MASTERS DATA SCIENCE PROJECT

AI-Enhanced Clustering: Comparative Analysis of Unsupervised Learning on Multimodal Datasets

by
Suhaib Mahamoud

Supervisor: John Woodward

October 14th, 2024

Table of Contents

Abstract *4*

I. Introduction & Literature review *5*

1.1 Project background **5**

1.2 Main Problems **5**
1.2.1 Parameter Sensitivity: 5
1.2.2 Scalability Issues: 5
1.2.3 Dataset Variability and Generalization: 5
1.2.4 Outlier Detection Challenges: 6
1.2.5 Dimensionality Reduction and Data Complexity: 6
1.2.6 Model Interpretability and Performance Evaluation: 6
1.2.7 Class imbalance 6

1.3 Aims and Objectives **7**

II. Methodology & Design *7*

2.1 Data Preprocessing **7**
2.1.1 Iris Dataset 7
2.1.2 Image Dataset 7

2.2 Dimensionality Reduction **8**

2.3 Outlier Detection **8**

2.4 Clustering Algorithms **9**
2.4.1 K-Means Clustering 9
2.4.2 Agglomerative Hierarchical Clustering (AHC) 9
2.4.3 BIRCH Clustering 9

2.5 Performance Evaluation **9**
2.5.1 Quantitative Metrics 9
2.5.2 Qualitative Visualisations: 10

2.6 Tools and Libraries **10**

2.7 Design Mind Map **10**

III. Experiments, Evaluation, Results & Discussion *10*

3.1 Image Dataset Results **10**
3.1.1 Initial Results from VGG16 Feature Extraction 10
3.2 K-means(Image Dataset) 11
3.2.1 K-means Clustering with PCA 11
3.2.2 PCA and Contamination 11
3.3 AHC 13
3.3.1 AHC Initial Results 13
3.3.2 Linkage methods & impact on Clustering performance 13
3.3.3 Introduction of PCA & Contamination 14
3.4 BIRCH 16
3.4.1 Initial Results BIRCH 16
3.4.2 Incoperation of Threshold, Branching, PCA and Contamination 16

3.6 K-Means Iris Dataset **17**
3.6.1 K-means initial results 17
3.6.2 PCA introduction & contamination 17

3.7 AHC Iris Dataset **18**
3.7.1 AHC initial results 18
3.7.2 Linkage methods, PCA & Contamination 19

3.8 Birch Iris Dataset **20**
3.8.1 Initial Birch Results 20
3.8.2 PCA & Contamination 20

3.9 Performance Evaulation/ Statistical Analysis **21**
3.9.1 K-means 21
3.9.2 AHC 22
3.9.3 BIRCH 24

3.10 Evaluation **25**

IV. Conclusion & Future works *26*

4.1 Conclusion on dataset performance & AI techniques **26**

4.2 Future works **26**

V. Declaration *26*

References *27*

Appendices *28*

Appendix A **28**

Appendix B- **28**

Table of Figures

Figure 1- Class height for Image Dataset(Training) 6
Figure 2- Class Width for Image Dataset (Training) 7
Figure 3- Class Height for Image Dataset(Testing/ Validation) 7
Figure 4- Class Width for Image Dataset (Testing/ Validation) 7
Figure 5- Training Images with labels (Image Dataset) 8
Figure 6- Training images without labels(Image Dataset) 8
Figure 7- Brief Mind map of Methodology and Design 10
Figure 8- Initial K-means Cluster scores 11
Figure 9- Graph showing PCA 2 vs 3 K-means 11
Figure 10- Various Configurations scores for K-Means Image dataset 12
Figure 11- Initial clustering from initial Results of K-means (Training) 12
Figure 12- Initial clustering from initial Results of K-means (Validation) 12
Figure 13- Best Configuration clustering K-Means (Training) 12
Figure 14- Best Configurations Clustering K-Means (Validation) 12
Figure 15- Worst Configuration Clustering K-Means (Training) 13
Figure 16- Worst Configuration Clustering K-Means (Validation) 13
Figure 17- Initial Silhouette Score Graph 13
Figure 18- Differing linkage methods 14
Figure 19- graph showing the best silhouette scores per cluster for each linkage 14
Figure 20- Initial Clustering for Training 15
Figure 21- Initial Clustering for Validation 15
Figure 22- Best Configuration for AHC (Training) 15
Figure 23- Best Configuration for AHC (Validation) 15
Figure 24- Worst Configuration for AHC (Training) 15
Figure 25- Worst Configuration for AHC (Validation) 15
Figure 26- Graph showing the best score for each cluster per contamination level 16
Figure 27- Best clustering for BIRCH (Training) 16
Figure 28- Best Clustering for BIRCH (Validation) 17
Figure 29- Initial Clustering scores for K-Means 17
Figure 30- best score for both PCA's K-Means 18
Figure 31- Best and worst Clustering for K-Means 18
Figure 32- Scores for 0.05 contamination per linkage 19
Figure 33- Scores for 0.1 contamination per linkage 19
Figure 34- Scores for 0.2 contamination per linkage 19
Figure 35- Differing clustering images for the same configuration for 3 Clusters. 19
Figure 36- Small exert of the results from differing configurations prior to PCA & contamination 20
Figure 37- Difference in scores per contamination level 20
Figure 38- Best cluster silhouette score clustering 21
Figure 39- Worst Cluster Silhouette score clustering 21
Figure 40- Silhouette Distribution graph for best clustering Iris (K-Means) 22
Figure 41- Silhouette Distribution graphs for Worst clustering Iris (K-Means) 22
Figure 42- Silhouette Distribution Graph for best Training clustering Image Dataset (K-means) 22
Figure 43- Silhouette Distribution Graph for Worst Training clustering Image Dataset (K-means) 22
Figure 44- Silhouette Distribution graph for best clustering Iris (AHC) 23
Figure 45- Silhouette Distribution graph for worst clustering Iris (AHC 23
Figure 46- Silhouette Distribution graph for best train clustering Image dataset (AHC) 24
Figure 47- Silhouette Distribution graph for worst train clustering Image dataset (AHC) 24
Figure 48- Silhouette Distribution graph for best clustering Iris (BIRCH) 25
Figure 49- Silhouette Distribution graph for worst clustering Iris (AHC) 25
Figure 50- Silhouette Distribution graph for best train clustering Image dataset (BIRCH) 25
Figure 51- Silhouette Distribution graph for worst train clustering Image dataset (BIRCH) 25

Table Of Tables

Table 1- Initial Silhouette scores 10
Table 2- K-means results PCA (Image Dataset) 11
Table 3- K-means results W/ PCA & Contamination 11
Table 4- AHC Initial Results Image Dataset 13
Table 5- Differing linkage Methods (Image Dataset) 13
Table 6- Example of the parameters result that were ran 14
Table 7- Initial BIRCH results 16
Table 8- Initial K-Means results (Iris Dataset) 17
Table 9- K-means for Iris with differing configurations for PCA 2 17
Table 10- K-means for Iris with differing configurations for PCA 3 18
Table 11- initial AHC results for Iris 18

ABSTRACT

This project explores the application and optimization of three unsupervised clustering algorithms—K-Means, Agglomerative Hierarchical Clustering (AHC), and BIRCH—on two benchmark datasets: the Iris dataset and a university-provided image dataset. A key focus of the study is to enhance the adaptability and performance of these algorithms through the integration of AI-driven techniques, including hyperparameter optimization, dimensionality reduction using PCA, and outlier detection via Isolation Forest.

The Iris dataset provides a controlled environment for testing, with well-defined clusters, while the image dataset introduces challenges such as high dimensionality and complex feature structures. Feature extraction using the VGG16 model was applied to the image dataset to reduce dimensional complexity before clustering. PCA and t-SNE were employed for further dimensionality reduction and visualization. Outlier removal was conducted using Isolation Forest, with contamination levels of 0.05, 0.1, and 0.2, to improve clustering robustness.

The performance of each clustering algorithm was evaluated using metrics like the silhouette score, Davies-Bouldin index, Calinski-Harabasz index, and Dunn index. K-Means consistently outperformed AHC and BIRCH across both datasets, particularly when paired with PCA (2 components) and moderate outlier removal (contamination 0.2). AHC showed strong performance with Ward linkage, but its effectiveness declined with higher cluster counts, particularly on the image dataset. BIRCH performed well on small clusters but struggled with scalability and complex feature spaces.

This study demonstrates that AI-driven techniques can significantly improve the efficiency and performance of clustering algorithms, especially in high-dimensional and noisy datasets. Future work will explore further automation of hyperparameter tuning and generalization across more diverse datasets.

I. INTRODUCTION & LITERATURE REVIEW

1.1 Project background

Machine Learning (ML), a core subset of artificial intelligence (AI), has transformed the way systems perform tasks without explicit programming [1]. Bishop and Nasrabadi (2006) emphasize how ML models, especially in pattern recognition, can automatically extract insights from data, reducing the need for manual coding. ML's versatility spans industries like healthcare and finance, but its role in data analysis is particularly critical for unsupervised learning [1].

Murphy (2012) highlights that both supervised and unsupervised machine learning algorithms often require manual intervention for tasks like parameter tuning and feature selection [2]. These tasks are time-consuming and require domain expertise, as each dataset typically needs specific adjustments. My project addresses these challenges by integrating AI-driven techniques to minimize human dependency, making unsupervised learning more efficient and effective.

Schmidhuber (2015) illustrates how AI is transitioning from a machine learning application to a tool for optimizing and generating algorithms [3]. This is especially relevant for unsupervised learning, where the lack of labeled data complicates tasks. His research on deep neural networks highlights AI's ability to address challenges that conventional algorithms face. This insight is pivotal to my project, which focuses on enhancing the adaptability of clustering algorithms by automating hyperparameter tuning—a process traditionally dependent on expert intervention.

Jain et al. (1999) reviewed clustering algorithms, focusing on their use in detecting patterns from unlabeled data, noting that techniques like K-Means and AHC show inconsistent performance based on data structure [4]. My project addresses this variability by tackling challenges from diverse datasets, including noise and high dimensionality. Xu and Wunsch (2005) further examined these issues, noting the difficulty many popular clustering methods face in maintaining performance across different datasets [5]. These findings underscore the need for adaptive solutions, which my project aims to provide through AI-driven optimizations.

1.2 Main Problems

1.2.1 Parameter Sensitivity:

Clustering algorithms are highly sensitive to hyperparameter settings, as noted by Jain (2010), who discusses how K-Means performance relies on choosing the right number of clusters, contamination levels, and distance metrics [9]. Misjudging these can lead to over-clustering or under-clustering, significantly reducing accuracy. My project tackles this by using AI-based parameter optimization to minimize manual trial-and-error efforts.

Hennig (2015) also highlights inefficiencies in manual parameter selection, suggesting that AI-driven approaches, such as genetic algorithms or reinforcement learning, can automate this process, reducing human bias and improving accuracy [10]. Ester et al. (1996) emphasizes how DBSCAN and other density-based methods suffer from contamination-level sensitivity, which leads to misidentifying clusters and outliers [11]. My project addresses these limitations by employing AI-driven outlier detection to refine contamination parameters, enhancing clustering robustness.

Hinneburg and Gabriel (2007) explore the challenges posed by noisy datasets, where traditional clustering methods struggle to adapt without manual parameter tuning [12]. My project aligns with their findings by integrating automated AI-based parameter adjustments, aiming to improve clustering performance in complex data environments.

Clustering algorithms like K-Means, BIRCH, and AHC, as discussed by Jain (2010), are highly sensitive to hyperparameters such as cluster numbers, linkage criteria, and contamination levels [9]. For instance, inappropriate K-Means configurations can lead to over- or under-clustering, directly affecting the validity of the results. Similarly, BIRCH and AHC depend on hyperparameters like linkage criteria and branching factors, which can cause overfitting or underfitting in complex datasets [14].

AI techniques, such as genetic algorithms and neural networks, can optimize these hyperparameters based on dataset characteristics. As Hinton et al. (2006) demonstrated, deep learning models can self-adjust by learning from data, a principle now applied to unsupervised learning to minimize the need for manual tuning [8].

1.2.2 Scalability Issues:

Traditional clustering algorithms face significant scalability challenges when applied to large datasets. BIRCH, though designed for such data, still encounters memory constraints when the dataset exceeds its internal capacity [14]. AHC, with its time complexity of $O(n^2)$, becomes computationally expensive for large datasets due to the need to calculate pairwise distances [5].

To address these issues, AI techniques such as parallel processing and distributed computing can enhance scalability. Sculley (2010) demonstrated how parallelized operations made web-scale K-Means clustering feasible, reducing time complexity and enabling the algorithm to handle billions of data points [13]. In this project, similar principles will be applied to BIRCH and AHC, optimizing their performance on larger datasets through AI-driven parallelism.

Moreover, data reduction techniques like PCA (Principal Component Analysis) offer a practical solution to scalability concerns. By reducing the dimensionality of the data, PCA decreases the computational load on clustering algorithms while retaining most of the dataset's variance. Hinneburg and Gabriel (2007) showed that dimensionality reduction significantly enhances computation speed without compromising clustering quality [12]. This project will employ PCA alongside clustering to improve the scalability and efficiency of BIRCH and AHC, particularly for the image dataset used in this research.

1.2.3 Dataset Variability and Generalization:

A significant challenge in unsupervised learning is the variability in clustering performance across different datasets. Clustering algorithms often deliver inconsistent results when faced with datasets of varying sizes, noise levels, and dimensionality. K-Means, for instance, excels with compact, well-separated clusters but struggles with clusters of unequal size, density, or noisy data [5].

AI-driven enhancements can address these inconsistencies by enabling clustering algorithms to adapt dynamically to various datasets. Neural networks, for example, can learn the structure of different datasets and recommend optimal clustering strategies. By integrating AI techniques for outlier and anomaly detection, algorithms like AHC and BIRCH can become more robust against noise and outliers, improving their performance across diverse datasets [11].

Recent advancements using autoencoders—neural networks that learn data representations—have shown promise in improving clustering performance for complex data. Autoencoders transform data into a lower-dimensional latent space, simplifying the clustering task and reducing noise, which ensures more consistent results across datasets [8]. Although autoencoders are not implemented in my project, understanding their impact highlights how AI techniques can enhance the consistency and generalization of unsupervised learning algorithms, which aligns with my project's goals.

1.2.4 Outlier Detection Challenges:

Outliers significantly impact the performance of clustering algorithms in unsupervised learning, often leading to poor cluster formation and skewed performance metrics. For instance, Ester et al. (1996) highlighted how algorithms like DBSCAN are sensitive to noise, with outliers distorting cluster boundaries and resulting in incorrect classifications [11].

Traditional outlier detection often involves manual intervention, where analysts rely on domain knowledge to identify and remove outliers. This approach is both time-consuming and prone to bias, especially in large datasets. To automate this process, AI-driven methods like Isolation Forest have gained prominence. Isolation Forest isolates observations based on random feature selection and split values, with outliers requiring fewer splits, resulting in higher anomaly scores [8].

In my project, Isolation Forest was incorporated to automate outlier detection, significantly improving cluster quality. The impact was evident in higher silhouette scores and clearer cluster boundaries across both the image and Iris datasets. Additionally, contamination levels (0.05, 0.1, 0.2) were fine-tuned across all clustering methods (K-Means, AHC, BIRCH) to optimize outlier removal. This approach ensured that only the most anomalous points were removed, further enhancing clustering performance, as noted by Hinneburg and Gabriel (2007) [12].

1.2.5 Dimensionality Reduction and Data Complexity:

One major challenge in clustering high-dimensional data is the "curse of dimensionality," where the increased feature space reduces the effectiveness of distance-based clustering algorithms. As explained by Bishop (2006), higher dimensions cause data points to become sparse, leading to uniform distances between points, making clustering difficult [1]. Aggarwal et al. (2001) also highlight that K-Means and similar algorithms struggle in high-dimensional spaces as the distance between data points becomes less meaningful [15]. This issue is particularly relevant in image datasets, where the number of features can be vast.

Dimensionality reduction techniques like PCA and t-SNE are essential for improving clustering in such contexts. Maaten and Hinton (2008) demonstrated that reducing dimensionality helps reveal hidden structures by projecting data into lower dimensions [17]. In this project, PCA was applied to transform the high-dimensional image dataset into a smaller set of components while preserving variance, as recommended by Jolliffe (2002) [9]. This allowed algorithms like K-Means, AHC, and BIRCH to more effectively identify clusters. Additionally, t-SNE was used to visualize the non-linear structures, providing further insights into the cluster formation. The use of these techniques aligns with Tenenbaum et al. (2000), who emphasized that dimensionality reduction improves clustering performance by filtering out noise and irrelevant features [16].

1.2.6 Model Interpretability and Performance Evaluation:

A critical issue in both supervised and unsupervised learning is model interpretability, which becomes especially challenging in unsupervised learning due to the absence of labels to guide cluster interpretation. Rudin (2019) emphasizes that interpretable models are crucial for making AI systems' results understandable and actionable [18]. In this project, interpretability was enhanced by using visual tools like scatter plots and PCA projections to intuitively assess cluster cohesion and separation, beyond traditional metrics like silhouette scores, Davies-Bouldin index, and Calinski-Harabasz index. These visualizations provided a clearer understanding of how well the clusters were formed.

Xu and Wunsch (2005) note that the evaluation of clustering algorithms is subjective since there is no ground truth in unsupervised learning [5]. Metrics like the silhouette score assess the compactness and separation of clusters but can be misleading, particularly in high-dimensional spaces where distance-based metrics lose significance (Amid and Warmuth, 2019) [19]. Therefore, qualitative methods such as visualizing clusters alongside these metrics are necessary for a more comprehensive understanding of clustering performance. This dual approach was essential in this project to avoid relying solely on numerical scores, providing a holistic evaluation of the clustering results.

1.2.7 Class imbalance

Class imbalance is a major challenge in unsupervised learning, where some classes may be significantly underrepresented, making it difficult for clustering algorithms to identify patterns across groups. This can lead algorithms like K-Means to overlook smaller clusters. He and Garcia (2009) and Japkowicz and Stephen (2002) explain that imbalanced data makes algorithms prioritize well-represented classes, reducing their ability to detect rare patterns [21], [22]. Although the image dataset used in this project consists of balanced classes, the issue remains critical for broader machine learning research, as Chawla (2009) and Estabrooks and Japkowicz (2001) suggest that traditional algorithms struggle with unevenly distributed classes, often misidentifying smaller groups [23], [4].

The balanced nature of the dataset in this project simplifies evaluating clustering algorithms, but the impact of class imbalance on unsupervised learning is crucial for future applications on more complex datasets. Even with balanced data, clustering algorithms must be adaptable across varied datasets with differing structures, noise, and dimensionality. My project aims to address these challenges by integrating AI techniques to enhance the flexibility and performance of K-Means, AHC, and BIRCH in diverse scenarios.

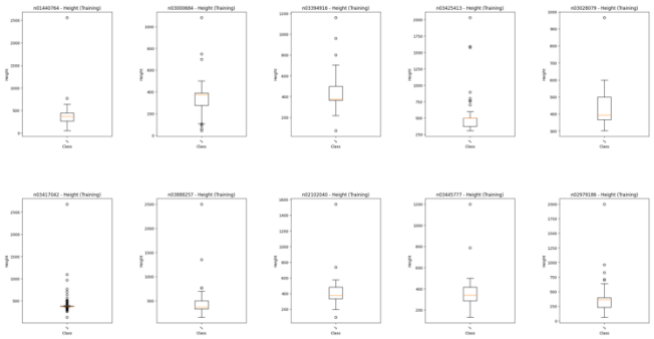


Figure 1- Class height for Image Dataset(Training)

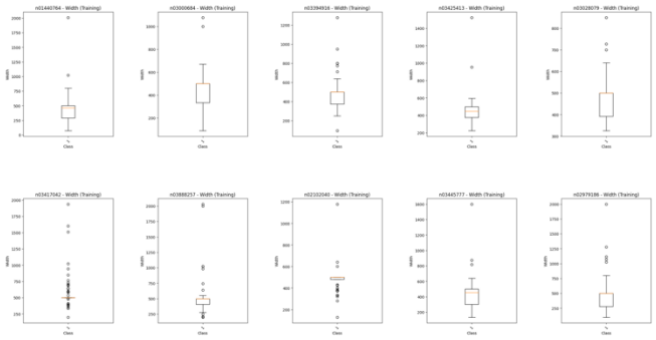


Figure 2- Class Width for Image Dataset (Training)

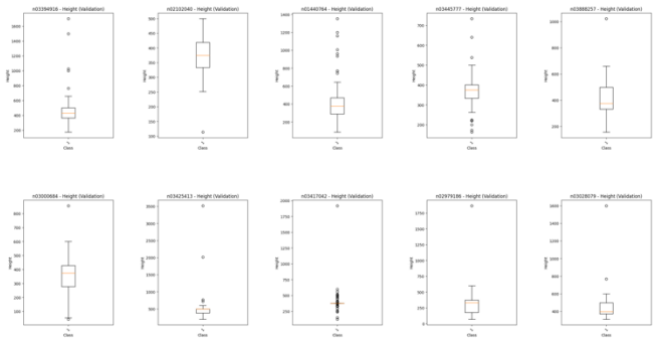


Figure 3- Class Height for Image Dataset(Testing/ Validation)

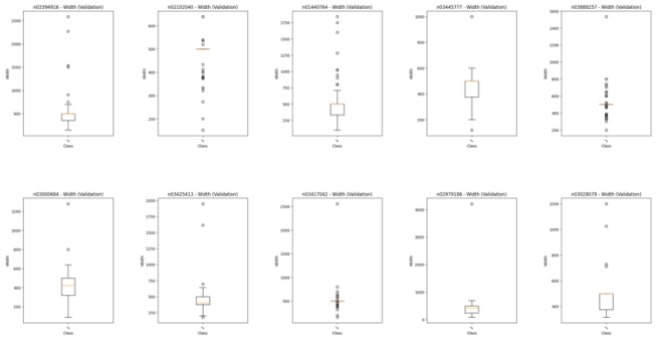


Figure 4- Class Width for Image Dataset (Testing/ Validation)

1.3 Aims and Objectives

The primary aim of this project is to enhance the adaptability and performance of unsupervised clustering algorithms through AI-driven techniques. The focus will be on K-Means, Agglomerative Hierarchical Clustering (AHC), and BIRCH, applied to both the Iris dataset and a university-provided image dataset. The specific objectives include:

1.Automating Hyperparameter Tuning:
Using grid search and manual tuning to optimize hyperparameters, such as cluster numbers and contamination levels, for more effective clustering. Genetic algorithms were researched but deemed resource-intensive, so they were not implemented in favor of more feasible methods.

2. Improving Robustness with Outlier Detection:
Incorporating AI-based methods, like Isolation Forest, to automatically detect and remove outliers, improving clustering results in noisy datasets. Consistent contamination levels (0.05, 0.1, 0.2) will be applied to maintain scope.

3. Enhancing Scalability:
Addressing scalability challenges by integrating PCA for dimensionality reduction, reducing computational complexity without compromising accuracy.

4. Generalization Across Datasets:
Improving clustering algorithm adaptability to handle different dataset characteristics, ensuring consistent performance across both the Iris and image datasets.

5. Evaluating Performance Using Multiple Metrics:
Assessing clustering algorithms through quantitative metrics (silhouette score, Davies-Bouldin index, Calinski-Harabasz index) and qualitative visualizations (PCA and t-SNE plots), providing a comprehensive evaluation of algorithm performance.

II. METHODOLOGY & DESIGN

2.1 Data Preprocessing

Pre-processing is a fundamental step in ensuring that the data is in a suitable format for clustering algorithms. Both the iris dataset and the university-provided image dataset(obtained via applied Ai and machine learning module) underwent different pre-processing steps based on their unique characteristics.

2.1.1 Iris Dataset

The Iris dataset consists of 150 flower samples, each described by four features: sepal length, sepal width, petal length, and petal width, with the aim of clustering them into three groups corresponding to different species.

Why the Iris Dataset?
The Iris dataset is a standard benchmark for clustering algorithms due to its simplicity, well-defined clusters, and low dimensionality. Its manageable size and feature count (four dimensions) make it ideal for testing multiple algorithms with minimal computational overhead.

This provides a reliable baseline to evaluate clustering algorithms effectiveness in a controlled, well-structured environment.

Feature Scaling with RobustScaler:
The dataset's features were scaled using the RobustScaler, which scales based on the interquartile range (IQR), making it less sensitive to outliers compared to MinMaxScaler or StandardScaler. This approach ensures that potential outliers do not distort clustering algorithms, particularly those like K-Means and AHC, which are sensitive to distance metrics.

Why RobustScaler?
RobustScaler was chosen because the Iris dataset has some variability in feature values. By using this method, the influence of outliers is minimized, resulting in more stable and reliable clustering results. This improves the robustness of distance-based algorithms like K-Means.

2.1.2 Image Dataset

The image dataset used in this project contains approximately 12,000 images, split into 9,000 for training and 3,000 for testing. To reduce computational complexity, a random sample of 1,000 images from each set was selected for the analysis. The dataset originally had 10 labeled classes, but to suit unsupervised learning, the labels were removed using a custom Python script. This ensured the dataset was fully unlabeled and ready for clustering algorithms. [Code for label removal shown in the Appendices].

Data Augmentation for Training Data:

Data augmentation was applied using the ImageDataGenerator to enrich the training set and improve clustering robustness. Techniques included:

- Rescaling: Each image was scaled by 1/255.
- Rotation: Random rotations up to 20 degrees.
- Shifting:
Width and height were shifted up to 20%.
- Shearing:
A 20% shear transformation was applied.
- Zooming:
Images were zoomed up to 20%.
- Horizontal Flipping:
Random horizontal flips introduced variation.
- Fill Mode:
Missing pixels from transformations were filled using the nearest pixel strategy.

These augmentation techniques expanded the dataset, making the clustering algorithms more robust to variations such as changes in orientation, scale, and positioning.

Validation Data:
For the validation set, only rescaling (1/255) was applied, ensuring that the clustering evaluation was based on unaltered images, providing a more realistic measure of performance.

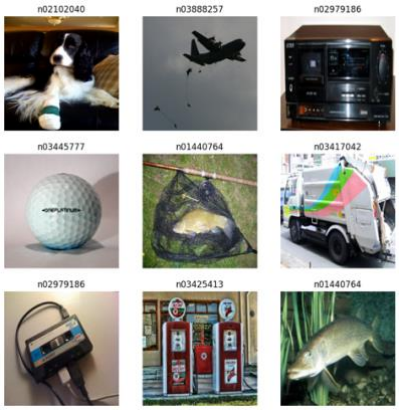


Figure 5- Training Images with labels (Image Dataset)

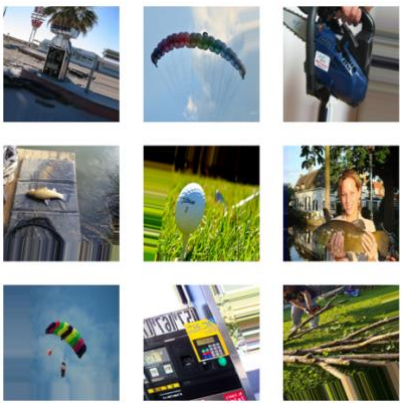


Figure 6- Training images without labels(Image Dataset)

Why the Image Dataset?
The image dataset presents a more challenging environment for clustering due to the high dimensionality and complexity of visual features. Testing clustering algorithms on this real-world data helps evaluate their adaptability and scalability. It also demonstrates how well the algorithms generalize from structured datasets like Iris to more complex scenarios.

Feature Extraction Using VGG16:
To transform raw images into a more manageable feature space, the pre-trained VGG16 model was employed. VGG16, trained on ImageNet, extracts high-level features such as shapes, textures, and patterns, making it an ideal tool for feature extraction in this project.

Justification for VGG16:
Despite the lossy compression of JPEG images, VGG16's robust pre-trained architecture ensures high-quality feature extraction. A careful review of the images ensured consistent quality across the dataset. VGG16's balance between accuracy and computational efficiency was critical, making it a more feasible option compared to other models like ResNet, especially given the cost constraints of running models on Google Colab.

2.2 Dimensionality Reduction

Dimensionality reduction was essential to counteract the curse of dimensionality, improving both clustering performance and visualization.

Principal Component Analysis (PCA):
PCA reduced the dimensionality of both datasets, keeping the most significant variance. For Iris, it reduced the four features to 2 and 3 components, improving visualization and computational efficiency. For the image dataset, PCA played a crucial role in simplifying the high-dimensional VGG16-extracted features.

Why PCA?:
PCA is a widely used technique that effectively reduces dimensionality while preserving key variance, allowing clustering algorithms to focus on meaningful structures [21]. It was critical in reducing noise and redundancy in the data, especially for high-dimensional image features.

t-SNE (t-distributed Stochastic Neighbor Embedding):
t-SNE was used for visualizing the clusters in two dimensions. Unlike PCA, t-SNE is a non-linear technique, better suited for capturing local structures and revealing clusters that are not linearly separable.

Why t-SNE?
t-SNE excels at maintaining neighborhood relationships between data points, which makes cluster visualization easier [17]. While it was used primarily for visualization, it provided valuable insights into how well the algorithms grouped the data.

2.3 Outlier Detection

Outliers and Isolation Forest:
Outliers can distort clustering results by forming artificial clusters or altering the shape of existing ones. To address this, Isolation Forest was employed to detect and remove outliers from the datasets.

Why Isolation Forest?:
Isolation Forest works by recursively partitioning the data through random splits, isolating outliers more quickly due to their distinctive characteristics. This makes it computationally efficient and highly effective for detecting anomalies, especially in high-dimensional datasets like the image dataset [8].

Contamination Levels:
Three contamination levels (0.05, 0.1, 0.2) were tested to determine the best threshold for outlier detection. By fine-tuning these levels, I ensured that only the most anomalous data points were removed,

improving clustering results without discarding too much valuable data.

2.4 Clustering Algorithms

Three different clustering algorithms K-Means, Agglomerative Hierarchical Clustering (AHC), and BIRCH were applied to the datasets. Each algorithm has distinct characteristics, making them suitable for different types of datasets and clustering structures.

2.4.1 K-Means Clustering

What is K-Means?
K-Means is a centroid-based clustering algorithm that partitions data into k clusters by assigning each point to the nearest centroid. The algorithm iteratively adjusts centroids to minimize intra-cluster variance, aiming for similarity within clusters and distinctiveness between clusters [9].

Why K-Means?
K-Means was selected for its efficiency and simplicity. It performs well on smaller datasets, like Iris, where clusters are well-separated, providing a solid baseline for comparison with more complex methods.

Limitations
K-Means assumes clusters are spherical and equal in size, which can be limiting for complex datasets like the image dataset. To overcome this, outlier detection and dimensionality reduction were applied to enhance its performance.

2.4.2 Agglomerative Hierarchical Clustering (AHC)

What and Why AHC?
Agglomerative Hierarchical Clustering (AHC) is a clustering algorithm that builds a nested structure of clusters based on linkage criteria, resulting in a dendrogram that visually represents the cluster hierarchy [10]. AHC is particularly useful for datasets with substructures or non-spherical clusters, as it explores hierarchical relationships without needing to specify the number of clusters in advance. However, for consistency, I pre-set the number of clusters across all algorithms to facilitate comparison. Its flexibility in testing various linkage methods (single, complete, Ward's, and average) made it suitable for both the Iris and image datasets.

Linkage Criteria:
Several linkage methods were tested:
- Single:Elongates clusters.
- Complete: Forms compact clusters.
- Ward's: Minimizes variance within clusters.
- Average:Computes the average distance between clusters.

This flexibility made AHC ideal for datasets with non-spherical or unevenly distributed clusters.

2.4.3 BIRCH Clustering

What and Why BIRCH?
BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) is designed for large-scale, high-dimensional datasets, incrementally building a Clustering Feature (CF) Tree to cluster data efficiently while minimizing memory usage [14]. Its scalability makes it ideal for processing large datasets, such as the image dataset in this project. BIRCH clusters data in memory-efficient ways, handling data in chunks, which is crucial when dealing with the high-dimensional feature vectors extracted from VGG16.

Threshold and Branching Factor:
The threshold controls cluster compactness, while the branching factor limits child nodes in the CF tree. These hyperparameters were tuned to prevent overfitting or underfitting the data.

Justification:
BIRCH was chosen for its ability to handle large datasets incrementally and efficiently. While AHC struggles with memory constraints, BIRCH's iterative, scalable approach makes it more suitable for clustering high-dimensional image data.

2.5 Performance Evaluation

The performance of the clustering algorithms was evaluated using both quantitative metrics and qualitative visualizations, ensuring a comprehensive understanding of how well each algorithm performed on both the Iris and image datasets.

2.5.1 Quantitative Metrics

Silhouette Score:
Evaluates how well-separated clusters are by comparing intra-cluster distances to nearest-cluster distances. A score closer to 1 indicates well-defined clusters, while a score near -1 suggests poor clustering with overlap [9].

*Why Silhouette Score?**
Silhouette score combines compactness (points close together within clusters) and separation (distinct clusters), making it ideal for both Iris and image datasets. It was applied to the best and worst-performing configurations to assess cluster quality in detail.

Silhouette Distribution:
Analyzing the spread of silhouette scores provides deeper insight into individual point distribution within clusters, offering a more detailed view of clustering performance.

Davies-Bouldin Index:
Measures the ratio of within-cluster scatter to between-cluster separation. Lower values indicate more compact and better-separated clusters [9].

Why Davies-Bouldin?
Useful for datasets with varying cluster sizes or densities, such as the image dataset. Complements the silhouette score by assessing the overall formation of clusters.

Calinski-Harabasz Index:
Also known as the variance ratio criterion, it measures the ratio of between-cluster dispersion to within-cluster dispersion. Higher values indicate better clustering [9].

Why Calinski-Harabasz?
This metric quantifies distinctness and compactness, particularly effective for well-separated clusters like those in the Iris dataset.

Dunn's Index:
Measures the ratio of within-cluster scatter to between-cluster separation. Higher scores indicate better clustering, as clusters are compact and well-separated.

Why Dunn's Index?
Dunn's index is reliable for assessing compactness and separation, especially in high-dimensional datasets like the image dataset. It was applied to compare the best and worst clusters for each technique, revealing how well the clusters hold their shape and separation under different conditions.

2.5.2 Qualitative Visualisations:

Visualising the clusters is essential to understand the structure and separation between different groups. Scatter plots of the clustered data, after dimensionality reduction using PCA and t-SNE, were generated for both datasets.

Scatter Plots:
After applying PCA and t-SNE to reduce the dimensionality of the data, scatter plots were generated to provide visual insights into the clustering results. The plots helped assess the compactness and separation of clusters visually.

Why Scatter Plots?:
Visualizing clusters in 2D or 3D space provides an intuitive understanding of how well the algorithms are grouping the data. This is particularly useful in high-dimensional datasets like the image dataset, where quantitative metrics alone may not fully capture the nuances of the clustering structure.

2.6 Tools and Libraries

The project was implemented using Python in Google collab, and the following libraries were utilized:

scikit-learn:
K-Means, AHC, BIRCH, dimensionality reduction (PCA, t-SNE), and evaluation metrics.

Pandas:
Data manipulation, preprocessing, and handling datasets.

NumPy:
For handling numerical operations and matrix computations.

Matplotlib and Seaborn:
generating visualizations of the clustering results (scatter plots, dendrograms).

TensorFlow/Keras:
For feature extraction from the image dataset using the pre-trained VGG16 mode

2.7 Design Mind Map

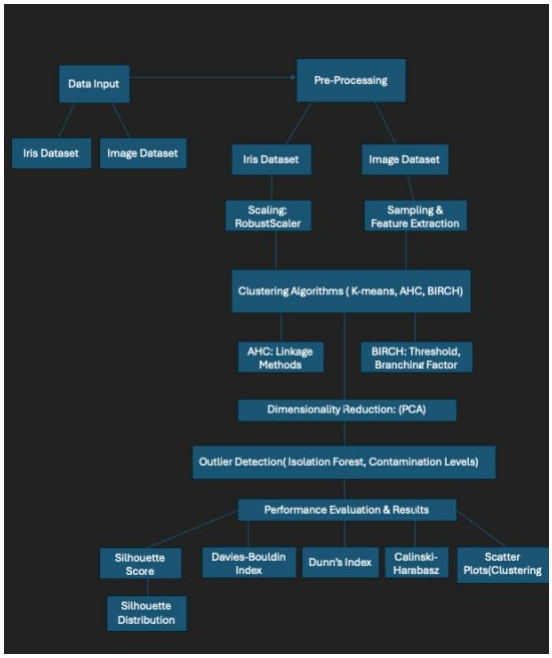


Figure 7- Brief Mind map of Methodology and Design

III. EXPERIMENTS, EVALUATION, RESULTS & DISCUSSION

This section evaluates the performance of three clustering algorithms—K-Means, Agglomerative Hierarchical Clustering (AHC), and BIRCH—on the Image and Iris datasets. For the image dataset, clustering follows feature extraction using VGG16, while the Iris dataset utilizes preprocessed features.
Each algorithm's performance is measured using metrics like the Silhouette Score, Davies-Bouldin Index, Calinski-Harabasz Index, and Dunn Index, alongside PCA and t-SNE visualizations for qualitative insights. Results are analyzed separately for each dataset, followed by a final comparison to highlight key trends across algorithms and datasets.

3.1 Image Dataset Results

3.1.1 Initial Results from VGG16 Feature Extraction

The initial clustering analysis was conducted on 1000 training and 1000 validation images from the image dataset. Feature extraction was done using the pre-trained VGG16 model on ImageNet, generating high-dimensional vectors for K-Means, AHC, and BIRCH clustering.

Table 1- Initial Silhouette scores

Number of Clusters	*K-Means (Train)*	*K-Means (Val)*	*AHC (Train)*	*AHC (Val)*	*BIRCH (Train)*	*BIRCH (Val)*
2	*0.0265*	*0.0304*	*0.0185*	*0.0118*	*0.0185*	*0.0118*
3	*0.0882*	*0.1210*	*0.0157*	*0.0126*	*0.0157*	*0.0126*
4	*0.0257*	*0.0298*	*-0.0352*	*-0.0016*	*-0.0352*	*-0.0016*
5	*0.0112*	*0.0136*	*-0.0326*	*0.0030*	*-0.0326*	*0.0030*
6	*0.0044*	*0.0072*	*-0.0310*	*-0.0109*	*-0.0310*	*-0.0109*
7	*0.0006*	*0.0015*	*-0.0299*	*-0.0084*	*-0.0299*	*-0.0084*
8	*-0.0048*	*-0.0014*	*-0.0283*	*-0.0063*	*-0.0283*	*-0.0663*
9	*-0.0048*	*-0.0019*	*-0.0304*	*-0.0037*	*-0.0304*	*-0.0037*
10	*-0.0039*	*-0.0035*	*-0.0294*	*-0.0049*	*-0.0294*	*-0.0049*

Key Observations:
1. K-Means:
Achieved the highest silhouette score at 3 clusters (0.0882 for training, 0.1210 for validation). Scores declined and turned negative beyond 7 clusters, indicating poor performance.

2. AHC:
Showed lower silhouette scores than K-Means, peaking at 0.0185 for 2 clusters (training). Scores also became negative after 4 clusters.

3. BIRCH:
Mirrored AHC's performance, with diminishing silhouette scores beyond 3 clusters and negative scores after 4 clusters.

Evaluation:
Best Clustering:
All algorithms performed best with 3 clusters, suggesting a balance between compactness and separation.

Poor Performance at Higher Clusters: Scores dropped as cluster counts increased, indicating over-clustering or difficulty separating complex image features.

K-Means vs AHC/BIRCH:
K-Means outperformed AHC and BIRCH in cluster separation, likely due to its centroid-based approach being better suited for spherical clusters.

Challenges of High-Dimensional Data:
Low silhouette scores reflect the difficulty in clustering high-dimensional image data, highlighting the need for further optimization like PCA to improve results.

3.2 K-means(Image Dataset)

This section explores the impact of K-Means clustering on the extracted VGG16 features from the image dataset. The algorithm was applied with 2-10 clusters, and the silhouette scores provide insight into the clustering performance before applying dimensionality reduction or outlier removal techniques. These initial results serve as a benchmark for evaluating further improvements.

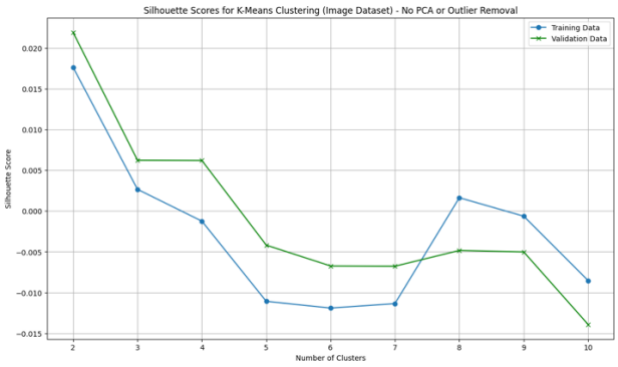


Figure 8- Initial K-means Cluster scores

3.2.1 K-means Clustering with PCA

After the initial K-Means clustering, Principal Component Analysis (PCA) was applied to reduce the dimensionality of the VGG16 features, aiming to improve clustering by reducing noise. PCA was conducted with 2 and 3 components, and silhouette scores were computed for 2-10 clusters. The results for both training and validation sets are shown in Figure 9.

Table 2- K-means results PCA (Image Dataset)

Number of Clusters	PCA Components	Training Score	Validation Score
2	2	0.3464	0.3639
3	2	0.4100	0.4068
4	2	0.3706	0.3850
5	2	0.3374	0.3689
6	2	0.3414	0.3683
7	2	0.3709	0.3638
8	2	0.3504	0.3548
9	2	0.3553	0.3424
10	2	0.3646	0.3484
2	3	0.2709	0.2890
3	3	0.3104	0.3017
4	3	0.3032	0.3035
5	3	0.2774	0.2931
6	3	0.2585	0.2923
7	3	0.2728	0.2781
8	3	0.2619	0.2777
9	3	0.2544	0.2728
10	3	0.2565	0.2635

Key Observations:

1. PCA (2 Components):
The best silhouette score was at 3 clusters for both training (0.4100) and validation (0.4068), showing improved clustering performance compared to the non-PCA results. Scores decline after 3 clusters but still outperform the initial results.

2. PCA (3 Components):
Though 3 clusters again yielded the best scores, the values (0.3104 for training, 0.3017 for validation) were lower than with 2 components, suggesting that 2-component PCA provides better clustering due to simpler structure.

3. General Trends:
Across both PCA setups, 3 clusters consistently performed best, with 2-component PCA generally outperforming 3-component PCA. Performance remained more stable compared to the non-PCA results, particularly for 2-5 clusters.

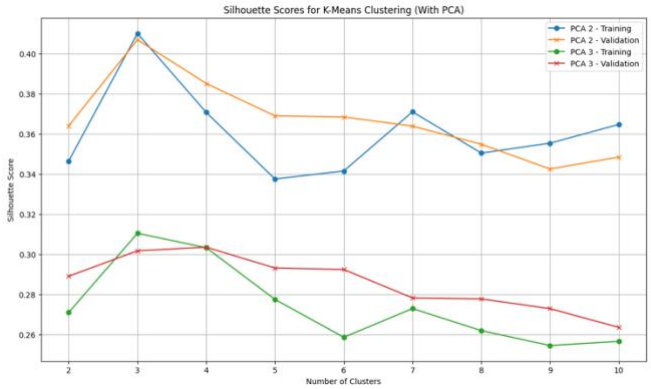


Figure 9- Graph showing PCA 2 vs 3 K-means

3.2.2 PCA and Contamination

Table 3- K-means results W/ PCA & Contamination

Clusters	PCA 2 - Cont. 0.05 (Train)	PCA 2 - Cont. 0.05 (Val)	PCA 2 - Cont. 0.1 (Train)	PCA 2 - Cont. 0.1 (Val)	PCA 2 - Cont. 0.2 (Train)	PCA 2 - Cont. 0.2 (Val)	PCA 3 - Cont. 0.05 (Train)	PCA 3 - Cont. 0.05 (Val)	PCA 3 - Cont. 0.1 (Train)	PCA 3 - Cont. 0.1 (Val)	PCA 3 - Cont. 0.2 (Train)	PCA 3 - Cont. 0.2 (Val)
2	0.3327	0.3654	0.3353	0.3684	0.3404	0.3714	0.2576	0.2966	0.2604	0.3010	0.2662	0.3115
3	0.4101	0.4114	0.4076	0.4153	0.4060	0.4128	0.3130	0.3071	0.3164	0.3091	0.3113	0.3059
4	0.3628	0.3870	0.3625	0.3907	0.3622	0.3856	0.3065	0.2768	0.2818	0.3089	0.3022	0.3032
5	0.3568	0.3787	0.3439	0.3843	0.3420	0.3527	0.2742	0.3028	0.2721	0.3043	0.2717	0.3034
6	0.3453	0.3719	0.3580	0.3678	0.3543	0.3761	0.2658	0.3005	0.2696	0.2906	0.2710	0.2925
7	0.3640	0.3669	0.3494	0.3494	0.3354	0.3628	0.2591	0.2790	0.2584	0.2808	0.2676	0.2819
8	0.3596	0.3581	0.3471	0.3560	0.3645	0.3557	0.2603	0.2657	0.2583	0.2734	0.2585	0.2723
9	0.3630	0.3476	0.3607	0.3405	0.3523	0.3546	0.2521	0.2659	0.2463	0.2805	0.2677	0.2729
10	0.3662	0.3391	0.3669	0.3445	0.3613	0.3319	0.2466	0.2717	0.2457	0.2609	0.2586	0.2755

1. Impact of Contamination:
Contamination levels (0.05, 0.1, 0.2) improve clustering, especially in the 2-3 cluster range where silhouette scores peak. Removing outliers enhances clustering performance, particularly in high-dimensional data.

2. Effect of PCA:
PCA with 2 components consistently yields higher silhouette scores than 3 components, especially at lower contamination levels (0.05 and 0.1). Higher contamination levels (0.2) slightly reduce scores, suggesting potential misidentification of useful data points as outliers.

3. Optimal Clustering:
The best results are observed with 3 clusters, 0.05 contamination, and PCA with 2 components, achieving the highest silhouette scores (0.4101 for training, 0.4114 for validation). PCA with 2 components continues to outperform PCA with 3 components.

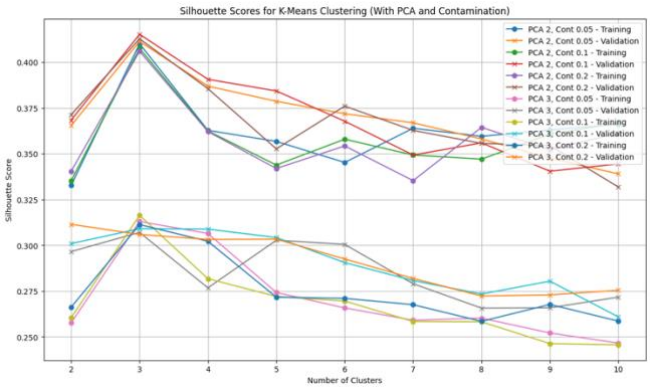


Figure 10- Various Configurations scores for K-Means Image dataset

The t-SNE dimensions are abstract and don't correspond to actual values; they simply capture relationships between data points. This helps visualize the similarity or difference between images and their extracted features.

1. Initial t-SNE (Before Parameter Changes):
The first plots show clustering results without PCA or contamination filtering, directly applying K-Means to VGG16-extracted features. The clusters appear less distinct with overlapping boundaries, indicating difficulty in clearly separating data points.

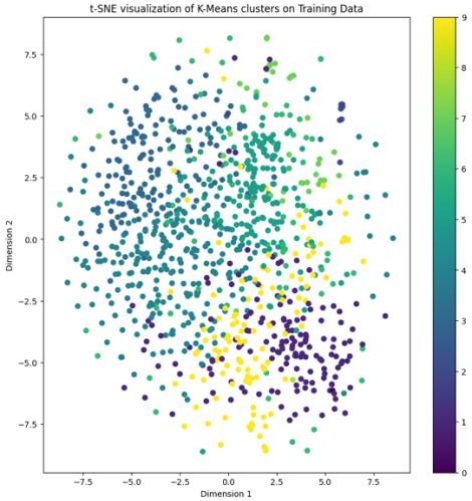


Figure 11- Initial clustering from initial Results of K-means (Training)

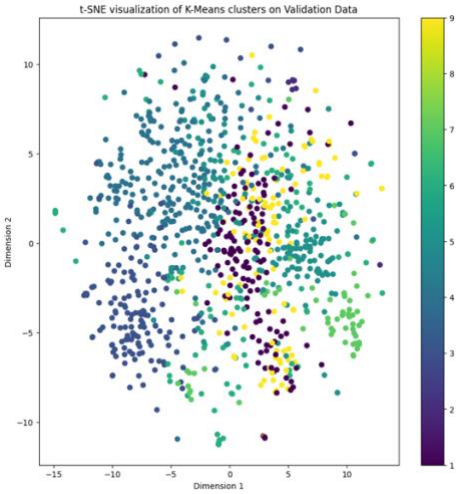


Figure 12- Initial clustering from initial Results of K-means (Validation)

2. **t-SNE with PCA (Best Configuration)**:

The third and fourth plots show the clustering results after applying PCA (2 components) and contamination filtering (0.05), which produced the highest silhouette scores. The clusters are more distinct, with better separation between the three groups (K=3). This clearer grouping reflects the optimal number of clusters based on the evaluation metrics. The tighter clustering shows that K-Means, with PCA and contamination handling, identified more compact and well-separated clusters.

3.Effect of Contamination Handling:
Outlier removal at contamination levels (0.05, 0.1, 0.2) improved clustering, with the best configuration (K=3, PCA=2, contamination=0.05) achieving the highest silhouette scores (0.4101 training, 0.4114 validation) and clearly defined clusters in t-SNE visualizations.

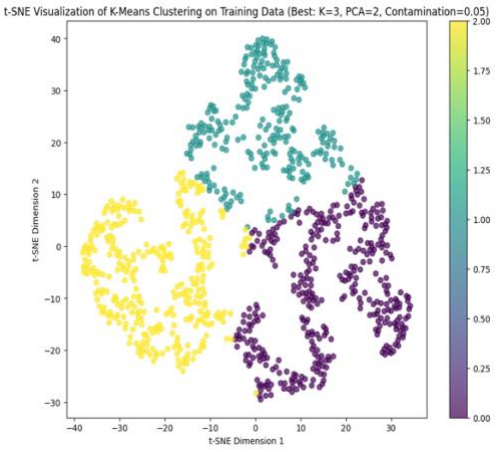


Figure 13- Best Configuration clustering K-Means (Training)

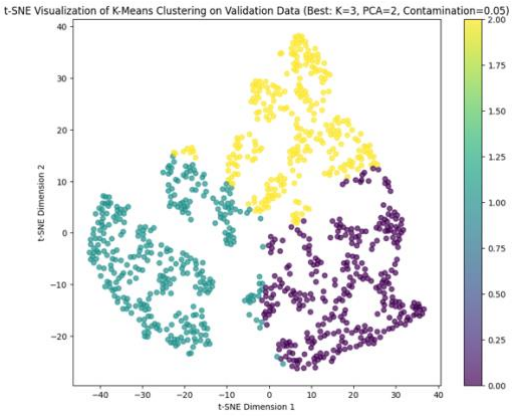


Figure 14- Best Configurations Clustering K-Means (Validation)

4. **t-SNE for Worst Configuration**:
The final two plots depict the clustering results for the worst configuration (K=9, PCA with 3 components, contamination=0.1). Here, the clusters are more scattered and poorly defined, with significant overlap between them. This correlates with the lower silhouette scores for this setup, indicating that the clustering algorithm struggled to form distinct groups due to the increased cluster count and higher contamination level.

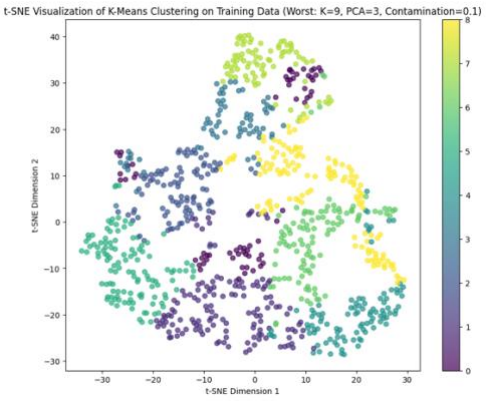


Figure 15- Worst Configuration Clustering K-Means (Training)

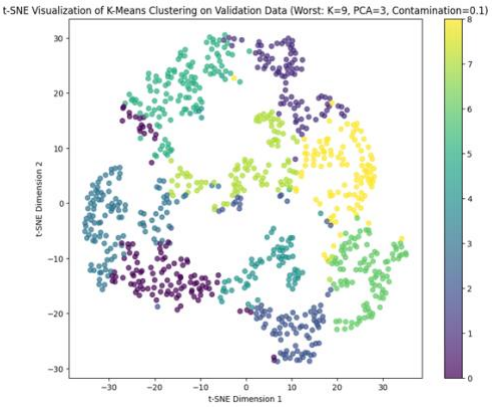


Figure 16- Worst Configuration Clustering K-Means (Validation)

3.3 AHC

3.3.1 AHC Initial Results

Table 4- AHC Initial Results Image Dataset

Number of Clusters	Training Data Silhouette Score	Validation Data Silhouette Score
2	0.0185	0.0118
3	0.0157	0.0126
4	-0.0352	-0.0016
5	-0.0326	0.0030
6	-0.0310	-0.0109
7	-0.0299	-0.0084
8	-0.0283	-0.0063
9	-0.0304	-0.0037
10	-0.0294	-0.0049

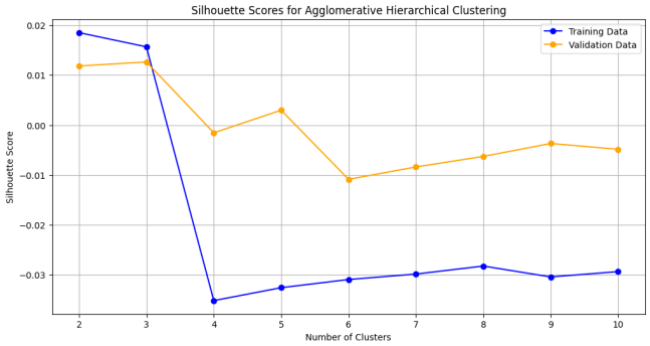


Figure 17- Initial Silhouette Score Graph

AHC was applied to VGG16-extracted features with 2 to 10 clusters for both training and validation sets, and silhouette scores were plotted.

Key Observations:
1. Performance Trends:
AHC's highest silhouette scores were at 2 clusters (0.0185 for training, 0.0126 for validation), significantly lower than K-Means. Scores drop sharply beyond 3 clusters, turning negative at 4, indicating poor cluster separation as the cluster count increases.

2. Training vs. Validation:
Validation scores remain close to zero, while training scores turn negative after 4 clusters, suggesting AHC struggles with the high-dimensional features.

3. K-Means Comparison:
AHC consistently underperforms compared to K-Means, likely due to its hierarchical approach, which may not handle complex data as well.

Evaluation:
1. Best Performance:
AHC performs best with 2 clusters, but even then, it's weaker than K-Means in separating clusters effectively.

2. Negative Scores:
Negative scores after 4 clusters indicate poor cluster quality, with overlapping or misclassified data points.

3. Dimensionality Issues:
AHC struggles with the complexity of high-dimensional data, highlighting the need for dimensionality reduction techniques like PCA for better results.

3.3.2 Linkage methods & impact on Clustering performance

Table 5- Differing linkage Methods (Image Dataset)

Number of Clusters	Ward (Train)	Ward (Val)	Complete (Train)	Complete (Val)	Average (Train)	Average (Val)	Single (Train)	Single (Val)
2	0.0185	0.0118	0.2577	0.2538	0.3023	0.2538	0.3023	0.2538
3	0.0157	0.0126	0.1709	0.1555	0.2065	0.1870	0.1426	0.1706
4	-0.0352	-0.0016	0.0128	0.1351	0.1691	0.1694	0.1290	0.1662
5	-0.0326	0.0030	0.0120	0.0270	0.1439	0.1613	0.1246	0.1502
6	-0.0310	-0.0109	0.0118	0.0263	0.1318	0.1400	0.1222	0.1240
7	-0.0299	-0.0084	0.0120	0.0257	0.1239	0.1288	0.1021	0.1182
8	-0.0283	-0.0063	0.0034	0.0108	0.1198	0.1197	0.1020	0.1171
9	-0.0304	-0.0037	0.0037	0.0107	0.1142	0.1167	0.0987	0.1153
10	-0.0294	-0.0049	0.0038	0.0109	0.1073	0.1131	0.0967	0.1110

Ward Linkage:
Best silhouette score: 0.0185 for 2 clusters on training data and 0.0126 for 3 clusters on validation. Performance drops sharply with more than 3 clusters, leading to negative scores.

Complete Linkage:
Best score: 0.2577 for 2 clusters on training, 0.2538 for validation. Scores fall rapidly after 3 clusters.

Average and Single Linkage:
Highest initial silhouette score: 0.3023 for 2 clusters, but scores decline as clusters increase. Average linkage performs relatively well up to 5 clusters, while single linkage underperforms after 2 clusters.

Key Observations:
1. Low Silhouette Scores:
All linkage methods show low scores compared to K-Means, with several negative scores indicating poor cluster separation.

2. Best Methods:
Average and complete linkage outperform Ward and single linkage, particularly for smaller cluster numbers.

3. Struggles with Higher Clusters:
Like K-Means, AHC struggles beyond 3 clusters, with diminishing scores indicating overlapping and less distinct clusters.

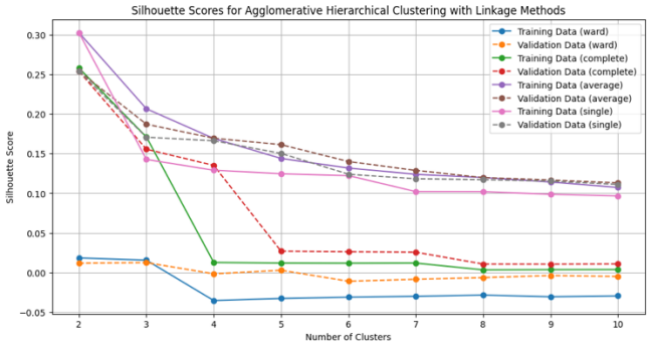


Figure 18- Differing linkage methods

3.3.3 Introduction of PCA & Contamination

I ran a parameter seeking code chunk that ran through all the configurations I was seeking(shown in code submitted). It's important to note that I focused PCA 2 here which will be discussed why later.

Table 6- Example of the parameters result that were ran

Linkage	Clusters	Training (0.05)	Training (0.1)	Training (0.2)	Validation (0.05)	Validation (0.1)	Validation (0.2)
Ward	2	0.3381	0.3222	0.3431	0.3724	0.3368	0.3603
	3	0.3562	0.3511	0.3383	0.3863	0.3852	0.3124
	4	0.3362	0.3330	0.3103	0.3189	0.3493	0.3253
	5	0.3160	0.3320	0.3024	0.3387	0.3277	0.3418
	6	0.3135	0.3409	0.3385	0.3188	0.3333	0.3311
	7	0.3275	0.3513	0.3438	0.3093	0.3124	0.3083
	8	0.3238	0.3305	0.3375	0.3078	0.3174	0.3084
	9	0.3312	0.3235	0.3368	0.3050	0.3093	0.2968
	10	0.3069	0.3309	0.3240	0.2982	0.3169	0.3002

As you can see, this is an example of the parameters it gave me for Ward.

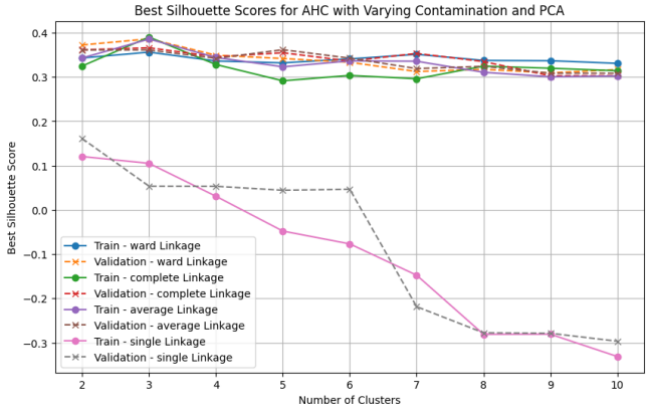


Figure 19- graph showing the best silhouette scores per cluster for each linkage

Evaluation:

*1. **Ward Linkage:***
Best Performance: Peaks at 0.3562 (training) and 0.3863 (validation) for K=2 or K=3 clusters, showing strong separation.

Stability: Maintains decent performance even at K=10, proving robust to larger cluster counts.

Contamination Impact:
Minimal, with scores stable across contamination levels.

2. Complete Linkage:
Strong for Lower Clusters:
Scores 0.3902 (training) and 0.3660 (validation) at K=3 but declines after K=4.

Contamination Effect:
Relatively stable across contamination levels, robust to outliers.

3. Average Linkage:
Consistent Performance:
Scores remain stable, peaking at 0.3627(training) and 0.3611 (validation) at K=3, with a gradual decline but maintaining good performance even at K=10.

Outlier Sensitivity:
Minimal impact from contamination levels.

4. Single Linkage:
Poor Performance:
Negative scores for K=4 and above, worsened by outliers and higher-dimensional data.

Key Insights:
Best Methods:
Ward and Average Linkage are the top performers, especially for K=2 and K=3 clusters.

Complete Linkage:
Strong at lower clusters but declines with higher cluster numbers.

Single Linkage:
Underperforms, especially with higher clusters and contamination levels.

Conclusion:
Ward and Average Linkage, particularly at K=3 with contamination levels of 0.05 or 0.1, offer the best clustering results for AHC on this dataset.

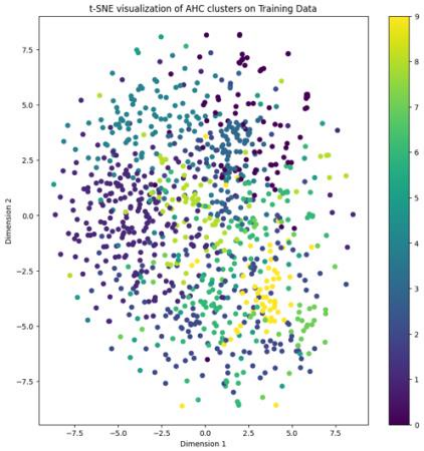


Figure 20- Initial Clustering for Training

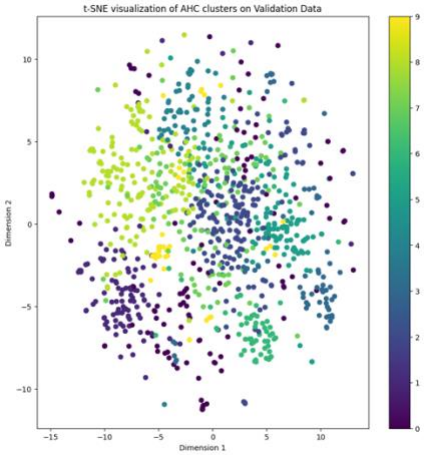


Figure 21- Initial Clustering for Validation

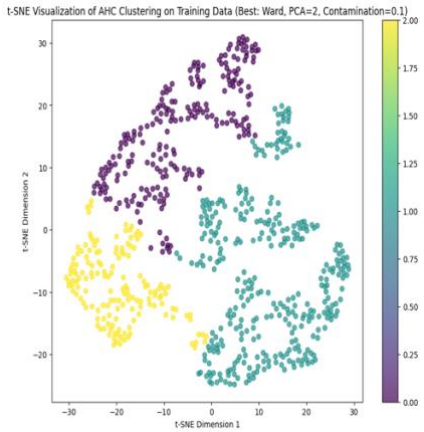


Figure 22- Best Configuration for AHC (Training)

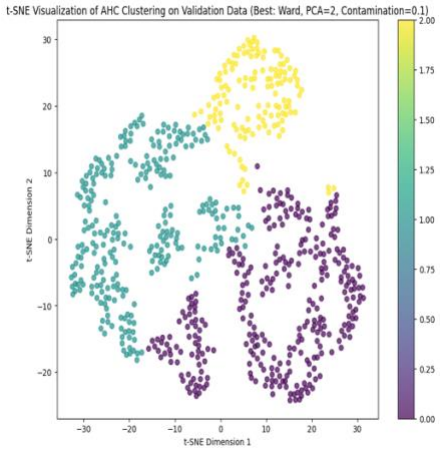


Figure 23- Best Configuration for AHC (Validation)

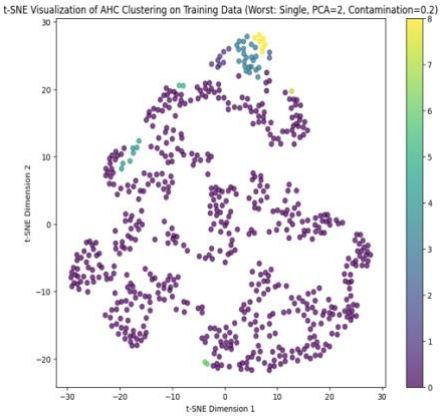


Figure 24- Worst Configuration for AHC (Training)

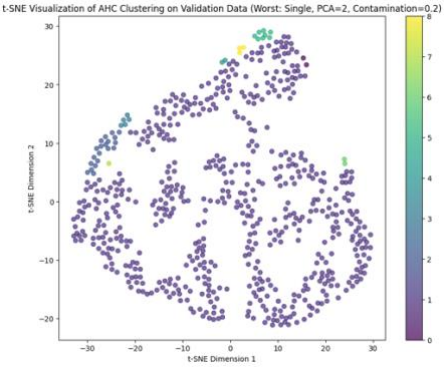


Figure 25- Worst Configuration for AHC (Validation)

3.4 BIRCH

3.4.1 Initial Results BIRCH

Table 7- Initial BIRCH results

Clusters	Training Silhouette	Validation Silhouette
2	0.0185	0.0118
3	0.0157	0.0126
4	-0.0352	-0.0016
5	-0.0326	0.0030
6	-0.0310	-0.0109
7	-0.0299	-0.0084
8	-0.0283	-0.0063
9	-0.0304	-0.0037
10	-0.0294	-0.0049

1. General Performance: The silhouette scores for BIRCH were consistently low, turning negative beyond 3 clusters. Negative scores indicate poor clustering, with data points likely assigned to incorrect clusters or poorly defined groups.

2. Best Performance:
Training Data: The highest silhouette score (0.0185) was achieved with 2 clusters, while 3 clusters gave a slightly lower score (0.0157), indicating better clustering with fewer groups.

Validation Data:
Similarly, 2 clusters produced the highest silhouette score (0.0118), followed by 3 clusters (0.0126), though the scores remained low across the board.

3. Degrading Performance:
Starting from 4 clusters, both training and validation silhouette scores turned negative, signaling that BIRCH struggles with more clusters. This trend continued up to 10 clusters, indicating poor cluster compactness and separation.

4. Validation vs. Training:
Training scores were consistently slightly higher than validation, but the overall pattern of performance degradation was mirrored across both datasets, indicating that BIRCH doesn't generalize well to unseen data.

3.4.2 Incoperation of Threshold, Branching, PCA and Contamination

BIRCH performance was further analyzed with PCA (2 components) for dimensionality reduction and contamination filtering (0.05, 0.1, and 0.2) to remove outliers. Silhouette scores were computed for 2 to 10 clusters, across different thresholds and branching factors, offering insights into the impact of these adjustments on clustering quality.

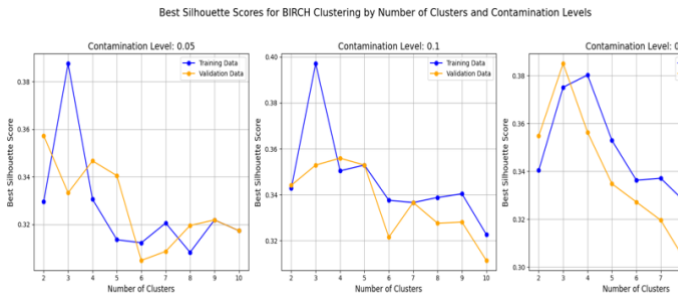


Figure 26- Graph showing the best score for each cluster per contamination level

BIRCH Clustering Performance - Summary of Results

1. Contamination Levels and Number of Clusters:
Contamination 0.05:
Best silhouette score of 0.3876 with 3 clusters on the training set, and 0.3573 with 2 clusters on validation, indicating good clustering but slight overfitting.

Contamination 0.1: The highest score of 0.3971 with 3 clusters on the training set, and 0.3559 with 4 clusters for validation, showing consistent performance across both datasets.

Contamination 0.2: The validation set peaked at 0.3850 for 3 clusters, while training saw a slightly lower score of 0.3751, with more variability but stability at 3 clusters.

2. Threshold and Branching Factors:

Threshold:
A lower threshold of 0.1 consistently gave the best silhouette scores across all contamination levels, producing tighter and more cohesive clusters.

Branching Factor: A factor of 20 provided the most compact clusters, leading to higher silhouette scores. Increasing the branching factor reduced cohesion slightly.

3. Best Configuration:
- Contamination: 0.1
- Threshold: 0.1
-Branching Factor: 20
- Clusters: 3

This configuration yielded the highest silhouette score (0.3971) for training and a solid performance (0.3559) on validation, ensuring a balance between training performance and generalization.

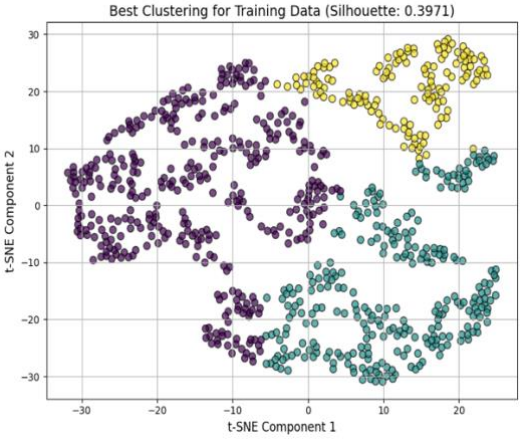


Figure 27- Best clustering for BIRCH (Training)

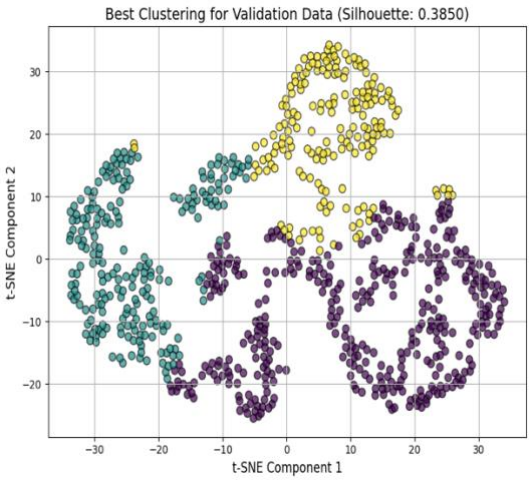


Figure 28- Best Clustering for BIRCH (Validation)

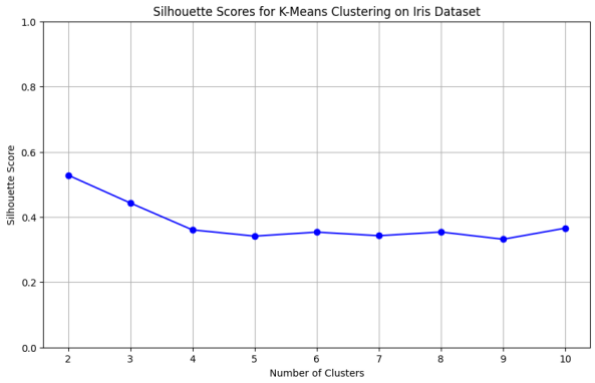


Figure 29- Initial Clustering scores for K-Means

3.6 K-Means Iris Dataset

3.6.1 K-means initial results

Table 8- Initial K-Means results (Iris Dataset)

Number of Clusters	Silhouette Score
2	0.5284
3	0.4431
4	0.3605
5	0.3414
6	0.3536
7	0.3426
8	0.3539
9	0.3317
10	0.3657

K-Means clustering on the Iris dataset produced the highest silhouette score of 0.5284 with 2 clusters, followed by 0.4431 for 3 clusters. Performance steadily declined with more clusters, reaching a low of 0.3317 for 9 clusters before slightly improving to 0.3657 for 10 clusters.

Key Observations:

1. Best Performance at 2 Clusters:
The highest score suggests the data naturally separates into two groups.

2. Decline Beyond 3 Clusters:
Adding more clusters reduced separation quality, increasing complexity without improving clarity.

3. Stable Performance at Higher Clusters:
From 6 clusters onwards, the scores stabilized but remained lower, reflecting diminished clustering quality.

In conclusion, K-Means performed best with 2 clusters, indicating two clear groups. Beyond this, the clustering performance declined, suggesting that the Iris dataset may not support more than a few well-separated clusters

3.6.2 PCA introduction & contamination

After applying the K-Means, PCA of 2 & 3, just like Image Dataset.

Table 9- K-means for Iris with differing configurations for PCA 2

Clusters	Outlier Contamination	Silhouette Score
2	0.05	0.5802418006352600
2	0.1	0.5958117801852370
2	0.2	0.6177775592073480
3	0.05	0.4982316365831560
3	0.1	0.5250499419599120
3	0.2	0.5488062853422400
4	0.05	0.4745113459517080
4	0.1	0.4770379586326400
4	0.2	0.48728091769881900
5	0.05	0.44426042093816300
5	0.1	0.45787580118020800
5	0.2	0.46857758209124500
6	0.05	0.46135088175990000
6	0.1	0.46832992643939300
6	0.2	0.4089922157140760
7	0.05	0.4243935170140650
7	0.1	0.45746813924371500
7	0.2	0.40214351048098500
8	0.05	0.4148279781744980
8	0.1	0.4428398603684580
8	0.2	0.43588549986853700
9	0.05	0.4134019235872620
9	0.1	0.4377209877601310
9	0.2	0.44293334450600400
10	0.05	0.402390099005959
10	0.1	0.42056188124071200
10	0.2	0.4319485052002330

Table 10- K-means for Iris with differing configurations for PCA 3

Clusters	Outlier Contamination	Silhouette Score
2	0.05	0.5589908065867140
2	0.1	0.571231993420344
2	0.2	0.5957160938093340
3	0.05	0.488537253506328
3	0.1	0.4837444604042310
3	0.2	0.5210833402975830
4	0.05	0.41821029418862700
4	0.1	0.4157302991409890
4	0.2	0.43733059459146300
5	0.05	0.393981796388385
5	0.1	0.38763037990212700
5	0.2	0.4060973299171910
6	0.05	0.40393958670578400
6	0.1	0.40296731236414200
6	0.2	0.40534941791429000
7	0.05	0.3405814479303720
7	0.1	0.4025362394651120
7	0.2	0.41405269698774400
8	0.05	0.3855936228727010
8	0.1	0.3684762616671530
8	0.2	0.378494348759315
9	0.05	0.35670092799143800
9	0.1	0.34099986592462200
9	0.2	0.345958897250678
10	0.05	0.34986051028736000
10	0.1	0.32841728381581600
10	0.2	0.3481314676083890

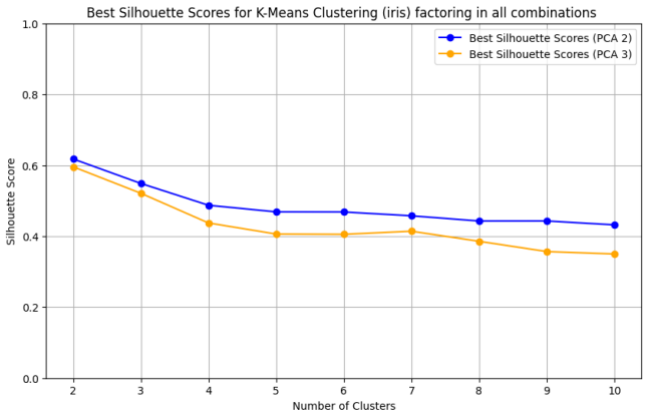


Figure 30- best score for both PCA's K-Means

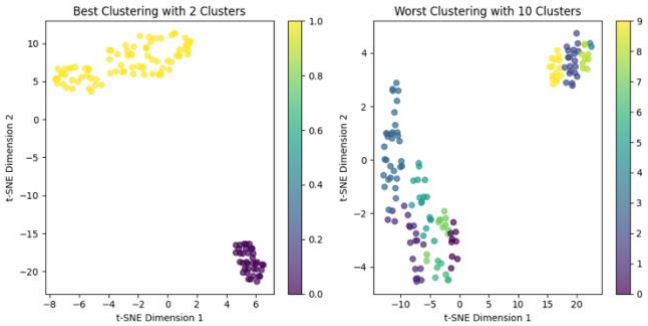


Figure 31- Best and worst Clustering for K-Means

K-Means Clustering on the Iris Dataset: PCA and Contamination Analysis

This analysis examines the effect of PCA and contamination filtering on K-Means clustering for the Iris dataset.

1. PCA with 2 Components:
The highest silhouette score (0.6178) was achieved with 2 clusters and 0.2 contamination. As clusters increased, scores dropped, with 3 clusters yielding a lower score of 0.5488. Reducing the data to 2 components effectively highlighted two distinct groups, with outlier removal (at 0.1 and 0.2 contamination) further improving clustering performance.

2. PCA with 3 Components:
The best silhouette score (0.5957) was achieved with 2 clusters and 0.2 contamination. However, this performance was consistently lower than the PCA 2 results. The addition of a third dimension introduced complexity and noise, lowering clustering quality, especially as the number of clusters increased.

3. Effect of Contamination:
Contamination filtering enhanced clustering quality at 0.2 contamination, producing higher silhouette scores compared to 0.05, which retained more outliers and resulted in poorer clustering.

4. General Trends:
The best configuration used 2 clusters, PCA with 2 components, and 0.2 contamination, with a score of 0.6178.

PCA 2 consistently outperformed PCA 3, demonstrating that fewer dimensions simplified and improved cluster separation.

3.7 AHC Iris Dataset

3.7.1 AHC initial results

Table 11- initial AHC results for Iris

Number of Clusters	Silhouette Score
2	0.5284
3	0.4473
4	0.4125
5	0.3872
6	0.3932
7	0.3929
8	0.3621
9	0.3449
10	0.3304

Key Observations:
1. Best Performance:
The highest silhouette score (0.5284) was achieved with 2 clusters, indicating well-separated clusters aligned with the Iris dataset's natural structure.

2. Moderate Performance for 3 Clusters:
A silhouette score of 0.4473 for 3 clusters reflects AHC's moderate ability to separate the three natural classes, suggesting overlap in feature space.

3. Decreasing Silhouette Scores:

Beyond 3 clusters, scores steadily declined:

- *4 clusters: 0.4125*
- *5-7 clusters: Scores around 0.3872 to 0.3932*
- *8-10 clusters: Scores dropping to 0.3621, 0.3449, and 0.3304*

General Trend:

The best results were obtained with fewer clusters (2-3), while increasing the number of clusters led to diminishing returns, as AHC struggled to form distinct groups beyond the natural three classes of the dataset.

3.7.2 Linkage methods, PCA & Contamination

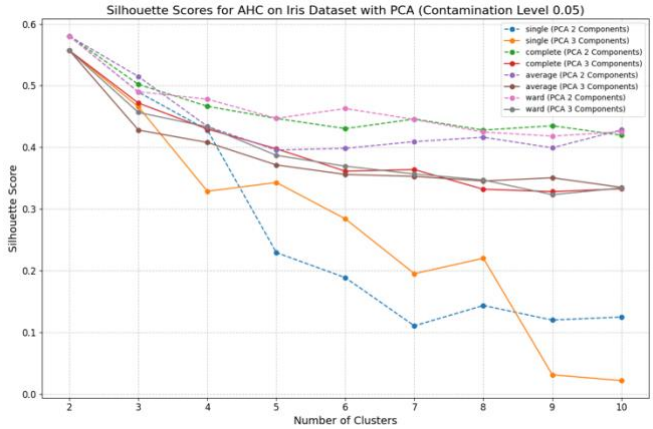


Figure 32- Scores for 0.05 contamination per linkage

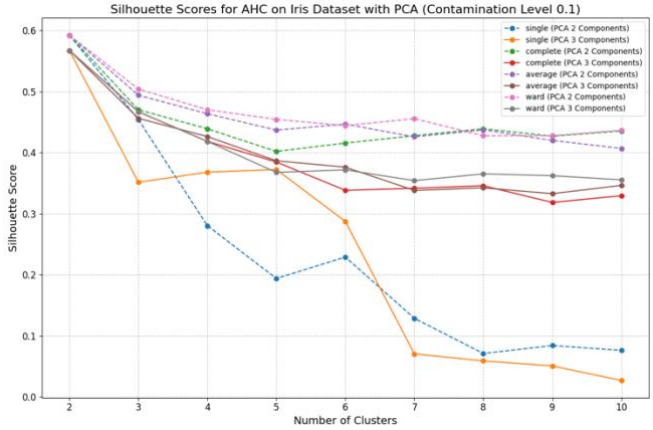


Figure 33- Scores for 0.1 contamination per linkage

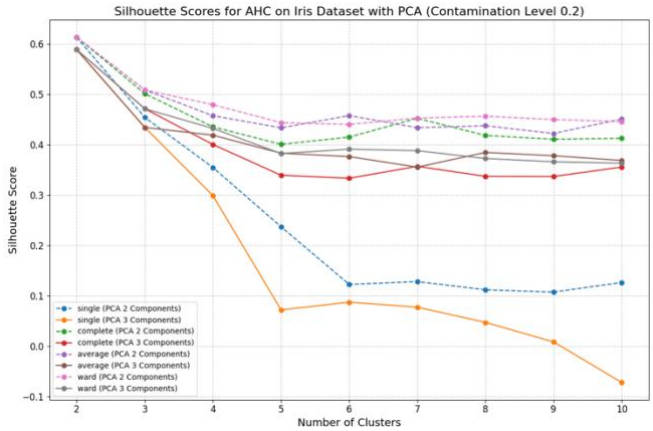


Figure 34- Scores for 0.2 contamination per linkage

AHC Silhouette Scores with PCA and Contamination :

Contamination Level 0.05:
Single Linkage:
Best with 2 clusters, PCA 2 components: 0.5803. Scores drop sharply to 0.1251 at 10 clusters.

Complete Linkage:
Peaks at 0.5803 (2 clusters), maintains stability up to 10 clusters (0.4197).

Average Linkage:
Best at 0.5803 (2 clusters), 0.4283 for 10 clusters.

Ward Linkage:
Best at 0.5803 (2 clusters), solid up to 10 clusters (0.4251).

Contamination Level 0.1:
Single Linkage:
Peaks at 0.5923 for 2 clusters, drops to 0.0762 at 10 clusters.

Complete Linkage:
0.5923 (2 clusters), stable at 0.4355 (10 clusters).
Average Linkage:
0.5923 (2 clusters), declining to 0.4068 (10 clusters).

Ward Linkage:
Best at 0.5923 (2 clusters), stable at 0.4367 (10 clusters).

Contamination Level 0.2:
Single Linkage:
0.6134 (2 clusters), negative at 10 clusters (-0.0723).

Complete Linkage:
0.6134 (2 clusters), maintains 0.4127 (10 clusters).

Average Linkage:
0.6134 (2 clusters), drops to 0.4509 (10 clusters).

Ward Linkage:
Best at 0.6134 (2 clusters), 0.4457 for 10 clusters.

Key Insights:

- *Ward Linkage consistently performs best across contamination levels and clusters, especially with PCA 2 components.*
- *Single Linkage performs the worst, with negative scores at contamination 0.2 for higher clusters.*
- *PCA with 2 components consistently outperforms 3 components across all methods.*

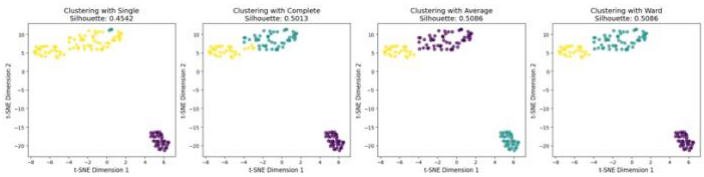


Figure 35- Differing clustering images for the same configuration for 3 Clusters.

The t-SNE visualizations of the Iris dataset show how four different linkage methods (Single, Complete, Average, Ward) perform in clustering. Silhouette scores range from 0.4542 (Single) to 0.5086 (Average/Ward), with Average and Ward linkage showing the best cluster separation and distinct grouping into three clusters. Single linkage, despite a reasonable score, shows overlapping points, while

Complete linkage offers better separation but still falls short compared to the clearer clusters seen with Average and Ward linkage.

3.8 Birch Iris Dataset

3.8.1 Initial Birch Results

For Iris, I applied threshold, and branching instantly to the dataset to reduce computational resources. Below is an example of the table given to me after I ran the parameter search for all combinations

Number of Clusters	Threshold	Branching Factor	Silhouette Score
2	0.1	20	0.5284
2	0.1	50	0.5284
2	0.1	100	0.5284
2	0.3	20	0.5284
2	0.3	50	0.5284
2	0.3	100	0.5284
2	0.5	20	0.5284
2	0.5	50	0.5284
2	0.5	100	0.5284
3	0.1	20	0.4473
3	0.1	50	0.4473
3	0.1	100	0.4473
3	0.3	20	0.4485

Figure 36- Small exert of the results from differing configurations prior to PCA & contamination

1. Consistent Performance for 2 Clusters:
Silhouette scores for 2 clusters remained stable at 0.5284, showing BIRCH can easily identify two distinct clusters across all thresholds and branching factors.

2. Declining Performance with More Clusters:
Silhouette scores decreased as cluster counts increased, particularly beyond 3 clusters. For 10 clusters, the highest score was 0.3551, indicating BIRCH struggles with higher cluster numbers.

3. Threshold Sensitivity:
Lower thresholds, like 0.1, performed better, especially for 3 and 4 clusters. For example, 3 clusters at 0.1 threshold yielded a score of 0.4473, compared to 0.4332 at a 0.5 threshold.

4. Minimal Impact of Branching Factor:
Changes in the branching factor (20 to 100) had little effect on silhouette scores, showing it is not a critical parameter.

In summary, BIRCH performs best with 2 or 3 clusters. Thresholds significantly affect performance, while branching factor has minimal impact.

3.8.2 PCA & Contamination

1. Stable 2-Cluster Performance:
Across contamination levels, 2 clusters consistently achieved high silhouette scores, with the highest being 0.6178 at a 0.2 contamination level.

2. Improved Performance with Contamination:
Higher contamination levels (from 0.05 to 0.2) generally increased scores. For example, with 3 clusters, scores improved from 0.5005 to 0.5276 as contamination increased.

3. Threshold Sensitivity:
Lower thresholds, like 0.1, provided slightly better results across contamination levels, but differences were small.

4. Decreasing Performance with More Clusters:
Beyond 6 clusters, performance dropped, with 10 clusters scoring 0.4313 despite contamination filtering.

5. Minimal Impact of Branching Factor:
Like the earlier results, branching factor had little effect on performance.

In conclusion, BIRCH clustering improves with contamination filtering, especially for 2 or 3 clusters, but performance drops significantly with more clusters. Thresholds play a key role, while branching factors remain less critical.

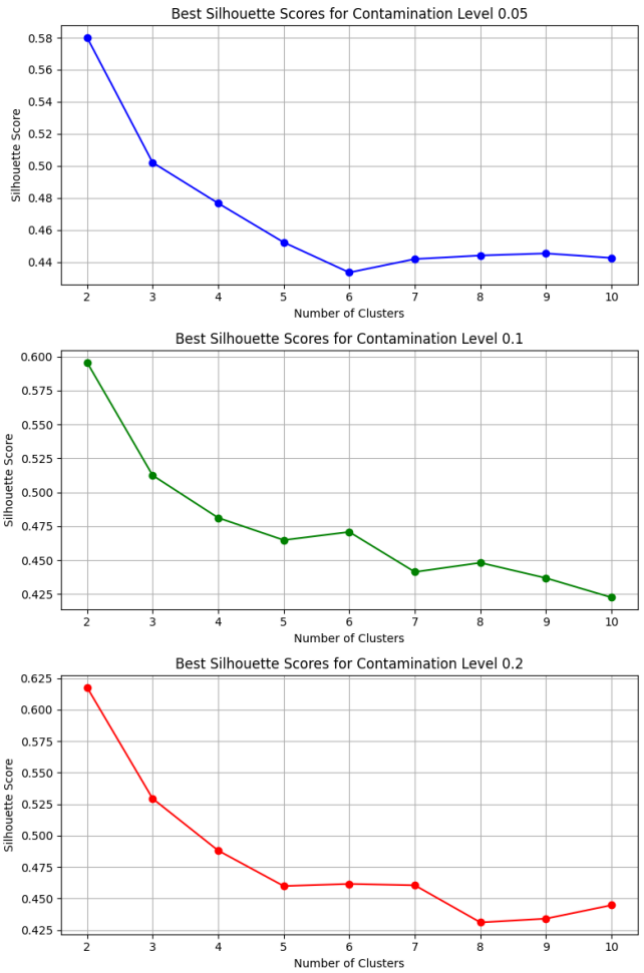


Figure 37- Difference in scores per contamination level

The provided graph and data showcase the effect of varying contamination levels (0.05, 0.1, and 0.2) on BIRCH clustering's silhouette scores across different numbers of clusters (from 2 to 10).

1. Contamination 0.05:
- *Best performance: 0.5802 (2 clusters).*
- *Performance decreases steadily to 0.4426 (10 clusters).*
- *Effective up to 5 clusters, but scores stabilize around 0.44 for higher cluster numbers.*

2. Contamination 0.1:

- *Higher initial score: 0.5958 (2 clusters).*
- *Sharp decline to 0.4225 (10 clusters).*
- *Slightly better than contamination 0.05 up to 5 clusters, then declines rapidly.*

3. Contamination 0.2:

- *Best overall score: 0.6178 (2 clusters).*
- *Gradual decline, reaching 0.4447 (10 clusters).*
- *Outperforms lower contamination levels at 2 clusters but converges with other levels for higher cluster counts.*

Key Takeaways:

- *Best Performance at 2 Clusters Contamination 0.2 offers the highest scores.*
- *Decline with More Clusters: Scores drop across all contamination levels as clusters increase, though higher contamination levels show a slower rate of decline.*
- *Optimal Range:*
- *The ideal cluster range for performance is 2 to 5 clusters, beyond which clustering quality diminishes.*

This analysis highlights that contamination filtering improves BIRCH clustering, particularly at 2 clusters, with performance tapering off as cluster count increases.

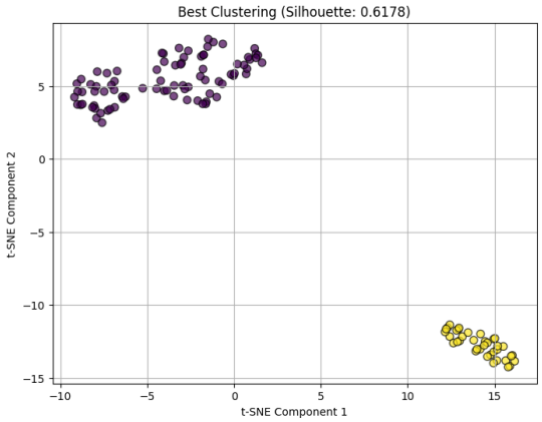


Figure 38- Best cluster silhouette score clustering

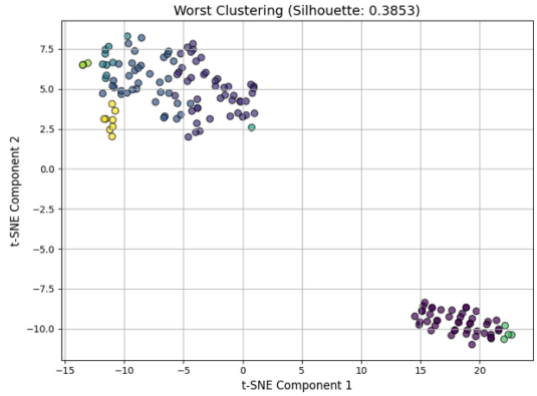


Figure 39- Worst Cluster Silhouette score clustering

The t-SNE visualizations highlight clear differences in clustering quality between the best and worst BIRCH configurations:

1. Best Clustering (Silhouette: 0.6178):
2 clusters, contamination level 0.2, and threshold 0.1 resulted in well-separated, tightly grouped clusters. Points within each cluster are closely aligned, indicating strong intra-cluster cohesion. The high silhouette score reflects the optimal parameter choice, yielding distinct, coherent clusters.

2. Worst Clustering (Silhouette: 0.3853):
8 clusters, contamination level 0.05, and threshold 0.5 produced overlapping, poorly defined clusters.

The boundary between clusters is unclear, with points assigned incorrectly, leading to weak intra-cluster cohesion. The lower silhouette score reveals the impact of suboptimal parameter settings on clustering quality.

This comparison illustrates that optimal parameter selection is critical for BIRCH clustering, significantly influencing cluster cohesion and separation.

3.9 Performance Evaulation/ Statistical Analysis

3.9.1 K-means

Iris Dataset (K-Means):

1. Silhouette Score:
Best configuration (2 clusters, PCA 2, contamination 0.2):
0.5856, showing well-separated clusters.

Worst configuration (10 clusters, PCA 2, contamination 0.05):
0.3234, indicating poor separation due to over-partitioning.

2. Davies-Bouldin Index:
Best configuration: 0.5982, reflecting good cluster separation.

Worst configuration: 0.9562, showing higher overlap between clusters.

3. Calinski-Harabasz Index:
Best configuration:
201.9560, indicating strong intra-cluster cohesion.

Worst configuration:
133.0991, reflecting weaker cluster separation with higher clusters.

4. Dunn Index:
Best configuration:
0.4237, showing well-defined clusters.

Worst configuration:
0.1132, highlighting poor separation with higher cluster counts.

Image Dataset (K-Means):

1. Silhouette Score:
Best configuration (3 clusters, PCA 2, contamination 0.05): 0.5856 for both training and validation, indicating strong cluster separation.
*Worst configuration (9 clusters, PCA 3, contamination 0.1):** Lower silhouette scores around 0.1-0.2, reflecting poor clustering performance.*

2. Davies-Bouldin Index:
Best configuration:
1.083 for training, 1.148 for validation, indicating well-separated clusters.

Worst configuration:
1.764 for validation, showing poorly defined clusters.

3. Calinski-Harabasz Index:

Best configuration:
479.7 for training, 465.1 for validation, showing cohesive clusters.

Worst configuration:
95.3 for training, 118.2 for validation, indicating weak cluster separation.

4. Dunn Index:
Best configuration:
0.0607 for training, consistent across best settings.

Worst configuration:
0.294 for training, highlighting poor separation.

Silhouette Distributions Summary:

Best configurations:
High silhouette scores around 0.6 in both datasets reflects well-defined clusters.

Worst configurations:
Lower or near-zero silhouette scores in worst settings show poor clustering and significant overlap (other silhouette distribution visuals in Appendix B).

The evaluation highlights that K-Means performs well with 2-3 clusters but deteriorates as cluster counts increase, particularly in the worst-case configurations, where higher complexity reduces clustering effectiveness.

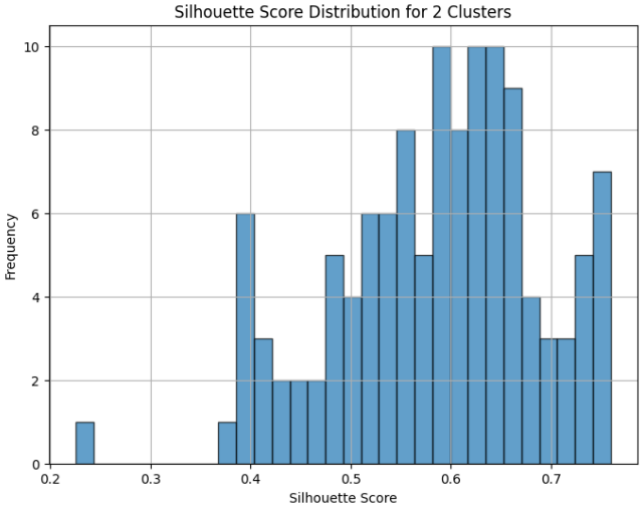


Figure 40- Silhouette Distribution graph for best clustering Iris (K-Means)

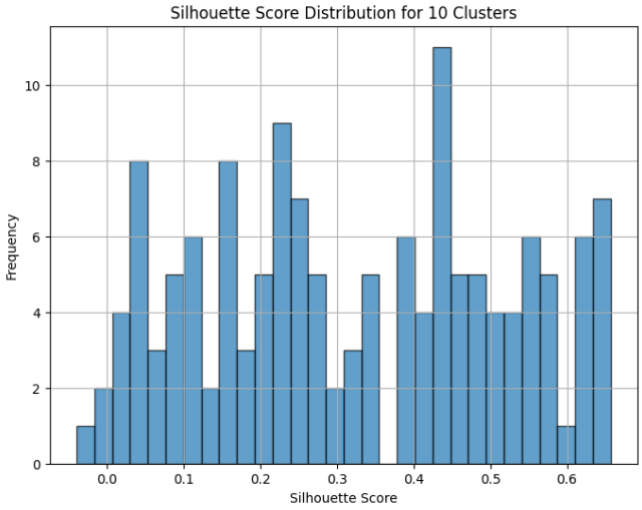


Figure 41- Silhouette Distribution graphs for Worst clustering Iris (K-Means)

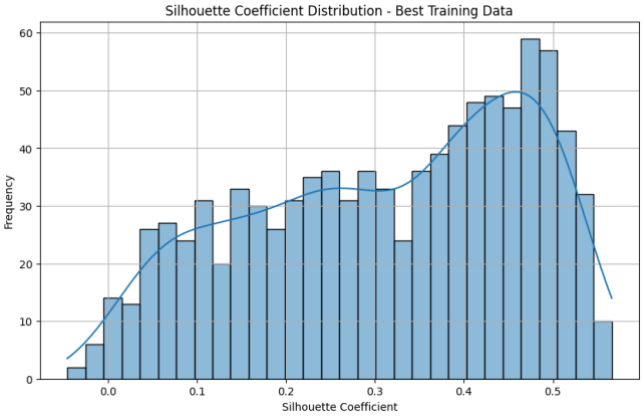


Figure 42- Silhouette Distribution Graph for best Training clustering Image Dataset (K-means)

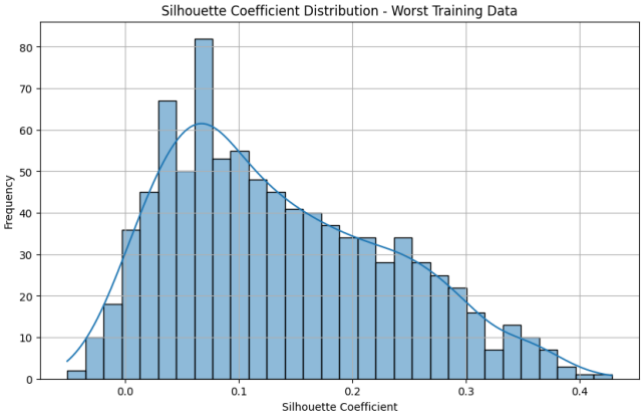


Figure 43- Silhouette Distribution Graph for Worst Training clustering Image Dataset (K-means)

3.9.2 AHC

Iris Dataset (AHC):

1. Silhouette Score:
Best configuration (2 clusters, complete linkage):

0.6134, indicating well-separated and cohesive clusters.

Worst configuration (10 clusters, single linkage):
0.0762, showing poor separation and overlapping clusters.

2. Davies-Bouldin Index:
Best configuration:
0.5556, reflecting strong cluster separation.

Worst configuration:
0.5598, showing minimal change but still poor separation in the worst case.

3. Calinski-Harabasz Index:
Best configuration:
223.3614, suggesting good intra-cluster cohesion.

Worst configuration:
45.4126, demonstrating poor cohesion with high cluster counts.

4. Dunn Index:
Best configuration:
2.4011, indicating clear cluster boundaries and separation.
Worst configuration:
0.4279, reflecting weak cluster separation and overlap.

Image Dataset (AHC):

1. Silhouette Score:
Best configuration (Ward linkage):
Scores concentrated between 0.4 and 0.6, suggesting relatively well-separated clusters.

Worst configuration (single linkage): Scores ranging between 0.4 and 0.0, indicating poor clustering.

2.Davies-Bouldin Index:
Best configuration:
0.8309 for training, 0.8989 for validation, showing well-separated clusters.

Worst configuration:
0.8821 for training, 0.8651 for validation, reflecting poorly defined clusters.

3. Calinski-Harabasz Index:
Best configuration:
727.033 for training, 528.595 for validation, indicating strong cluster cohesion.

Worst configuration:
2.88 for training, 1.54 for validation, showing very weak cluster separation.

4. Dunn Index:
Best configuration:
0.0317 for training, 0.0240 for validation, indicating moderate cluster separation.

Worst configuration:
0.0924 for training, 0.0835 for validation, highlighting poor cluster boundaries.

Silhouette Distributions Summary:
Best configurations:
High silhouette scores (around 0.6) in both datasets indicate well-defined and separated clusters.

Worst configurations:
Negative or close-to-zero silhouette scores reflect misclassified points and poor separation, particularly with single linkage in higher cluster counts.

This evaluation shows that Ward and complete linkage perform best, while single linkage struggles with larger cluster counts, leading to poor cluster separation and weak cluster boundaries.

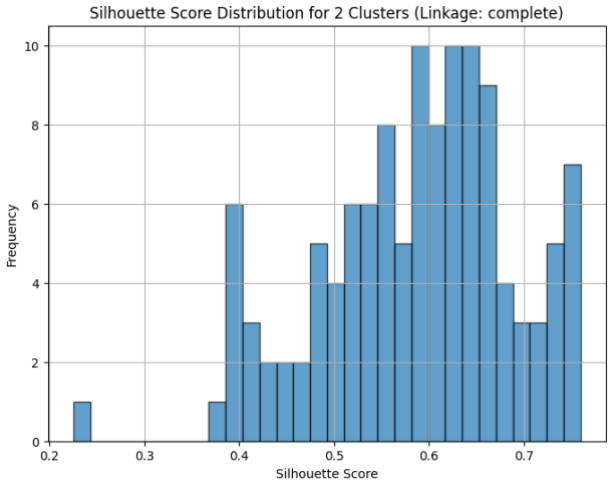


Figure 44- Silhouette Distribution graph for best clustering Iris (AHC)

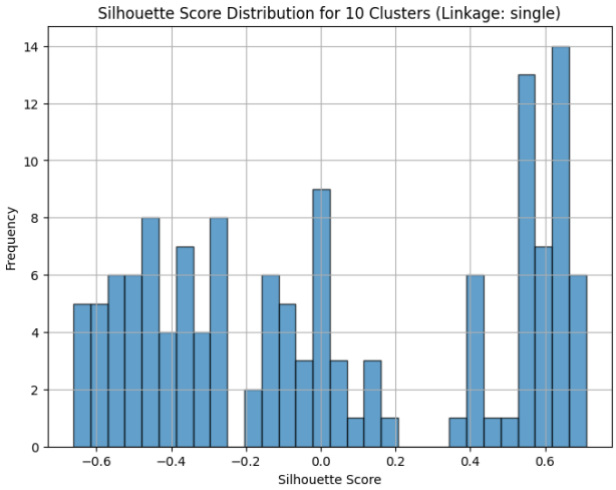


Figure 45- Silhouette Distribution graph for worst clustering Iris (AHC

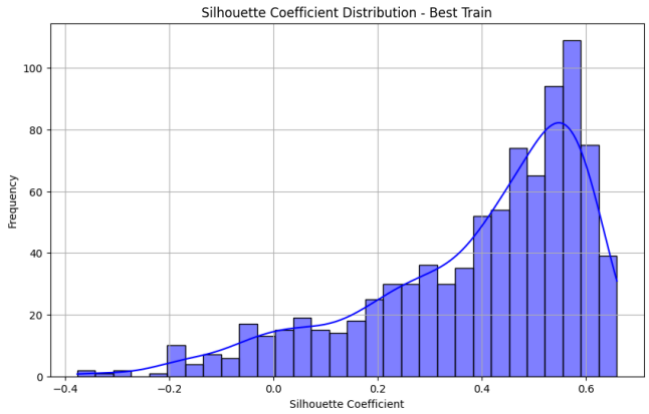


Figure 46- Silhouette Distribution graph for best train clustering Image dataset (AHC)

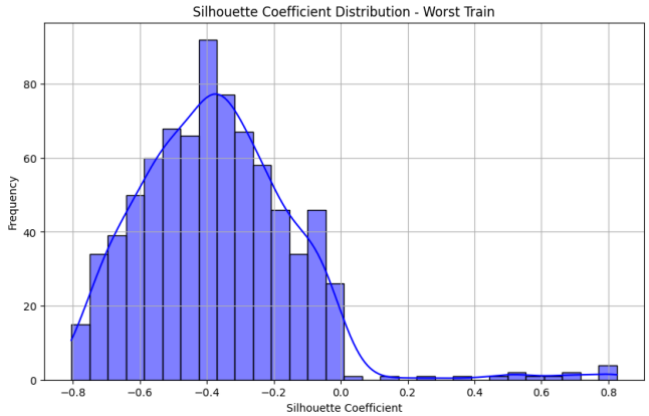


Figure 47- Silhouette Distribution graph for worst train clustering Image dataset (AHC)

3.9.3 BIRCH

Iris Dataset (BIRCH):

1. Silhouette Score:
Best configuration (2 clusters, 0.1 threshold, branching factor 20, contamination 0.2):
0.6178, indicating well-separated and cohesive clusters.

Worst configuration (9 clusters, 0.5 threshold, branching factor 20, contamination 0.05):
0.3835, showing weaker separation and overlap between clusters.

2. Davies-Bouldin Index:
Best configuration:
0.5492, reflecting good cluster separation.

Worst configuration:
0.6161, showing poorer separation between clusters.

3. Calinski-Harabasz Index:
Best configuration:
228.5621, suggesting strong intra-cluster cohesion.

Worst configuration:
220.6522, indicating a decline in clustering quality with higher clusters.

4. Dunn Index:
Best configuration:
0.4259, indicating clear cluster boundaries.

Worst configuration:
0.3023, reflecting weaker cluster separation and overlap.

Image Dataset (BIRCH):

1. Silhouette Score:
Best configuration (Training: 3 clusters, 0.1 threshold, branching factor 20):
0.3971, indicating moderate separation between clusters.

Worst configuration (Training: 10 clusters, 0.5 threshold, branching factor 20):
0.2716, showing poor separation with more overlapping clusters.

2. Davies-Bouldin Index:
Best configuration:
0.8593 for training, 0.8795 for validation, reflecting moderate cluster separation.

Worst configuration:
0.8228 for training, 0.8824 for validation, indicating little change but still poor clustering.

3. Calinski-Harabasz Index:
Best configuration:
586.2484 for training, 597.1095 for validation, showing relatively good intra-cluster cohesion.

Worst configuration:
589.0312 for training, 643.6985 for validation, indicating weaker clustering.

4. Dunn Index:
Best configuration: 0.0166 for training, 0.0060 for validation, showing weak cluster boundaries.

Worst configuration: 0.0186 for training, 0.0106 for validation, indicating poor separation between clusters.

Silhouette Distributions Summary:
Best configurations:
Silhouette scores are concentrated around 0.6 for Iris and around 0.4 for the image dataset, indicating well-defined clusters.

Worst configurations:
Scores range lower, particularly for the image dataset, with values close to 0.2, reflecting poor cluster separation, especially with higher cluster counts and thresholds.

This evaluation demonstrates that lower thresholds and fewer clusters tend to yield better clustering performance with BIRCH, while higher thresholds and more clusters result in weaker cluster separation and overlap.

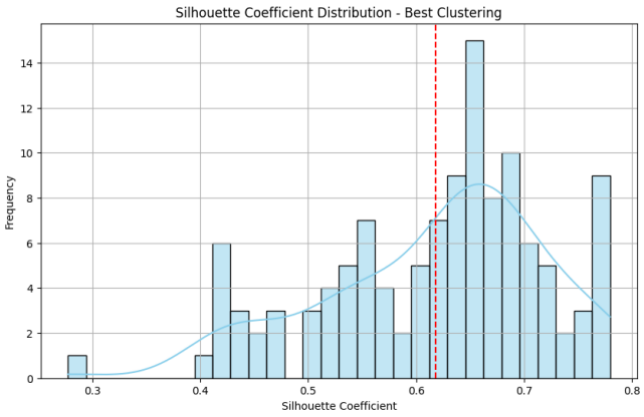


Figure 48- Silhouette Distribution graph for best clustering Iris (BIRCH)

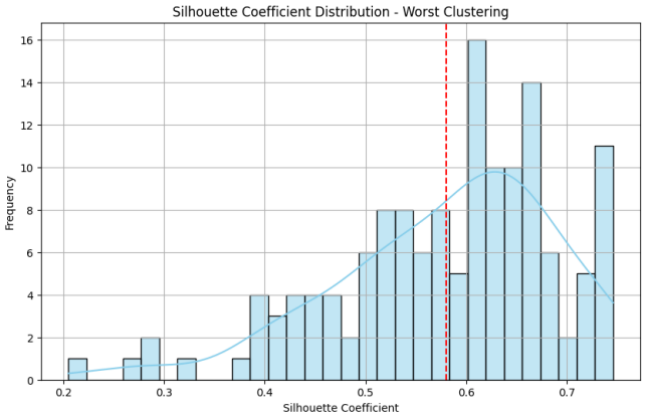


Figure 49- Silhouette Distribution graph for worst clustering Iris (AHC)

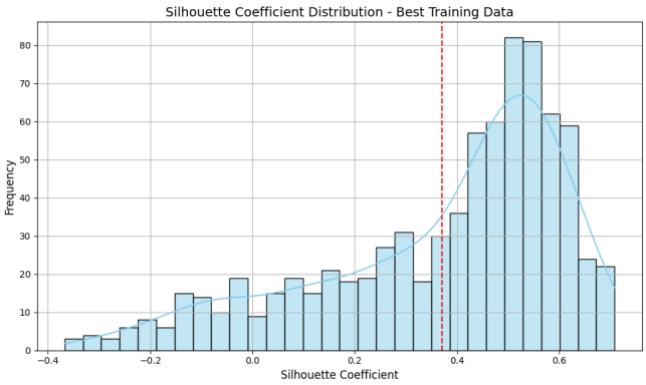


Figure 50- Silhouette Distribution graph for best train clustering Image dataset (BIRCH)

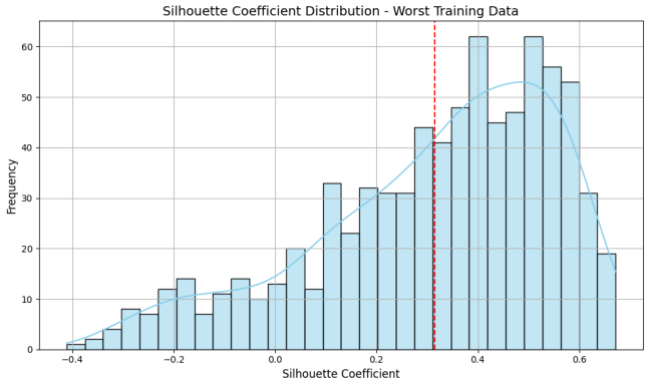


Figure 51- Silhouette Distribution graph for worst train clustering Image dataset (BIRCH)

3.10 Evaluation

Final Evaluation of Clustering Algorithms

1. K-Means Clustering:
Iris Dataset:
K-Means performed best with 2-3 clusters, achieving a silhouette score of 0.6178(PCA 2, contamination 0.2). Performance dropped significantly beyond 3 clusters, reflecting the dataset's inherent structure that supports fewer well-separated clusters.

Image Dataset:
The optimal configuration (3 clusters, PCA 2, contamination 0.05) yielded a silhouette score of 0.5856, but performance decreased with higher clusters (e.g., 9 clusters, PCA 3). Despite this, K-Means was the most effective algorithm overall for both datasets, with slightly better results on the simpler Iris dataset.

2. Agglomerative Hierarchical Clustering (AHC):
Iris Dataset:
AHC, particularly with Ward and Complete linkage, achieved a top silhouette score of 0.6134(contamination 0.2). However, its performance diminished with increasing clusters and single linkage, highlighting its limitations in handling more complex data structures.

Image Dataset:
AHC struggled more with the Image dataset, attaining a best silhouette score of 0.5856 with Ward linkage. Single linkage performed poorly as cluster counts increased, indicating that AHC handles lower-dimensional datasets better but struggles with more complex, high-dimensional data.

3. BIRCH Clustering:
Iris Dataset:
BIRCH performed adequately, with a best silhouette score of 0.6178 for 2 clusters (threshold 0.1). However, it struggled with higher cluster counts, dropping to 0.3835 in the worst configuration (9 clusters). BIRCH's ability to handle noise was an advantage but did not compensate for its challenges in separating clusters effectively.

Image Dataset:
BIRCH showed intermediate performance, with a best silhouette score of 0.3971 for 3 clusters. While more stable than AHC at higher clusters, it still struggled to separate clusters clearly in high-dimensional data, performing less effectively than K-Means.

IV. Conclusion & Future works

4.1 Conclusion on dataset performance & AI techniques

Overall, the Iris dataset consistently showed better clustering performance across all three algorithms—K-Means, AHC, and BIRCH—compared to the image dataset. The Iris dataset, being lower-dimensional and having well-defined clusters, allowed K-Means and AHC to achieve higher silhouette scores and stronger cluster separation. Specifically, K-Means achieved its best performance on the Iris dataset with a silhouette score of 0.6178, while AHC and BIRCH also showed comparable success. In contrast, the image dataset, with its higher dimensionality and more complex structure, presented challenges for all clustering methods, resulting in lower silhouette scores and weaker cluster cohesion. This difference demonstrates that unsupervised learning techniques are more efficient when applied to simpler, lower-dimensional data, while more complex datasets require additional strategies, such as more advanced dimensionality reduction methods or hybrid clustering approaches.

This project, titled "AI-Enhanced Clustering: Comparative Analysis of Unsupervised Learning on Multimodal Datasets" highlights the application of AI techniques, such as PCA for dimensionality reduction and contamination filtering, to enhance clustering performance. These techniques proved essential in handling the complex image dataset, where standard clustering methods alone struggled.

The project demonstrated the importance of incorporating AI-based techniques to optimize clustering outcomes, particularly for high-dimensional data, making a case for further exploration of hybrid and automated clustering approaches in future research.

4.2 Future works

While this dissertation demonstrates the effectiveness of unsupervised learning techniques such as K-Means, AHC, and BIRCH for clustering multimodal datasets, there remains significant room for improvement. Future research could investigate the integration of more advanced AI techniques like deep clustering frameworks [24], particularly leveraging autoencoders [25] and generative models like Variational Autoencoders (VAE) and Generative Adversarial Networks (GANs).It's important to note that in my project proposal I spoke on VAE's but in practical it was quite straining on my resources therefore it was scrapped. However, these approaches could enhance feature extraction and representation learning, resulting in improved clustering performance on complex datasets. Additionally, incorporating semi-supervised learning methods [26] that combine labeled and unlabeled data could further enhance clustering accuracy by using small amounts of annotated data to guide the clustering process.

Another promising direction for future research lies in the optimization of clustering hyperparameters through automated methods such as Neural Architecture Search (NAS)[27] or evolutionary algorithms[28]. These techniques could automate the discovery of optimal model configurations, reducing the time and expertise needed to fine-tune models. Finally, applying these techniques to larger, more diverse datasets with real-world applications, such as medical imaging or autonomous driving, could provide more insights into the robustness of these clustering methods in practical scenarios.

V. Declaration

I, Suhaib Mahamoud, declare I used AI tools for the sole use of proof reading and any other work has been explicitly referenced.

REFERENCES

[1] *Bishop, C.M. and Nasrabadi, N.M., 2006. Pattern recognition and machine learning (Vol. 4, No. 4, p. 738). New York: springer.*

[2] *Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.*

[3] *Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural networks, 61, pp.85-117.*

[4] *Jain, A.K., Murty, M.N. and Flynn, P.J., 1999. Data clustering: a review. ACM computing surveys (CSUR), 31(3), pp.264-323.*

[5] *Xu, R. and Wunsch, D., 2005. Survey of clustering algorithms. IEEE Transactions on neural networks, 16(3), pp.645-678.*

[6] *Macqueen, J., 1967. Some methods for classification and analysis of multivariate observations. In Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press.*

[7] *Rokach, L. and Maimon, O., 2005. Clustering methods. Data mining and knowledge discovery handbook, pp.321-352.*

[8] *Hinton, G.E., Osindero, S. and Teh, Y.W., 2006. A fast learning algorithm for deep belief nets. Neural computation, 18(7), pp.1527-1554.*

[9] *Jain, A.K., 2010. Data clustering: 50 years beyond K-means. Pattern recognition letters, 31(8), pp.651-666.*

[10] *Hennig, C., 2015. Clustering strategy and method selection. Handbook of cluster analysis, 9, pp.703-730.*

[11] *Ester, M., Kriegel, H.P., Sander, J. and Xu, X., 1996, August. A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).*

[12] *Hinneburg, A. and Gabriel, H.H., 2007, September. Denclue 2.0: Fast clustering based on kernel density estimation. In International symposium on intelligent data analysis (pp. 70-80). Berlin, Heidelberg: Springer Berlin Heidelberg.*

[13] *Sculley, D., 2010, April. Web-scale k-means clustering. In Proceedings of the 19th international conference on World wide web (pp. 1177-1178).*

[14] *Macqueen, J., 1967. Some methods for classification and analysis of multivariate observations. In Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press.*

[15] *Aggarwal, C.C., Hinneburg, A. and Keim, D.A., 2001. On the surprising behavior of distance metrics in high dimensional space. In Database theory—ICDT 2001: 8th international conference London, UK, January 4–6, 2001 proceedings 8(pp. 420-434). Springer Berlin Heidelberg.*

[16] *Tenenbaum, J.B., Silva, V.D. and Langford, J.C., 2000. A global geometric framework for nonlinear dimensionality reduction. science, 290(5500), pp.2319-2323.*

[17] *Van der Maaten, L. and Hinton, G., 2008. Visualizing data using t-SNE. Journal of machine learning research, 9(11).*

[18] *Rudin, C., 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature machine intelligence, 1(5), pp.206-215.*

[19] *Amid, E., Warmuth, M.K. and Srinivasan, S., 2019, April. Two-temperature logistic regression based on the tsallis divergence. In The 22nd International Conference on Artificial Intelligence and Statistics (pp. 2388-2396). PMLR.(Heres link to the full PDF https://arxiv.org/abs/1705.07210v2).*

[20] *He, H. and Garcia, E.A., 2009. Learning from imbalanced data. IEEE Transactions on knowledge and data engineering, 21(9), pp.1263-1284.*

[21] *Japkowicz, N. and Stephen, S., 2002. The class imbalance problem: A systematic study. Intelligent data analysis, 6(5), pp.429-449.*

[22] *Chawla, N.V., 2010. Data mining for imbalanced datasets: An overview. Data mining and knowledge discovery handbook, pp.875-886.*

[23] *Estabrooks, A. and Japkowicz, N., 2001, September. A mixture-of-experts framework for learning from imbalanced data sets. In International Symposium on Intelligent Data Analysis (pp. 34-43). Berlin, Heidelberg: Springer Berlin Heidelberg.*

[24] *Aljalbout, E., Golkov, V., Siddiqui, Y., Strobel, M. and Cremers, D., 2018. Clustering with deep learning: Taxonomy and new methods. arXiv preprint arXiv:1801.07648.*

[25] *Hinton, G.E. and Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. science, 313(5786), pp.504-507.*

[26] *Rasmus, A., Berglund, M., Honkala, M., Valpola, H. and Raiko, T., 2015. Semi-supervised learning with ladder networks. Advances in neural information processing systems, 28.*

[27] *Zoph, B., 2016. Neural architecture search with reinforcement learning. arXivpreprintarXiv:1611.01578. (https://arxiv.org/pdf/1611.01578)*

[28] *ogel, D.B., 2006. Evolutionary computation: toward a new philosophy of machine intelligence. John Wiley & Sons.*

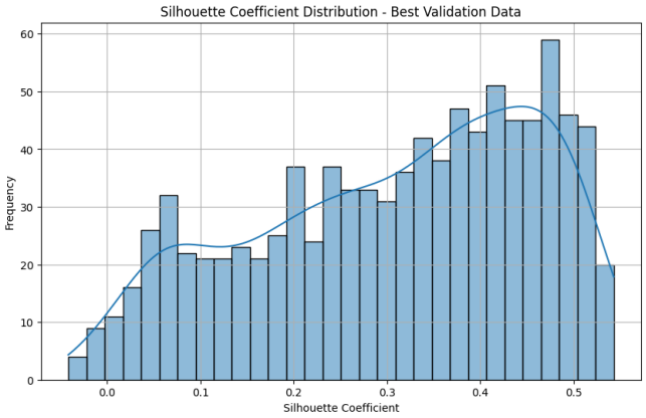
APPENDICES

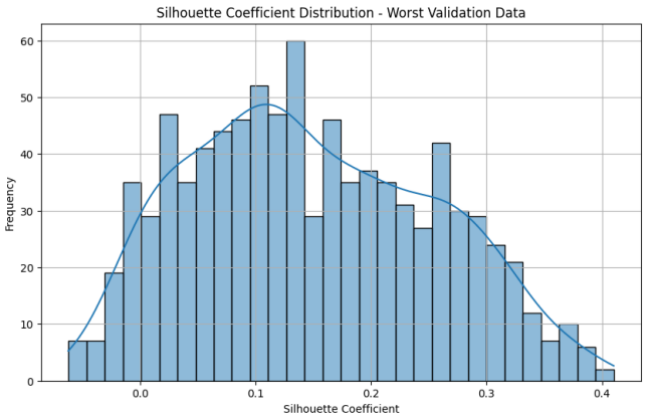
Appendix A

```
# Data augmentation and preprocessing for training data
train_datagen_unlabelled = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,  # Augmentation : rotate
    width_shift_range=0.2, # Augmentation:
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)

# Data preprocessing for validation data
val_datagen_unlabelled = ImageDataGenerator(rescale=1./255)

# Load and augment training data without labels
train_generator = train_datagen.flow_from_directory(
    train_data,
    target_size=IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode=None  # No labels
)

# Load and preprocess validation data without labels
val_generator = val_datagen.flow_from_directory(
    val_data,
    target_size=IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode=None  # This is important as it will bring back only the images rather than with their coresponding class, THis is crucial fo
)
```

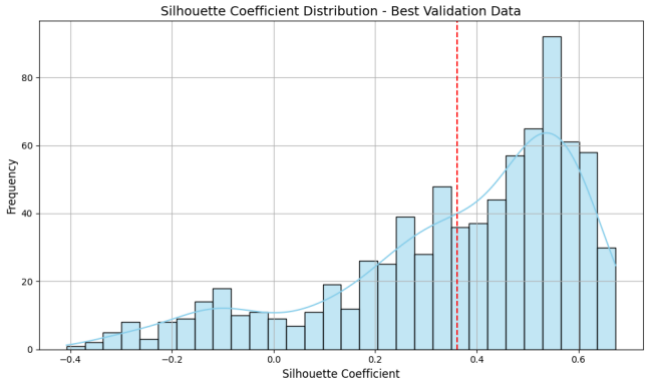
Appendix B-

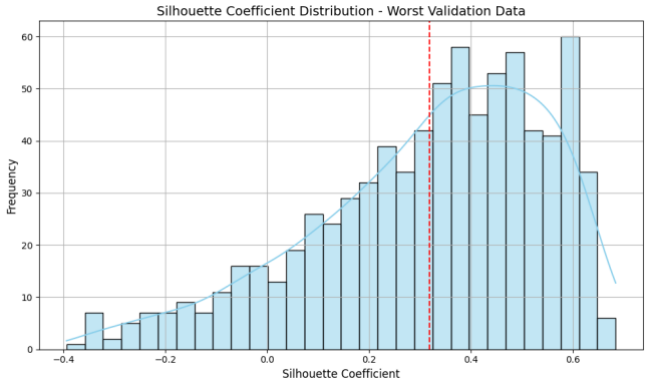
These two images below are the images for silhouette distribution for k-means image dataset validation



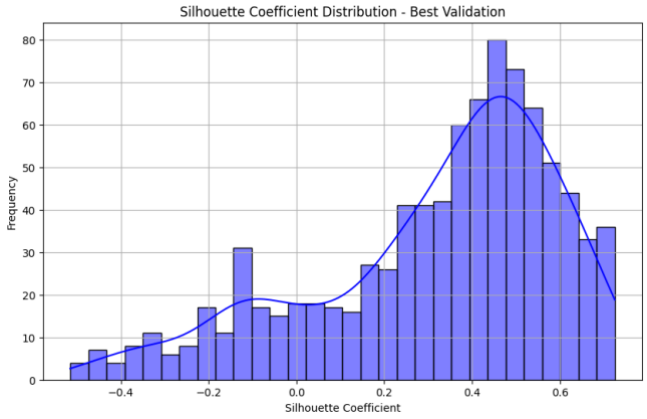


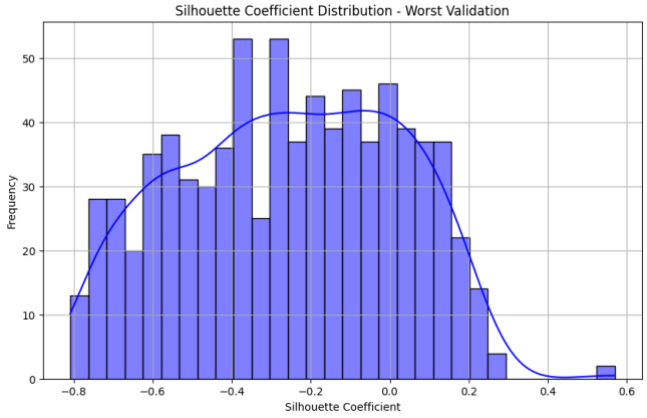
These images below are the silhouette distribution for validation BIRCH image dataset





These two images are below for silhouette distribution for validation AHC image dataset





IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published