forming a quantum kernel matrix that is then used by the SVM classifier to learn decision boundaries in the quantum feature space.
* Each of the 181 test sample pairs was processed through a quantum circuit to compute similarity scores, forming a quantum kernel matrix that is then used by the SVM classifier to learn decision boundaries in the quantum feature space.



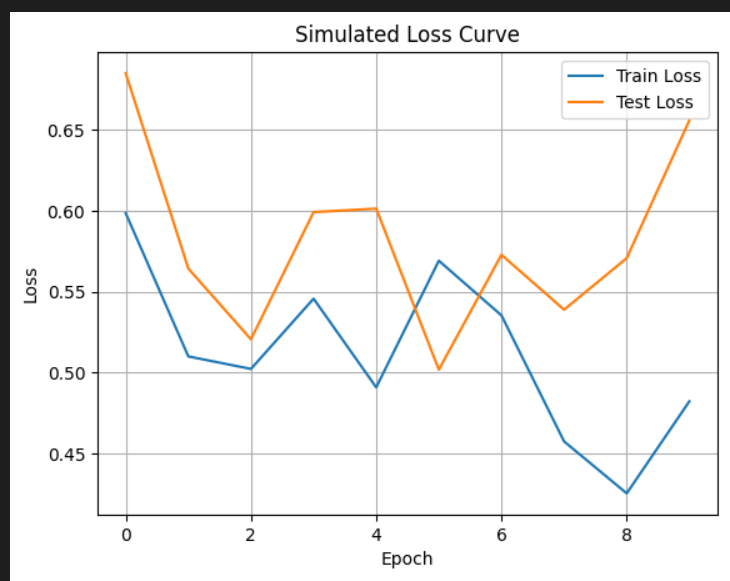
* The model’s performance on classifying **VIVO Y51A** and **iphone 12** is moderate:
* **Precision:** 68% (VIVO Y51A), 63% (iphone 12) — accuracy of positive predictions.
* **Recall:** 54% (VIVO Y51A), 75% (iphone 12) — ability to find true samples.
* **F1-score:** 0.60 (VIVO Y51A), 0.68 (iphone 12) — balance between precision and recall.
* **Accuracy:** Overall 65% of samples correctly classified.
* **Averages:** Macro and weighted averages are about 0.64, reflecting consistent performance across classes.



* Overall Accuracy: 64.6%, indicating that the quantum kernel model correctly classified about two-thirds of the test samples.
* Class-wise F1-scores:
* VIVO Y51A: 0.60
* iPhone 12: 0.68
* This demonstrates that the model performed slightly better in identifying ‘iPhone 12’ images compared to ‘VIVO Y51A’.
* AUROC Score: 0.6847, suggesting moderate ability of the model to distinguish between the two classes.
* Confusion Matrix:
* 48 samples were correctly predicted as ‘VIVO Y51A’, while 41 were misclassified.
* 69 samples were correctly identified as ‘iPhone 12’, while 23 were incorrectly labeled.



* The average epoch latency was 0.2008 seconds, reflecting the computational time needed per epoch when using quantum kernels. This is acceptable for a hybrid quantum-classical system and indicates the feasibility of deploying such models in practice.



* The loss curve shows the model’s learning behavior over 10 epochs:
* Training Loss gradually declined with minor fluctuations, suggesting the model was learning useful features.
* Test Loss showed higher variability, particularly in later epochs, which could indicate minor overfitting or sensitivity to test samples.
* Despite fluctuations, both curves remained relatively close, showing general stability.

# Conclusion:

The quantum kernel-based classification approach was successfully applied to distinguish between images of 'VIVO Y51A' and 'iPhone 12' using PCA-reduced features. The model achieved a 64.6% accuracy and an AUROC of 0.6847, indicating decent performance using quantum techniques. Although the test loss showed some fluctuations, the model was able to generalize fairly. The results show that quantum machine learning, even with a small number of qubits, can be a practical alternative for image classification problems when classical resources are limited.

# What I have Learned:

* Encoding classical data into quantum states using Amplitude-Encoding.
* Applying PCA for dimensionality reduction.
* Computing Quantum Kernel Matrices and using them with SVM for classification.
* Evaluate Performance using metrices like accuracy, F1-score, AUROC and loss curves.
* Beginner Experience with PennyLane and Quantum Machine Learning concepts.