Methodology

Hypothesis

Quantum Kernel Alignment is a fascinating tool for Quantum Kernel Methods which adjusts the kernel according to the specific dataset to improve the classification accuracy. Driven by the need of NISQ compatible quantum classification algorithm that can handle the industrially relevant data scale, there are different methods developed to reduce the circuit executions required for the quantum kernel alignment. [ref literature review]

Traditionally, Kernel matrix required for calculating the alignment between the classes with n training samples is n ^2. As discussed before there are various approaches that have been developed with linear and quadratic complexity.

Subsampling approach aligns the kernel with O (k ^2) when k subsamples are selected. This approach also scales quadratically as the traditional algorithm with reduced number of circuit executions. Further the development of QUACK algorithm for training the kernel based on the centroids of classes gives the linear complexity of O(n).

A graph showing the number of individuals in training

Description automatically generated

[Caption, The figure illustrates the number of circuit executions required for a single iteration of kernel alignment as a function of the number of training samples (n). It compares three methods: Full Kernel (O (n ^2)), Subsampling (n/8) samples, resulting in O ((n/8) ^2), and QUACK O(n/2). The logarithmic scale on the y-axis emphasizes the efficiency improvements of Subsampling and QUACK over the Full Kernel method.]

To improve computational complexity further, we propose a method that divides the given data into classes and computes centroids at two levels:

1. **Main Centroids**: Represent the overall mean of each class.
2. **Sub-Centroids**: Represent the centroids of smaller clusters within each class.

This hierarchical centroid calculation should reduce the data representation to a set of key points, making kernel calculations more efficient while preserving class structure and reducing circuit execution complexity.

A screenshot of a video game

Description automatically generated

[Half Figure half page algorithm]

The proposed method retains the overall data structure by utilizing sub-cluster centroids. During the kernel alignment process, the alignment is performed in a way that preserves the structure of these centroids, ensuring that no significant information about the class is lost, as can happen with subsampling approaches. By representing more information with fewer data points, the circuit executions are expected to be reduced to linear complexity, achieving lower overhead compared to the QUACK algorithm. Additionally, since the structure of the dataset is preserved, the classification accuracy should remain comparable to that of other existing algorithms.

To validate this hypothesis, we conduct a series of experiments using multiple datasets and various training hyperparameters. These experiments are designed to systematically test the effectiveness and robustness of the proposed approach under different conditions.

A diagram of a diagram

Description automatically generated

[Figure: A mind map illustrating various approaches and techniques related to Quantum Kernel Alignment. It highlights alignment techniques such as Random Subsampling, QUACK, Full Kernel, and the proposed Centroid-Based Kernel Alignment. Additionally, it showcases encoding methods, including Input Scaling, Data Reuploading, and Architectural considerations, along with a comparison to Classical SVM using an RBF Kernel.]

**Data Preparation**

Choosing meaning full datasets for general benchmarking studies is difficult. To validate the proposed methodology, we prepared diverse datasets to simulate various data distributions and complexities. The datasets include both synthetic and real-world data, ensuring comprehensive evaluation.

Synthetic Datasets:

Artificially generated datasets were use predominantly as it allows to vary the properties of dataset. While using artificially generated dataset may limit the “real world” applicability of the model, it provides the firm base to test and validate the model in initial stages of research. Three datasets were created, each varying in complexity, structure, and number of features. Collectively, they represent non-linear, overlapping, and geometrically complex data distributions.

Checkerboard Dataset:

A graph with blue and orange circles

Description automatically generated

The checkerboard dataset is used for testing out methods sensitivity towards non-linearity . The dataset consists of 30 training and 30 testing datapoints and were generated as follows, The dataset was generated in the domain [0, 1]² with a 4 x 4 grid of sites. Points were sampled uniformly centered about each grid site to prevent overlap between centroids. Alternating classes were assigned to the sites, with training points getting the class corresponding to their centroid.

*Input: grid\_size, sampling\_radius, dmin, dmax  
Output: X (data points), y (labels)  
  
Initialize cords = []  
For i = 0 to grid\_size - 1:  
 For j = 0 to grid\_size - 1:  
 x = (2 \* i + 1) / (2 \* grid\_size)  
 y = (2 \* j + 1) / (2 \* grid\_size)  
 Append (x, y) to cords  
Initialize points = [], labels = []  
Set cluster = 0  
For each (cx, cy) in cords:  
 label = 1 if cluster == 1 else -1  
 For \_ = 1 to random(1, 10):  
 angle = random(0, 2 \* π)  
 radius = random(0, sampling\_radius)  
 x = cx + radius \* cos(angle)  
 y = cy + radius \* sin(angle)  
 Clip x, y to [dmin, dmax]  
 Append (x, y) to points  
 Append label to labels  
 Toggle cluster (0 to 1 or 1 to 0)  
Return points, label*

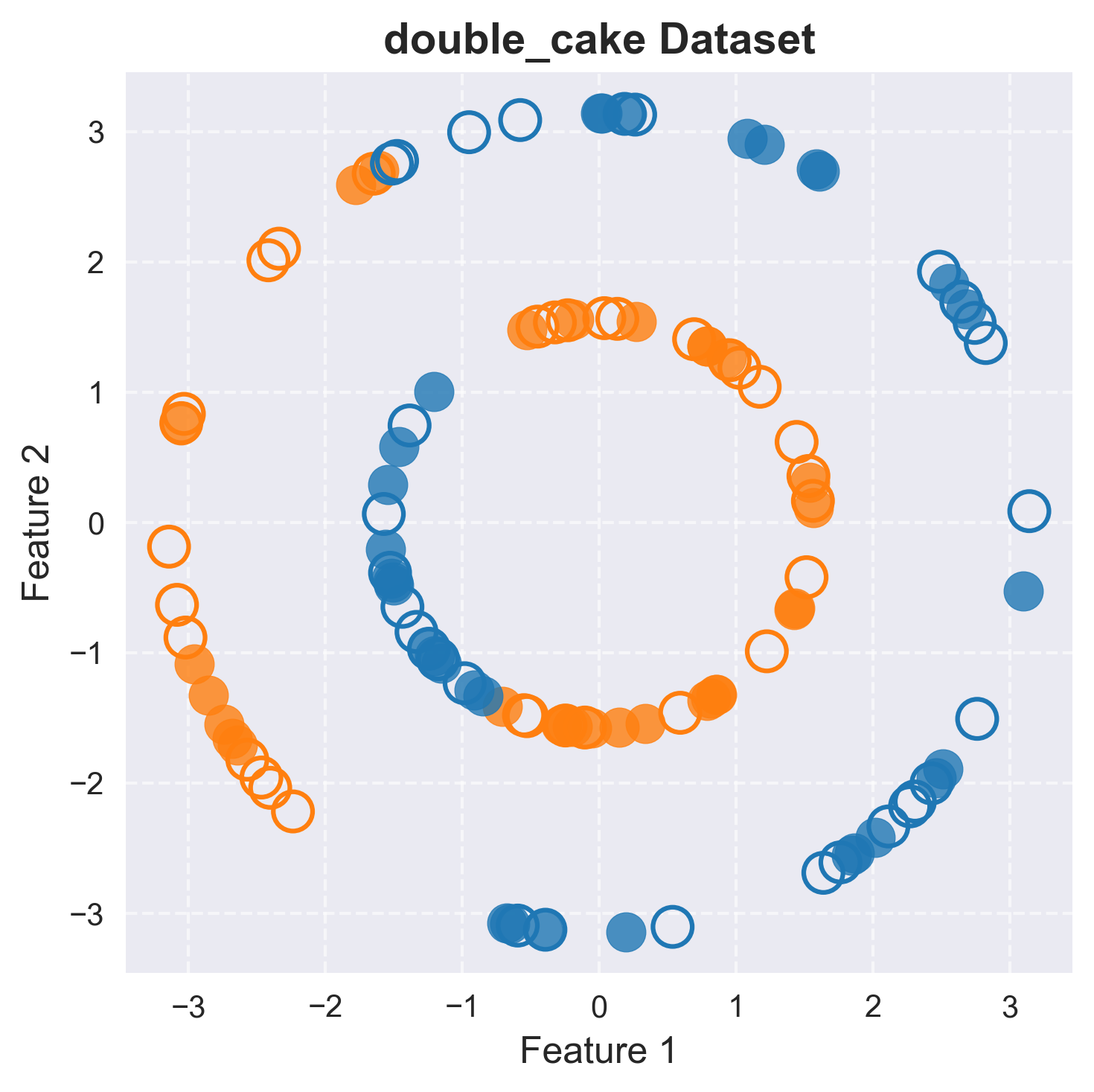
Corners Dataset:

A diagram of blue and orange circles

Description automatically generated

The corners dataset is used for testing out methods sensitivity towards overlapping and noise along with capability of identifying the clusters in the dataset. The dataset consists of 50 training and 50 testing datapoints.

Double Cake Dataset:



The **Double Cake Dataset** was selected to test quantum kernels and benchmark models in scenarios with non-linearities. Its geometric structure makes it ideal for challenging decision-making and classification tasks. The dataset was generated in a circular domain, where data points are arranged in two concentric circular layers (cakes). Each layer is divided into evenly spaced sectors. Points were sampled uniformly within each sector, and alternating classes were assigned to the sectors. The outer layer represents the "outer cake," while the inner layer, scaled down by a factor of 0.5, represents the "inner cake."  
*Input: num\_sectors, points\_per\_sector, radius\_outer, radius\_inner  
Output: X (data points), Y (labels)  
  
Initialize points\_outer = [], labels\_outer = []  
Initialize points\_inner = [], labels\_inner = []  
# Generate outer circle (Outer Cake)  
For sector in range(0, num\_sectors):  
 For \_ in range(0, points\_per\_sector):  
 angle = random(0, 2 \* π) + sector \* (2 \* π / num\_sectors)  
 x\_outer = radius\_outer \* cos(angle)  
 y\_outer = radius\_outer \* sin(angle)  
 label\_outer = 1 if sector % 2 == 0 else -1  
 Append (x\_outer, y\_outer) to points\_outer  
 Append label\_outer to labels\_outer  
# Generate inner circle (Inner Cake)  
For sector in range(0, num\_sectors):  
 For \_ in range(0, points\_per\_sector):  
 angle = random(0, 2 \* π) + sector \* (2 \* π / num\_sectors)  
 x\_inner = radius\_inner \* cos(angle)  
 y\_inner = radius\_inner \* sin(angle)  
 label\_inner = -1 if sector % 2 == 0 else 1  
 Append (x\_inner, y\_inner) to points\_inner  
 Append label\_inner to labels\_inner  
# Combine data  
X = points\_outer + points\_inner  
Y = labels\_outer + labels\_inner  
Return X, Y*

**Data Preprocessing**

Quantum Data embeddings are sensitive to the range on which the input data is confined. There were no preprocessing strategies implemented for models unless and until specified in respective studies. If no preprocessing strategies were defined, the input data was scaled to the natural range. For example for an angle embedding the angles were scaled to lie in the range of [-pi/2 , pi/2].

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Dataset | Training Samples | Testing Samples | Features | Fundamental Usage |
| Checkerboard | 30 | 30 | 2 | Tests sensitivity to non-linear interactions |
| Corners | 50 | 50 | 2 | Tests noise tolerance and clustering efficiency |
| Double Cake | 60 | 60 | 2 | Tests geometric complexity and radial variance handling |

**Implementation**

**Real World Datasets**

Downscaled Mnist Fashion: PCA   
n\_components = varied

Powergrid faults Dataset:

This dataset provides labeled data for fault detection and classification in a power system transmission line. The power system, modeled in MATLAB, consists of four 11kV generators connected via transformers to simulate and study faults at the midpoint of the transmission line.

The dataset contains approximately 120 data points which were divided into 60 training samples and 60 testing samples, including:

* **Line Voltages** and **Line Currents** measured at the output of the power system.

Includes six key measurements:

* Phase currents: Ia, Ib, Ic
* Phase voltages: Va, Vb, Vc
* Labels indicating normal operating conditions or fault across the three phases of the system.

Binary labels indicating the presence of a fault:

* 0: Normal operating conditions
* 1: Fault condition

@misc{Sathyaprakash2023, author = {E. Sathyaprakash}, title = {Electrical Fault Detection and Classification Dataset}, year = {2023}, publisher = {Kaggle}, url = {https://www.kaggle.com/datasets/esathyaprakash/electrical-fault-detection-and-classification/data}, note = {Accessed: [Your Access Date]} }

**Dataset Preprocessing**