UNSUPERVISED LEARNING - CLUSTERING

Kamanis Fotios (aid24010)
Kouti Aikaterini Maria (aid24007)

University of Macedonia
MSc in Artificial Intelligence and Data Analysis

December 2023

Abstract

This report presents a comprehensive exploration of the Fashion-MNIST dataset using various clustering algorithms. We investigate the efficacy of five dimensionality reduction techniques - Principal Component Analysis (PCA), Stacked Autoencoders (SAE), Convolutional Stacked Autoencoders (CNN-SAE), Linear Discriminant Analysis (LDA), and Uniform Manifold Approximation and Projection (UMAP) - followed by five clustering algorithms - MiniBatch K-Means, DBSCAN, Gaussian Mixture Models, Hierarchical Clustering, and Spectral Clustering. Our analysis emphasizes the critical synergy between dimensionality reduction techniques and clustering algorithms, demonstrating how dimensionality reduction facilitates the extraction of meaningful patterns from complex datasets like Fashion-MNIST, thereby significantly enhancing clustering performance. The performance of these combinations is evaluated using four key metrics: Calinski-Harabasz Index, Davies-Bouldin Index, Silhouette Score, and Adjusted Rand Index. The findings reveal that UMAP, combined with Hierarchical Clustering, emerges as the most effective approach for this dataset, providing a balance between computational efficiency and clustering performance. The report concludes with insights into the suitability of these techniques for different data types and objectives, highlighting the importance of tailored method selection in machine learning applications.

Contents

1 Introduction

This report presents a comprehensive analysis of the Fashion-MNIST dataset using clustering, an unsupervised learning technique. Our objective is to explore the efficacy of various dimensionality reduction techniques followed by clustering algorithms to uncover inherent groupings within the data. Our investigation is based on the premise that dimensionality reduction can significantly enhance the clustering process by distilling the essence of high-dimensional data into a more manageable form.

The Fashion-MNIST dataset, a modernized alternative to the classic MNIST dataset, consists of 28x28 grayscale images representing 10 different fashion categories. It poses a challenge for machine learning models due to its intricate patterns and high-dimensional nature. Ensuring a balanced class representation in the dataset is essential to prevent model bias and is addressed through our data preparation process.

To elucidate the underlying structure of the Fashion-MNIST dataset, we have implemented and assessed a suite of dimensionality reduction techniques. These include Principal Component Analysis (PCA), Stacked Autoencoders (SAE), Convolutional Stacked Autoencoders (CNN-SAE), Linear Discriminant Analysis (LDA), and Uniform Manifold Approximation and Projection (UMAP). The employment of these methods aims to reduce the complexity of the data while preserving as much of the significant information as possible, thus facilitating a more effective clustering experience.

Following the dimensionality reduction, our analysis incorporates a diverse array of clustering algorithms to categorize the transformed data. We explore the capabilities of MiniBatch K-Means, DBSCAN, Gaussian Mixture Models (GMM), Hierarchical Clustering and Spectral Clustering.

The efficacy of these clustering methods is evaluated through a spectrum of performance metrics, such as the Calinski–Harabasz Index, Davies–Bouldin Index, Silhouette Score and Adjusted Rand Index. We intend to dissect the clustering outcomes, employing these metrics to provide an incisive evaluation of each algorithm's performance.

Our methodical comparison is designed to reveal the unique capabilities and limitations of these algorithms when confronted with the high-dimensional space of fashion imagery. Through this exploration, we aspire to uncover the optimal strategies for uncovering latent patterns within the Fashion-MNIST dataset, thereby contributing valuable insights to the field of unsupervised learning in machine vision.

In the subsequent sections of this report, we will comprehensively explore the various aspects of our analysis. We begin with an outline of our methodologies for dimensionality reduction and clustering, along with a detailed description of the performance metrics employed for evaluation. Subsequently, we conduct an initial exploration of the dataset, establishing a foundation for our analysis, and detail the steps taken in preparing the data. Afterwards, we provide the implementation details of the dimensionality reduction techniques and the subsequent application of clustering algorithms. This section also includes a presentation of the results obtained from different methods, grounded in our predefined performance metrics. Following this, we embark on a detailed comparative analysis of the clustering outcomes. The report concludes with a summary of our pivotal findings, alongside a discussion of their broader implications.

2 Methodology

We will now delve into the project's methodology. This section clarifies the theoretical foundations of the dimensionality reduction and clustering techniques applied, along with a rationale for the metrics selected to assess the analysis in subsequent stages.

2.1 Dimensionality Reduction

Dimensionality reduction plays a pivotal role in data preprocessing and it is a crucial part of the analysis process. Its importance is multifold:

- **Reduction of Data Complexity:** High-dimensional data often contains a lot of redundant or irrelevant information. Dimensionality reduction techniques help in distilling this data to its most informative features, thereby simplifying the dataset without losing critical information.
- **Mitigation of the Complexities of Dimensionality:** In high-dimensional spaces, the volume of the space increases so rapidly that the available data become sparse. This sparsity is problematic for any method that requires statistical significance. By reducing the number of dimensions, we mitigate these issues, making patterns more discernible and accessible to analysis.
- **Enhanced Algorithm Efficiency:** Many algorithms struggle with the computational complexity introduced by high-dimensional data. Reducing the dimensionality can significantly decrease the computational cost and improve the algorithm's efficiency.
- **Improved Visualization:** Reducing dimensions to 2D or 3D makes it possible to visualize the data, providing insights into the data distribution and inherent groupings.

In the context of clustering, dimensionality reduction is particularly important:

- **Enhancing Clustering Performance:** Clustering algorithms often perform better on datasets with fewer dimensions. This is because in lower-dimensional space, the inherent structures of the data are easier to identify, and noise is reduced.
- **Uncovering Hidden Patterns:** Dimensionality reduction can reveal underlying structures in the data that may not be apparent in higher dimensions. These structures can guide the clustering algorithm to more natural and meaningful groupings.
- **Balancing Variance and Loss of Information:** Effective dimensionality reduction strikes a balance between retaining significant variance in the data (to maintain its structure) and removing noise or less informative features that might lead to poor clustering.
- **Facilitating Interpretability:** By reducing the number of features, the results of clustering become more interpretable. It is easier to understand the role of each feature in the clustering process when there are fewer, more significant features.

In this analysis, dimensionality reduction was initially applied to the training set and then extended to the test set, ensuring a uniform approach to data representation across both sets. The reduced test data were subsequently utilized in the clustering algorithms. This step was crucial to leverage the benefits of dimensionality reduction, previously outlined, thereby enhancing the performance of these algorithms.

Five dimensionality reduction techniques were applied and evaluated:

1. Principal Component Analysis (PCA)
2. Stacked Autoencoder (SAE)
3. Convolutional Stacked Autoencoder (CNN-SAE)
4. Linear Discriminant Analysis (LDA)
5. Uniform Manifold Approximation and Projection (UMAP)

Each technique has its own advantages and is selected to provide a comprehensive understanding of the data structure.

2.1.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique used to transform the original features of a dataset into a set of uncorrelated features, known as principal components. These components are linear combinations of the original features and are designed to capture the maximum variance present in the data.

By focusing on the most informative principal components, PCA enables the reduction of dataset dimensionality while retaining the essential patterns and structures. This reduction is achieved by selecting the top k principal components that collectively explain the majority of the variance in the data, resulting in a more concise and efficient representation of the dataset, particularly beneficial when working with high-dimensional data.

Advantages

- Simplicity and efficiency.
- No need for labeled data.
- Possible interpretability of components.

Disadvantages

- Assumes linear relationships.
- Sensitive to variable scaling.
- May not capture complex structures.

2.1.2 Stacked Autoencoder (SAE)

In a regular Stacked Autoencoder (SAE), multiple layers of neurons are stacked, forming an encoder-decoder structure. The encoder transforms the input data into a compressed representation at the bottleneck layer, and the decoder reconstructs the input from this representation.

During training, the network learns to capture essential features of the input data in the encoded representation. This can lead to a more concise and informative representation of the images on our dataset.

Advantages

- Captures non-linear relationships.
- Automatic feature learning.
- Robustness to noise.

Disadvantages

- Risk of overfitting.
- Computationally intensive.
- Requires ample training data.

2.1.3 Convolutional Stacked Autoencoder (CNN-SAE)

Convolutional Stacked Autoencoder (CNN-SAE) is a neural network architecture that combines the principles of Stacked Autoencoders (SAE) with convolutional layers commonly used in Convolutional Neural Networks (CNN). This hybrid architecture can be particularly effective for our problem, since it is excelling in tasks involving image data.

Advantages

- Efficient spatial feature extraction.
- Scalable for large-sized inputs.
- Enhanced feature representation.

Disadvantages

- Increased complexity.
- High computational demands.
- Suited mainly for spatial/temporal structured data.

2.1.4 Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis is a dimensionality reduction technique with a specific focus on preserving class-related information. In LDA, the algorithm works by finding linear combinations of features that maximize the separation between classes while minimizing the variance within each class. The transformed data, projected onto a lower-dimensional space, retains the most discriminative information for classification purposes. By capturing the inherent structure that distinguishes different classes, LDA provides a compact representation of the data.

Advantages

- Efficient for classification tasks.
- Emphasizes class-related information.
- Reduces dimensionality while preserving discriminative features.

Disadvantages

- Assumes normally distributed data.
- Sensitive to outliers.
- May not perform well with small sample sizes.

The selection of Linear Discriminant Analysis (LDA) for our analysis was driven by its specialized focus on class separation. LDA is adept at identifying and enhancing features crucial for distinguishing between different classes, a capability particularly relevant for the Fashion-MNIST dataset. In this context, LDA's methodology of finding linear combinations of features that maximize class separation while minimizing variance within each class is expected to facilitate more distinct clustering of fashion items based on their intrinsic characteristics. Although LDA assumes normally distributed data and may be sensitive to outliers, its ability to highlight class-related information renders it a valuable technique for dimensionality reduction in our dataset.

2.1.5 Uniform Manifold Approximation and Projection (UMAP)

UMAP, or Uniform Manifold Approximation and Projection, is a powerful dimensionality reduction technique used for visualizing high-dimensional data in a lower-dimensional space, typically two or three dimensions. It excels at preserving the local structures and relationships within the data. UMAP aims to maintain the geometry of the original data manifold, making it effective for revealing complex patterns and clusters.

Advantages

- Captures complex local structures in high-dimensional data.
- Provides a more faithful representation of data manifold.
- Well-suited for visualizing intricate patterns in datasets.

Disadvantages

- Sensitivity to hyperparameter tuning.
- Limited interpretability of the embedding.

Uniform Manifold Approximation and Projection (UMAP) was chosen for its unparalleled ability to capture and preserve intricate local structures within the high-dimensional Fashion-MNIST dataset. This capability is crucial for effectively revealing the complex and nuanced patterns inherent in fashion imagery. UMAP stands out in its ability to maintain the local geometry and relationships of the data manifold in a lower-dimensional space, making it particularly adept at identifying natural clusters and facilitating a more profound understanding of the dataset's structure.

In contrast to LDA's focus on linear separability, UMAP specializes in handling non-linear relationships, making it adept at uncovering both linear and non-linear structures within the dataset. This characteristic is expected to lead to more accurate and meaningful clustering results. Our hypothesis posits that UMAP, with its advanced manifold learning capabilities, will offer significant improvements in clustering performance. By maintaining the intrinsic geometry of the data, UMAP is anticipated to enable the discovery of more coherent and distinct clusters, making it a prime technique for achieving superior outcomes in our clustering analysis of the Fashion-MNIST dataset.

After the implementation of the aforementioned dimensionality reduction techniques, the dataset's intrinsic dimensionality has been reduced, facilitating efficient clustering in subsequent steps.

2.2 Clustering Techniques

In this part of the analysis, we employed five distinct clustering techniques to explore patterns within the dataset. The clustering techniques applied are the following:

- MiniBatch K-Means
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
- Gaussian Mixture Model (GMM)
- Hierarchical Clustering
- Spectral Clustering

Below is a summary of the clustering techniques utilized in our analysis, along with an examination of their respective advantages and disadvantages.

2.2.1 MiniBatch K-Means

MiniBatch K-Means is an efficient variant of the K-Means algorithm, particularly suited for large datasets. It partitions data into clusters by iteratively updating centroids based on subsets (batches) of the input. By using random subsamples, it speeds up the convergence process while maintaining similar clustering performance as the standard K-Means. The trade-off between computational efficiency and clustering performance makes MiniBatch K-Means a pragmatic choice, particularly when dealing with the computational complexities associated with high-dimensional image datasets.

Advantages

- Efficient for large datasets.
- Accelerated convergence..
- Suitable for scenarios with computational constraints

Disadvantages

- Sensitive to the choice of batch size.
- May be less accurate than standard K-Means in some cases.

2.2.2 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) excels in identifying clusters based on the density of data points. It distinguishes between core points, border points, and noise, making it effective in discovering clusters of varying shapes and sizes. Its ability to handle noise and automatically detect the number of clusters without prior specification contributes to its versatility.

Advantages

- Resilient to noise.
- Automatic determination of cluster count.

Disadvantages

- Sensitive to parameter settings.
- May struggle with varying density levels in the data.

2.2.3 Gaussian Mixture Model

Gaussian Mixture Model (GMM) assumes that data points are generated from a mixture of Gaussian distributions, providing a probabilistic and flexible approach to clustering. It models clusters as ellipsoids, accommodating more complex cluster structures compared to K-Means. GMM is a probabilistic model, providing soft assignments of data points to clusters and allowing for the expression of uncertainty in the assignment.

Advantages

- Models complex cluster structures.
- Provides soft assignments.
- Suitable for datasets with overlapping clusters.

Disadvantages

- Computationally demanding.
- Sensitive to initialization parameters.

For the Gaussian Mixture Model (GMM), its inclusion in the clustering techniques was motivated by its capacity to model complex cluster structures and provide soft assignments of data points. GMM assumes that data points are generated from a mixture of Gaussian distributions, allowing it to accommodate more intricate relationships within the data. The probabilistic nature of GMM enables the expression of uncertainty in cluster assignments, which is particularly beneficial when dealing with the diverse and overlapping nature of fashion items in the Fashion-MNIST dataset.

2.2.4 Hierarchical Clustering

Hierarchical Clustering, a versatile method for unveiling hierarchical relationships within data, constructs a tree-like structure of clusters through iterative merging or splitting. Its capacity to provide insights into different levels of granularity makes it valuable for understanding complex data structures, a characteristic beneficial for our diverse image dataset. It is particularly useful when the underlying structure of the data includes a nested or hierarchical organization.

Advantages

- Reveals hierarchical relationships.
- Provides a visual representation of clustering.
- Suitable for nested or hierarchical data structures.

Disadvantages

- Computationally intensive.
- Dendrogram interpretation subject to user bias.

The application of Hierarchical Clustering creates a dendrogram, which visually represents relationships between data points at different levels of granularity. The interpretation of a dendrogram is quite straightforward:

- Vertical Lines: Vertical lines represent clusters that are being merged or split. The height of the vertical line indicates the distance or dissimilarity between clusters being joined. In other words, the taller the vertical line, the less similar the joined clusters are.
- Horizontal Lines: Horizontal lines represent the merged clusters. The length of the horizontal line doesn't have a special meaning in most dendrogram visualizations.
- Color Coding: It typically signifies different cluster groups according to some distance threshold. The colors may change at different levels to show how clusters at a certain distance are grouped together.
- Leaf Nodes: Each leaf node at the bottom represents one data point. When the leaf nodes are connected directly by a vertical line, they belong to the same cluster. When a vertical line connects two clusters, it means these clusters are merged at that level.

It is crucial to consider the computational demands associated with hierarchical clustering, especially when dealing with large datasets. The interpretation of dendrogram structures introduces a subjective element, requiring careful consideration to avoid biases. Despite these challenges, the ability to reveal nested or hierarchical organization within the data positions hierarchical clustering as a valuable tool, and its judicious application can yield meaningful insights into the structure of the Fashion-MNIST dataset.

Hierarchical Clustering was selected due to its versatility in unveiling hierarchical relationships within the data. The Fashion-MNIST dataset, encompassing diverse fashion items, may exhibit a nested or hierarchical organization that can be effectively explored using this method. Despite the computational intensity and the subjectivity introduced by dendrogram interpretation, the ability of Hierarchical Clustering to reveal hierarchical relationships and provide a visual representation aligns with the exploration of complex structures within the Fashion-MNIST dataset.

2.2.5 Spectral Clustering

Spectral Clustering is a technique based on the spectrum (eigenvalues) of the similarity matrix of the data. It is particularly effective for identifying clusters that are not necessarily globular and can capture complex structures within the data. For the Fashion-MNIST dataset, Spectral Clustering can be particularly useful in discerning subtle differences among various fashion items, which may not be linearly separable. However, like many clustering algorithms, it requires careful tuning of parameters and can be computationally intensive for large datasets.

Advantages

- Effective in identifying non-globular clusters.
- Capable of capturing complex structures within data.
- Suitable for datasets where traditional clustering approaches like K-Means may fail.

Disadvantages

- Requires selection of an appropriate similarity measure.
- Computationally intensive, especially for large datasets.
- The number of clusters needs to be specified in advance, which can be challenging without prior knowledge of the data.

Spectral Clustering was chosen as a clustering technique due to its effectiveness in identifying non-globular clusters and capturing complex structures within the data. The Fashion-MNIST dataset, comprising various fashion items, may involve subtle differences that traditional clustering approaches like K-Means struggle to discern. Spectral Clustering, based on the spectrum of the similarity matrix, excels in situations where clusters are not necessarily globular. It is particularly useful when dealing with datasets where linear separability is challenging. The ability of Spectral Clustering to handle complex structures in the data makes it a valuable tool for exploring the nuanced patterns within the Fashion-MNIST dataset.

2.3 Performance Metrics

In our analysis of clustering models applied to the Fashion-MNIST dataset, a comprehensive evaluation was conducted using various metrics, each offering unique insights into the quality of the clustering results. The metrics employed are:

- Calinski–Harabasz Index
- Davies–Bouldin Index
- Silhouette Score
- Adjusted Rand Index

These metrics collectively assess factors like cohesion, separation, and the accuracy of clustering in relation to the known labels of the dataset.

2.3.1 Calinski–Harabasz Index

The Calinski–Harabasz Index measures the quality of clustering by calculating the ratio of the sum of between-clusters dispersion and of within-cluster dispersion for all clusters. The higher the value, the more distinct the clusters are, with a high inter-cluster variance and low intra-cluster variance, indicating well-separated and compact clusters.

2.3.2 Davies–Bouldin Index

The Davies–Bouldin Index evaluates the average 'similarity' between clusters, where the similarity is a measure that compares the distance between clusters with the size of the clusters themselves. A lower Davies-Bouldin Index relates to a model with better separation between the clusters.

2.3.3 Silhouette Score

The Silhouette Score measures how similar a data point is to its own cluster compared to other clusters. The score ranges from -1 to +1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If many points have a high value, the clustering configuration is appropriate.

2.3.4 Adjusted Rand Index

The Adjusted Rand Index (ARI) evaluates the similarity between the true classification and the clustering outcome, adjusting for the chance grouping of elements. It considers all pairs of samples and counts pairs that are assigned in the same or different clusters in the predicted and true clusters. Higher values of ARI indicate that the clustering algorithm has effectively identified the true underlying groupings in the dataset.

The selection of ARI as our metric of choice is due to its ability to provide a clear and quantifiable measure of how well the clustering matches the actual distribution of fashion items across various categories. This is especially significant in a dataset like Fashion-MNIST, where each image is labeled with its actual class, enabling us to leverage ARI as a performance metric. The presence of these labels allows for a direct comparison between the clusters formed by the algorithm and the true categories, offering a robust assessment of the clustering effectiveness in discerning the inherent groupings within the data.

2.3.5 Summary

The metrics outlined above collectively provide a comprehensive assessment of the clustering quality, each offering distinct insights into different aspects of cluster formation.

The Calinski-Harabasz Index is particularly useful for gauging the definition and distinctness of clusters. A higher value of this index suggests that the clusters are well-separated and clearly defined, indicating a successful clustering outcome where each cluster is distinct from others in terms of variance.

On the other hand, the Adjusted Rand Index (ARI) is crucial for evaluating the accuracy of the clustering process in relation to the true labels in the dataset. The ARI is a measure of the similarity between the ground truth and the clustering outcome.

Finally, lower Davies-Bouldin Index values and higher Silhouette Scores are indicative of tighter and more separated clusters, emphasizing cluster compactness and clear delineation between different cluster groupings.

3 Data Exploration and Preparation

3.1 Dataset Overview

The Fashion-MNIST dataset, accessed through the Keras library of TensorFlow, provides a collection of 28x28 grayscale images categorized into 10 different types of fashion items. Below is a figure illustrating sample images from the dataset:

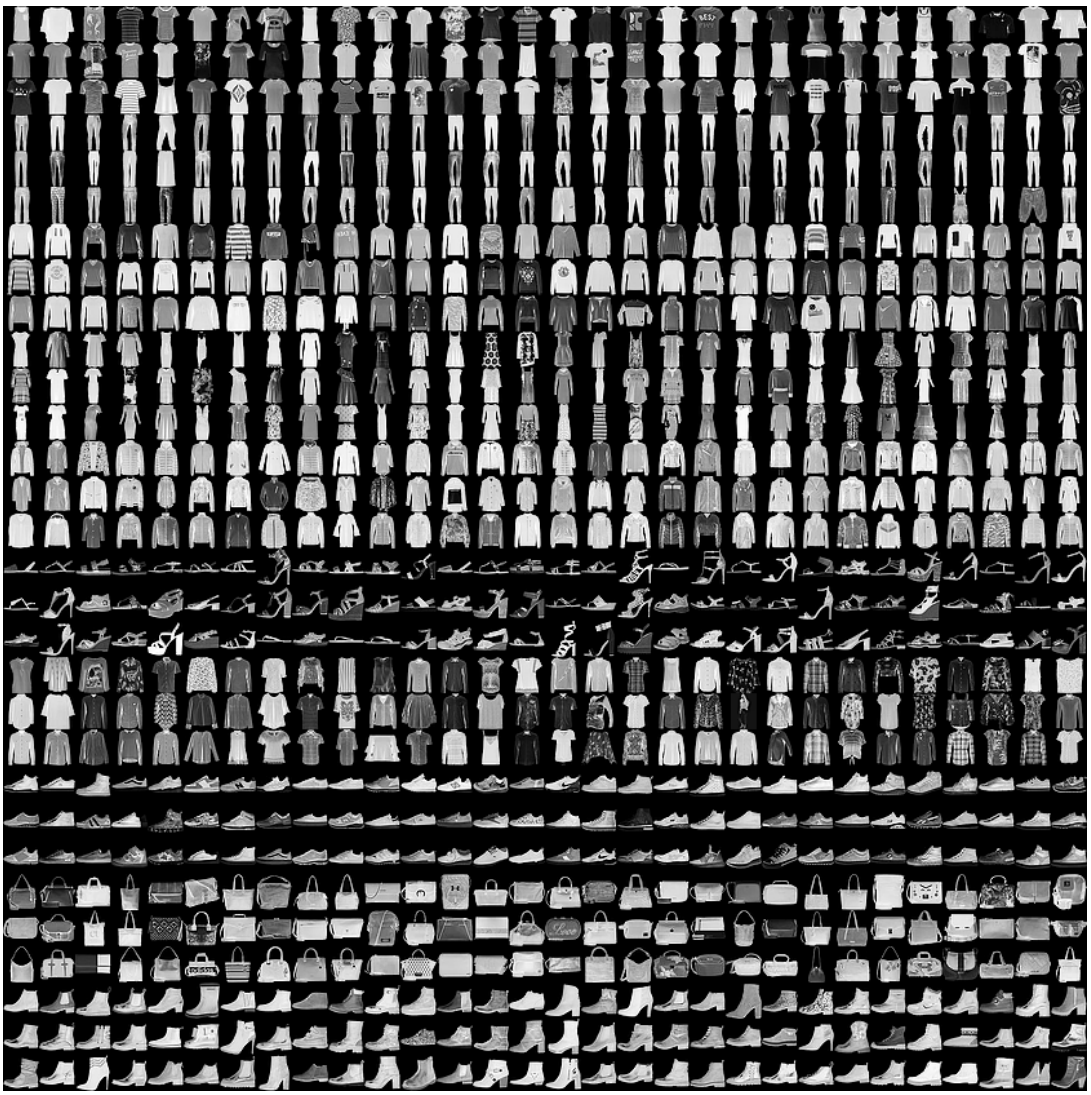


Figure 1: Fashion-MNIST Dataset Overview

The diverse classes present in Fashion-MNIST include T-shirts/tops, Trousers, Pullovers, Dresses, Coats, Sandals, Shirts, Sneakers, Bags, and Ankle boots. This diversity of fashion items introduces challenges for machine learning models due to the intricate patterns and styles that must be discerned from high-dimensional image data.

The dataset comprises 60,000 training images and 10,000 test images, each representing a unique item of clothing or accessory. Below are the bar plots that visually represent the distribution of classes across the training and test datasets:

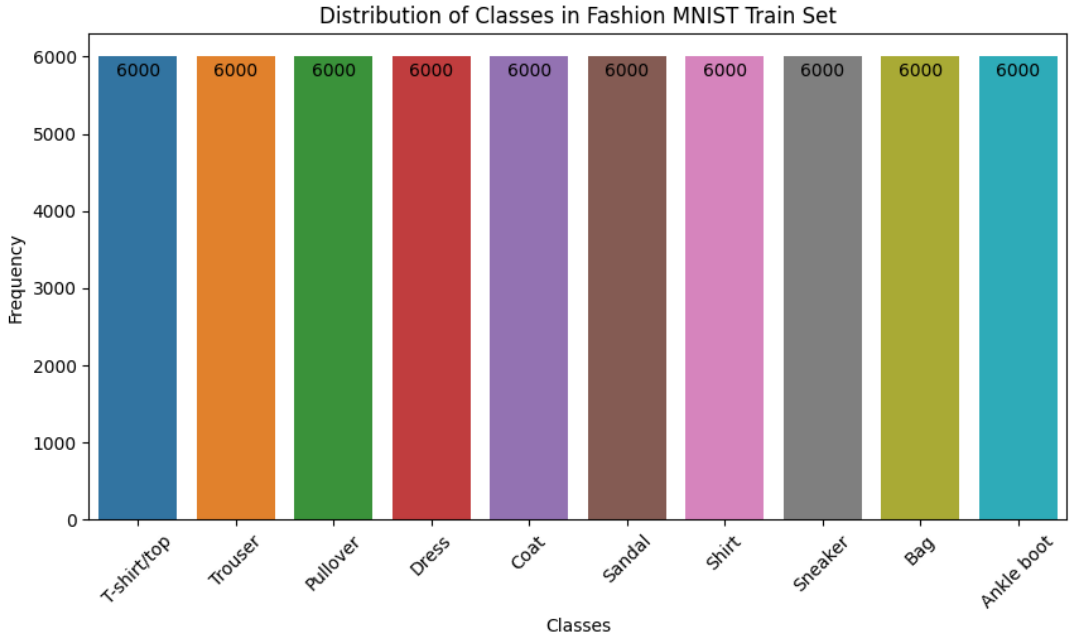


Figure 2: Distribution of Classes in the Initial Fashion MNIST Training Set

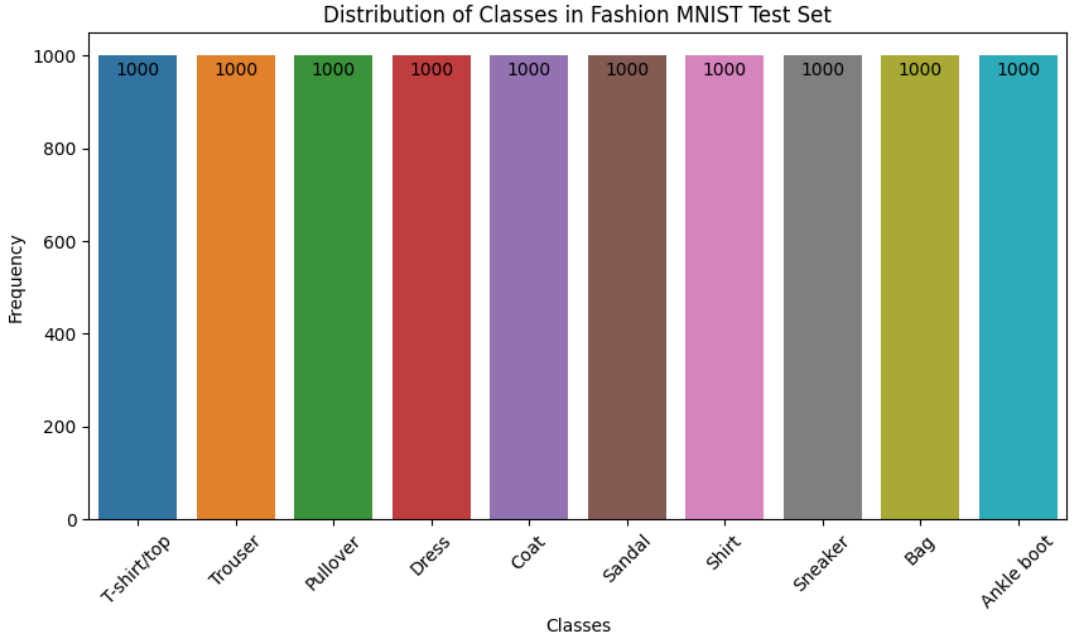


Figure 3: Distribution of Classes in the Initial Fashion MNIST Test Set

The distribution of classes in both the training and test sets is uniform, with each class represented by 6,000 images in the training set and 1,000 images in the test set, as shown in Figure 2 and Figure 3. This uniformity is crucial for maintaining a balanced dataset, which ensures that the machine learning models do not develop a bias towards a particular class due to uneven sample sizes.

3.2 Data Preparation

As part of the data preparation, we further divided the training set to create a validation set. This split is a common practice in machine learning to evaluate the performance of models on unseen data. The validation set allows for the tuning of model hyperparameters without compromising the test set's role as an unbiased evaluator.

The subsequent charts (Figure 4 and 5) illustrate the class distribution within the training set following its subdivision and the subsequent establishment of a validation set:

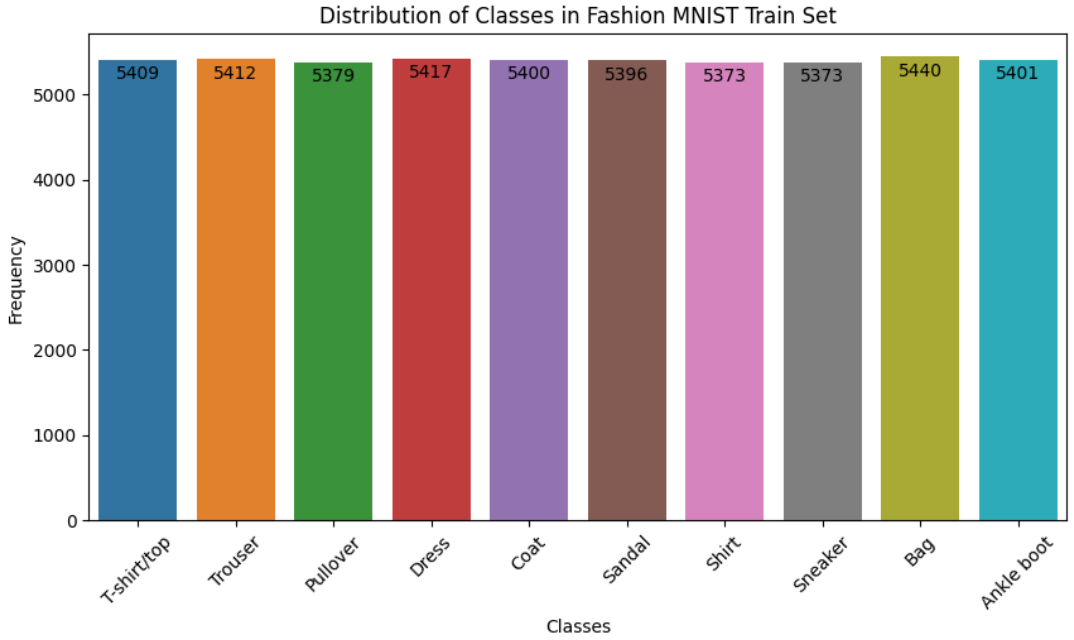


Figure 4: Distribution of Classes in the Fashion-MNIST Training Set After the Split

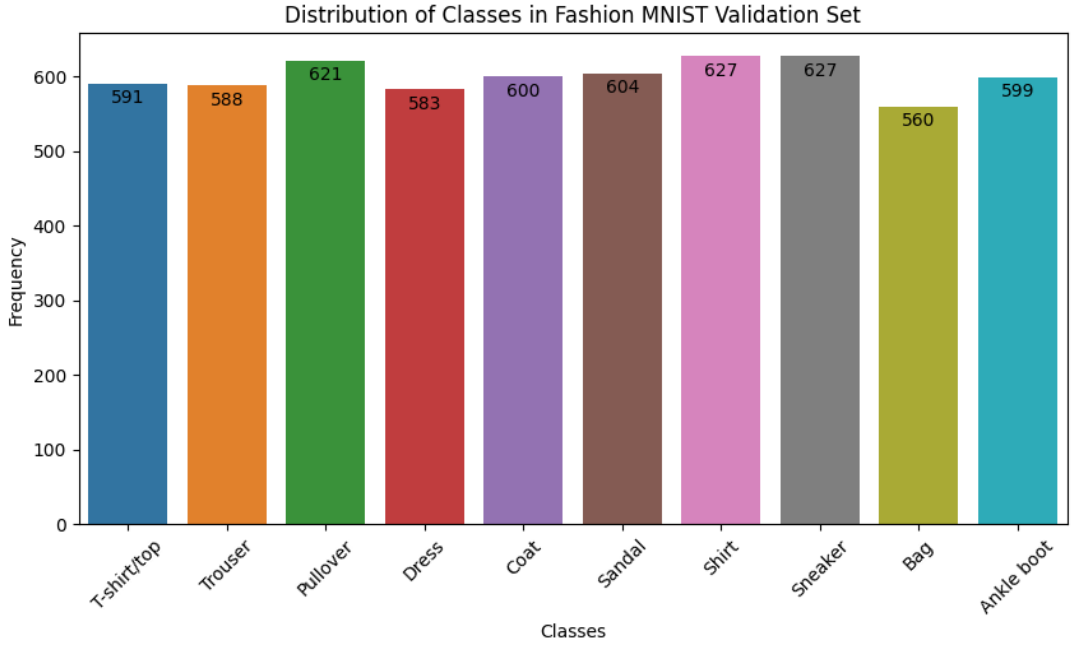


Figure 5: Distribution of Classes in the Fashion-MNIST Validation set

After the split, the new training set and validation set also exhibit a relative balanced class distribution. Each class is well-represented, which implies that the validation set is a reliable subset for tuning and validating our models. This balance is instrumental in achieving a generalizable performance across various types of fashion items.

3.3 Data Preprocessing

Prior to applying dimensionality reduction and clustering techniques, it is imperative to preprocess the data to ensure that it is in a suitable format for analysis. This preprocessing phase involves two crucial steps: flattening and normalizing the image data.

Each image in the Fashion-MNIST dataset is a 28x28 pixel grayscale image. Flattening is the process of transforming these 2D images into 1D arrays. This conversion is necessary because most machine learning algorithms, including the ones used for dimensionality reduction and clustering, expect input data in a flat, vectorized form. The flattening operation reshapes the training, test and validation arrays, converting each 28x28 image into a 1-dimensional array of 784 pixel values.

Normalization is another preprocessing step that scales the pixel intensity values to be within a range of 0 to 1. This is achieved by dividing each pixel value by 255, the maximum intensity value for an 8-bit grayscale image. Normalization is a standard practice that helps to speed up the training process and improve the performance of the model by providing a common scale for all the input features. It ensures that the pixel intensity values, which originally range from 0 to 255, do not disproportionately influence the outcome due to their larger range.

By flattening and normalizing the data, we prepare it for the subsequent stages of our analysis. These steps are vital for the successful application of the machine learning algorithms that will be discussed in the following sections of this report.

4 Implementation

The above clustering techniques were applied to both the raw, normalized data and the output from the distinct dimensionality reduction techniques. For each dimensionality reduction technique, the resulting reduced-dimensional representations were fed into the clustering models.

The evaluation of clustering performance for each model and for each set, was conducted using the metrics presented above: Calinski–Harabasz Index, Davies–Bouldin Index, Silhouette Score and Adjusted Rand Index.

The rationale behind applying clustering to both raw and reduced-dimensional data lies in understanding how dimensionality reduction impacts the ability of clustering models to discover meaningful patterns. By leveraging a variety of techniques, we aim to uncover insights into the interplay between dimensionality reduction and clustering.

4.1 Dimensionality Reduction Techniques

4.1.1 PCA

In the context of the present research, the PCA technique was applied to the training and test set of the dataset.

For the implementation of PCA, two key parameters need to be initialized: the number of components (n_components) and the solver that will be used (svd_solver).

- **n_components**: The number of principal components to keep. This determines the amount of variance retained in the reduced data. Selecting the right number of components is crucial for balancing information retention and dimensionality reduction. Too few components might lose essential information, while too many might retain noise.
- **svd_solver**: The solver used for computing the principal components. Different solvers can be more efficient depending on the size and nature of the data.

In our implementation, the n_components is set equal to 20. The validity of this choice is later explored, based on the cumulative explained variance achieved using 20 components. In addition, we used the default solver of the PCA implementation in scikit-learn, which is the randomized singular value decomposition (SVD) solver. This can be a suitable option when you need a fast approximation of the principal components for dimensionality reduction.

Having performed dimensionality reduction, a comparison is provided between the original images and the reconstructed images that emerged after applying PCA, for visualization purposes. Here is an illustrative instance of this comparison:

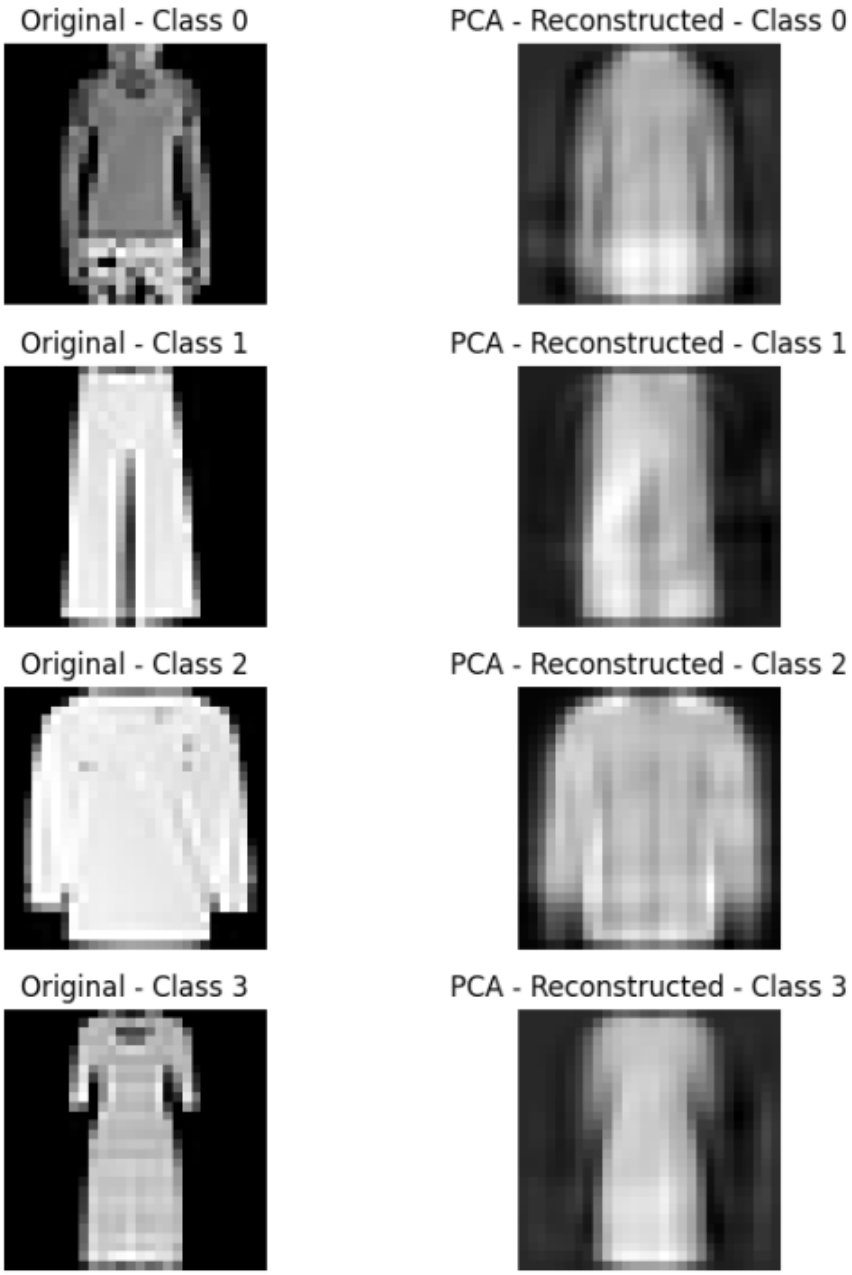


Figure 6: PCA: Original and Reconstructed Images Example

In order to support the validity of the dimensionality reduction, we plotted the individual and cumulative explained variance of principal components obtained through PCA. The chart that follows visually demonstrates the proportion of total variance captured by each component and the cumulative variance as more components are considered:

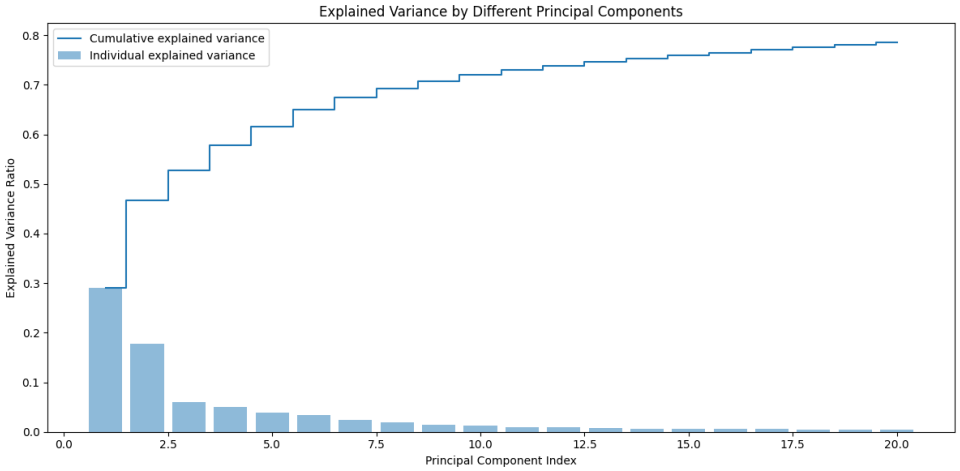


Figure 7: PCA: Explained and Comulative Variance by Principal Components

A higher cumulative explained variance indicates that the selected principal components retain a significant portion of the original data's variability. The substantial cumulative explained variance achieved through PCA, notably with 20 components capturing almost 80% of the total variability, underscores the efficacy of this dimensionality reduction technique in preserving essential information from the original data. This observation instills confidence in the method's capability to retain key features while reducing dimensionality, reinforcing its suitability for the task at hand.

4.1.2 SAE

The complexity of SAE directly affects its ability to learn representations. However, it requires careful tuning to avoid overfitting and ensure meaningful feature extraction.

The implementation of SAE requires the specification of two key parameters: the number of layers and the activation function.

- **Number of Layers**: The number of layers in an SAE is determined by the depth of the encoder-decoder structure, with each layer contributing to the hierarchical learning of features. Deeper autoencoders can capture more complex representations but are also more prone to overfitting.
- **Activation Function**: The choice of an activation function in a Stacked Autoencoder (SAE) significantly influences the learning capacity, training efficiency, and the types of patterns the model can capture. The specific choice may also depend on the nature of the data and the objectives of the autoencoder.

For the Stacked Autoencoder (SAE) architecture used in this implementation, the encoder-decoder structure comprises three densely connected layers each for both encoding and decoding phases. The input layer has 784 neurons corresponding to the flattened pixel values of the 28x28 Fashion MNIST images. The encoder progressively reduces the dimensionality with hidden layers of 128 and 64 neurons, ultimately reaching the specified encoding dimension of 50 neurons. The model was trained using a total of 50 epochs, where each epoch corresponds to one complete iteration through the entire training dataset.

The activation function utilized in these layers is the rectified linear unit (ReLU), chosen for its ability to introduce non-linearity and capture intricate patterns in the data, contributing to the model's capacity to learn complex representations.

The decoder then mirrors this structure, aiming to reconstruct the input.

A comparison is presented between the initial images and the reconstructed images resulting from the utilization of Stacked Autoencoders (SAE). Here is a representative example of this comparison:

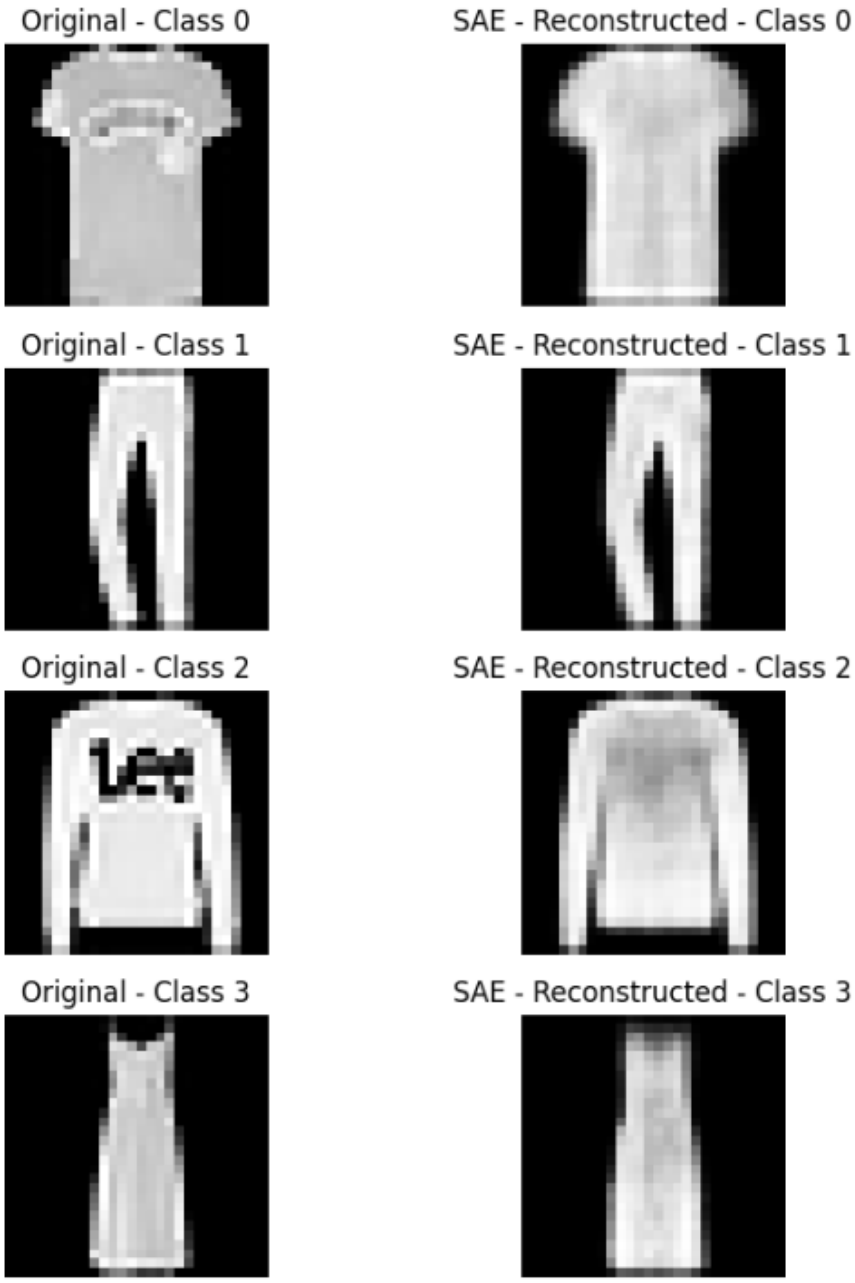


Figure 8: Original and Reconstructed Images Example for SAE

In order to support the notion that the SAE is likely to work efficiently, we plotted the training and validation loss over epochs for the model. The training loss curve represents the reconstruction error on the training dataset, while the validation loss curve reflects the model's performance on a separate validation set. Below is the relevant plot displaying the analysis:

Figure 9: Training and Validation Loss Over Epochs for SAE

The consistent decrease in both training and validation loss signifies that the SAE is effectively learning to reconstruct the input data. This alignment between the two loss curves indicates that the SAE is not overfitting to the training data and is generalizing well to unseen instances. The convergence of the loss curves suggests that the model is progressively improving its ability to capture essential features during training.

4.1.3 CNN-SAE

The implementation of CNN - SAE requires the specification of two important initialization parameters: the Filter Size and the Stride. Filter size and stride influence how effectively the CNN can extract local features and reduce dimensionality. Proper tuning is needed to capture relevant features without losing critical spatial information.

- **Filter Size**: The size of the convolutional filters; affects the field of view of the convolutional layers.
- **Stride**: The step size of the convolutional filters; impacts how the input is traversed and compressed.

The architecture designed for the CNN-SAE is as follows. The encoder consists of two convolutional layers followed by max-pooling operations, reducing spatial dimensions while increasing the number of learned features. The first convolutional layer employs 32 filters with a size of 3x3, followed by a max-pooling layer and rectified linear unit (ReLU) activation to introduce non-linearity. Subsequently, a second convolutional layer with 16 filters is applied, again followed by max-pooling and ReLU activation.

The decoder mirrors this structure, utilizing transposed convolutional layers for upsampling. The first transposed convolutional layer has 16 filters and is followed by upsampling with ReLU activation, while the second has 32 filters and further upsampling. The final layer employs a single filter with a sigmoid activation function to produce reconstructed images.

In this implementation, we used the default value for the stride parameter. In Keras, the default stride for both Conv2D and MaxPooling2D layers is set to (1, 1) unless specified otherwise. This means that the convolutional kernels move one unit at a time in both the horizontal and vertical directions during convolution, and the pooling operation is applied to non-overlapping regions.

The training process comprised a total of 50 epochs.

The Figure below shows an example of the comparison between the original and the reconstructed image resulting after applying CNN-SAE:

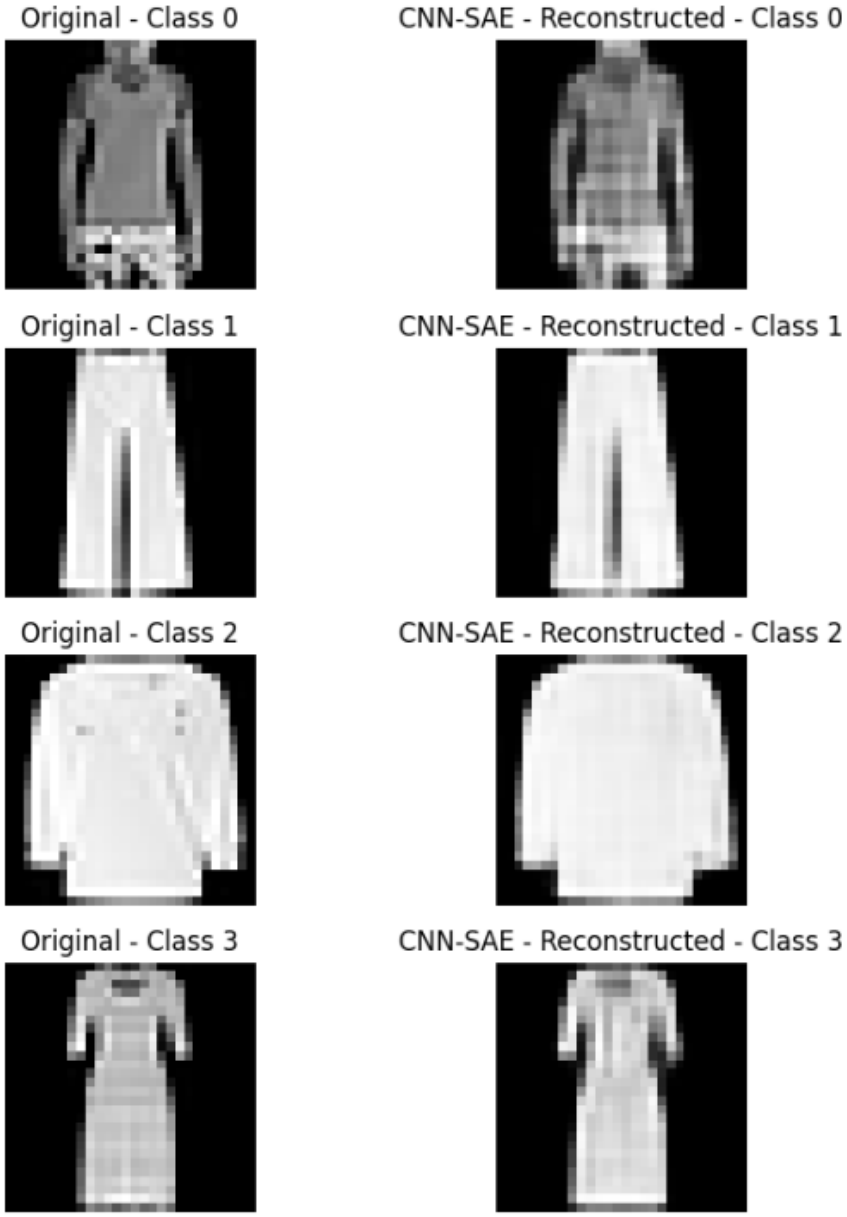


Figure 10: Original and Reconstructed Images Example for CNN-SAE

Analogous to the procedure applied previously for the standard Stacked Autoencoder (SAE), the training and validation loss curves for the Convolutional Stacked Autoencoder (CNN-SAE) were visualized. The plot is shown below:

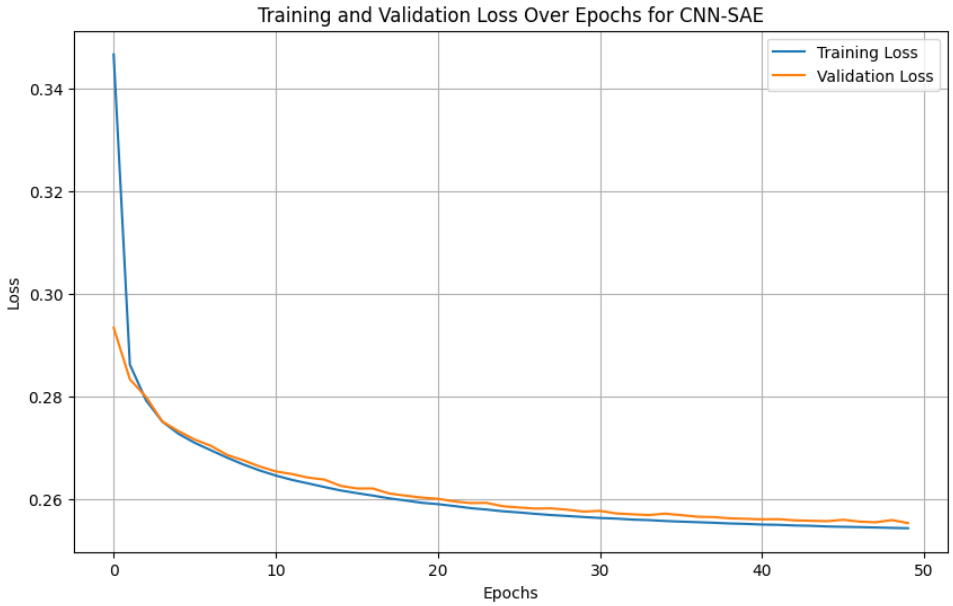


Figure 11: Training and Validation Loss Over Epochs for CNN-SAE

The diminishing trend in both training and validation loss indicates effective learning and generalization by the CNN-SAE, showcasing its capacity for dimensionality reduction.

4.1.4 LDA

The most important parameter that needs to be specified in the implementation of LDA is the number of components (n_components).

- **n_components**: Number of linear discriminants to retain. This affects how well the reduced dimensions separate the classes. Choosing the appropriate number of components is vital for maximizing class separability. It can enhance the clarity of clustering but may miss subtle between-class variations if set too low.

In our analysis, we applied LDA on the dataset, specifying the number of components to be 5. The effectiveness of this choice will be explored in greater extend later.

The reduced-dimensional representation obtained through LDA does not retain sufficient information for accurate image reconstruction. Thus, there is no such figure to be shown. Instead, the following diagram shows a visualization of the dataset after the application of LDA:

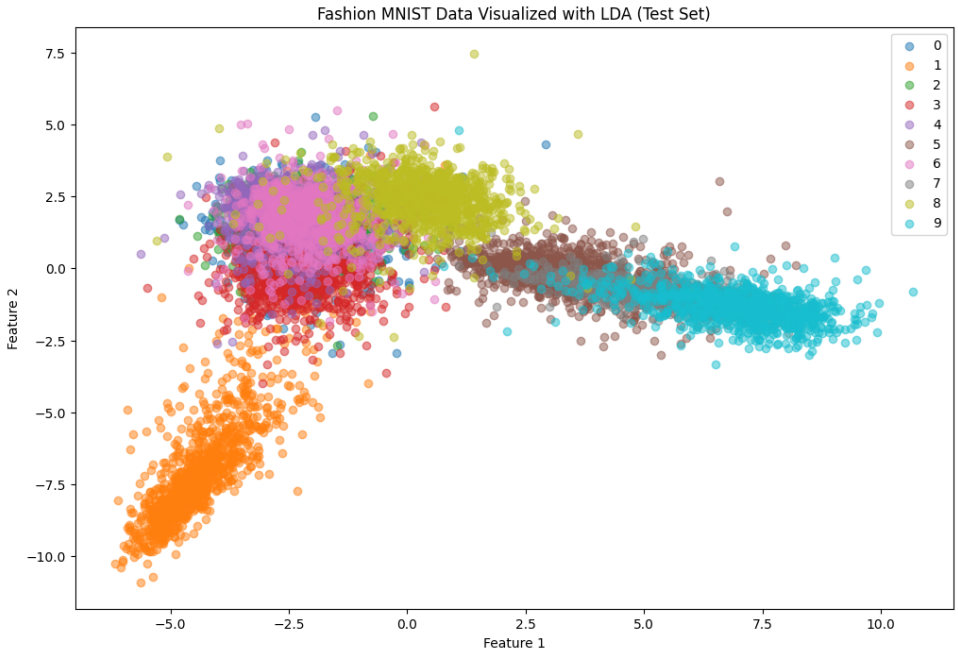


Figure 12: Visualization of the Dataset After Applying LDA

To assess the effectiveness of the dimensionality reduction using LDA, we included a bar chart depicting the individual explained variance for each component, alongside a line plot indicating the cumulative explained variance. The chart is illustrated below:

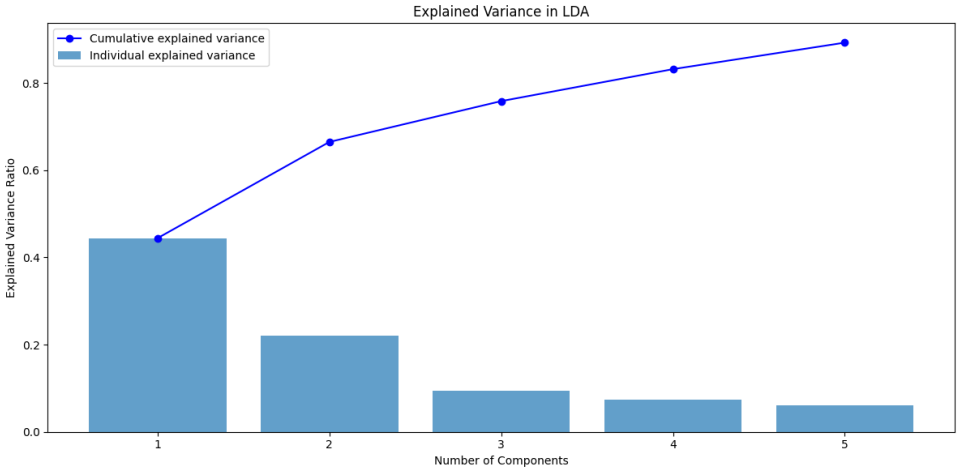


Figure 13: LDA: Explained and Comulative Variance by Different Principal Components

The considerable cumulative explained variance achieved by LDA is noteworthy, with the observation that merely 5 components encapsulate nearly 80% of the total variability. The clear progression in cumulative explained variance indicates that a significant amount of information is retained even with a reduced number of components. This suggests that the selected components effectively capture essential features for class discrimination in the lower-dimensional space. The upward trend in cumulative explained variance reinforces the efficacy of the LDA transformation, showcasing its ability to preserve discriminative information in a more compact representation.

4.1.5 UMAP

The implementation of UMAP requires the specification of multiple initialization parameters. The parameters of UMAP are crucial for defining the granularity at which clusters are formed. They dictate the balance between preserving global vs. local data structures and directly influence the clarity and separation of clusters. These parameters are the following:

- **n_components**: Specifies the number of dimensions to which the data is reduced, impacting the overall shape and spread of the clusters in the reduced space.
- **n_neighbors**: Determines the local neighborhood size for manifold approximation. It affects the balance between local and global structure in the data.
- **min_dist**: The minimum distance between points in low-dimensional space. Controls how tightly points are packed together.

For the purpose of our task, we employed UMAP on the dataset, specifying the "n_components" parameter for the transformation to be three. Choosing three components in UMAP for the Fashion MNIST dataset is motivated by the desire to create a visually rich and interpretable representation that leverages three-dimensional space to capture the complexity and diversity of clothing items in a more detailed manner.

The parameter "n_neighbors" was set to be 50. This indicates that each point in the high-dimensional space considers its 50 nearest neighbors when constructing the low-dimensional representation. A larger n_neighbors value tends to preserve more global structure in the data, capturing broader relationships between data points. This can be beneficial for revealing overarching patterns and ensuring a more comprehensive representation of the dataset.

The "min_dist" parameter was initialized as 0. Setting "min_dist" to 0 means there is no enforced minimum distance between points in the low-dimensional space. Points are allowed to be as close as needed to preserve intricate local structures without any imposed separation. A min_dist of 0 can be particularly useful when dealing with complex and densely packed clusters in the data. It allows UMAP to focus on accurately representing the local relationships between points without introducing artificial separations.

UMAP transforms high-dimensional data into a lower-dimensional space in a way that emphasizes preserving local neighborhoods, making it unsuitable for the task of reconstructing the original images. Instead of the reconstructed images, the following figure visualizes the dataset after applying UMAP. Each point in the plot represents a sample from the dataset, and the position of points in the 3D space is determined by their UMAP features.

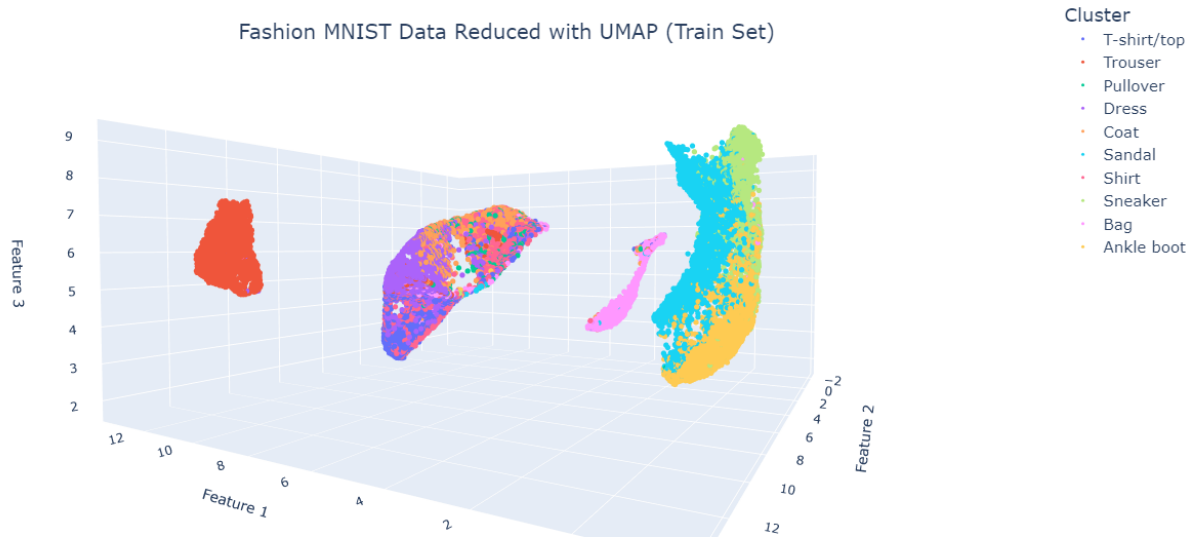


Figure 14: Visualization of the Dataset After Applying UMAP

4.2 Clustering Techniques

This section discusses the clustering techniques applied to both the raw normalized data and the output from the various dimensionality reduction techniques. We detail the parameter selection and the rationale for the chosen settings, alongside the implementation process and the subsequent analysis of the clustering outcomes.

4.2.1 MiniBatch K-Means

MiniBatch K-Means, a variant of the K-Means algorithm, was selected for its computational efficiency on large datasets. The key initialization parameters for K-Means are:

- **n_clusters**: The number of clusters to form. It determines the value of k in the clustering process.
- **batch_size**: The size of each minibatch for updating the cluster centroids. A smaller batch size can accelerate convergence but may result in less accurate centroids.
- **max_iter**: The maximum number of iterations. It controls the number of updates to the centroids during training.
- **init**: The method used for initialization of centroids.

We specified the number of clusters to match the number of classes in the dataset, aiming for a direct correspondence between clusters and fashion categories.

The MiniBatch size was set to 1024 samples to balance between computational speed and convergence stability.

The max_iter was set to 3. By performing three initializations, we mitigate the risk of the algorithm converging to a suboptimal solution due to the stochastic nature of K-Means. This approach provides a practical compromise, allowing for a more reliable estimation of cluster centroids while avoiding excessive computational overhead associated with a large number of initializations.

The init parameter for this implementation was set to default. In scikit-learn's MiniBatch K-Means implementation, the default value for init_params is "k-means++". This means that the algorithm initializes centroids using the "k-means++" initialization method, which intelligently places the initial centroids to enhance the convergence speed and overall clustering quality. In practical terms, "k-means++" initialization often leads to faster convergence and more accurate clustering results compared to random initialization. It helps mitigate the sensitivity of K-Means to the initial placement of centroids, making the algorithm less prone to converging to suboptimal solutions.

The following table summarizes the performance metrics obtained from applying MiniBatch K-Means clustering to both the raw data and the data transformed by each dimensionality reduction technique.

Technique	Clusters	Calinski-Harabasz	Davies-Bouldin	Silhouette Score	ARI
RAW	10	1164.287	2.272	0.127	0.351
PCA	10	2248.792	1.356	0.243	0.372
SAE	10	1776.381	1.698	0.162	0.335
CNN-SAE	10	1354.821	1.998	0.140	0.351
LDA	10	9126.905	1.012	0.395	0.516
UMAP	10	42203.110	0.683	0.532	0.465

Table 1: Clustering Performance Metrics for MiniBatch K-Means Clustering

The table clearly illustrates the impact of dimensionality reduction on the clustering performance. Notably, the application of LDA and UMAP resulted in a significant increase in the Calinski-Harabasz Index and Silhouette Score, suggesting that these methods enable a clearer separation between clusters compared to the original high-dimensional space. The Adjusted Rand Index also improved, particularly with LDA, indicating a better alignment with the true class labels.

The following diagram is an example of the results of the application of Minibatch k-means, after UMAP is used for dimensionality reduction. UMAP is applied using 3 components, so the results are plotted in 3D space. The 10 clusters created can be observed in the plot.

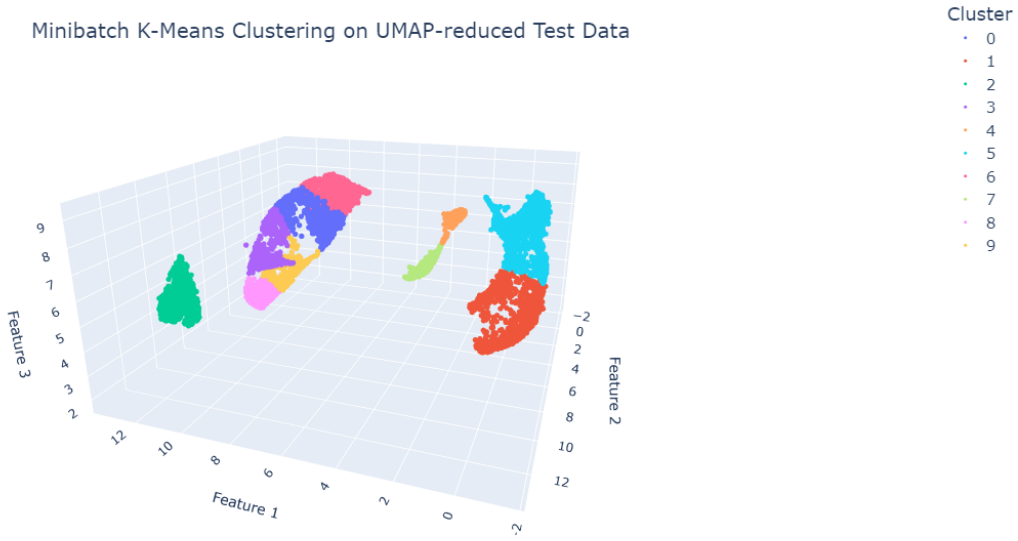


Figure 15: Minibatch K-Means Clustering Results After UMAP Dimensionality Reduction

4.2.2 DBSCAN

DBSCAN was chosen for its ability to find arbitrarily shaped clusters and identify outliers.

The DBSCAN algorithm has the following important parameters:

- **eps**: The maximum distance between two samples for one to be considered as part of the same neighborhood. It influences the size of the local neighborhood.
- **min_samples**: The number of samples in a neighborhood for a point to be considered a core point. It affects the sensitivity to noise and the minimum cluster size.

The parameters `eps` and `min_samples` have different values for each dimensionality reduction technique applied each time, depending on the quality of the results and the metric values. This approach allowed the algorithm to adapt to the density distribution of the data. The initialization values for these parameters per dimensionality reduction technique are summarized on the following table:

Technique	eps	min_samples
RAW	9	1000
PCA	5	100
SAE	3	100
CNN-SAE	5	100
LDA	1	13
UMAP	1	10

Table 2: Initialization Parameters for DBSCAN Clustering

It is noteworthy that in the case of DBSCAN, the predetermined number of clusters is not established in advance. Consequently, the clusters formed vary for each dimensionality reduction technique applied.

The following table shows the performance of DBSCAN for each Dimensionality reduction technique:

Technique	Clusters	Calinski-Harabasz	Davies-Bouldin	Silhouette Score	ARI
RAW	1	41.881	2.469	0.199	0.000
PCA	1	12.554	3.393	0.132	0.000
SAE	4	619.415	1.532	-0.061	0.038
CNN-SAE	1	422.199	4.426	0.081	0.039
LDA	4	4707.130	1.684	0.392	0.288
UMAP	4	20776.879	0.497	0.614	0.292

Table 3: Clustering Performance Metrics for DBSCAN Clustering

Notably, both LDA and UMAP demonstrate superior performance compared to other techniques in terms of the Calinski-Harabasz Index, Silhouette Score, and Adjusted Rand Index (ARI). With respect to the Davies-Bouldin Index, UMAP significantly enhanced the performance of DBSCAN.

In general, LDA and UMAP, with four clusters each, outperform the other techniques, showcasing their ability to capture meaningful patterns in the data.

The plot below is an example of the clustering performed by DBSCAN, after using UMAP for the dimensionality reduction. The clusters found were 4 and can be seen in the plot.

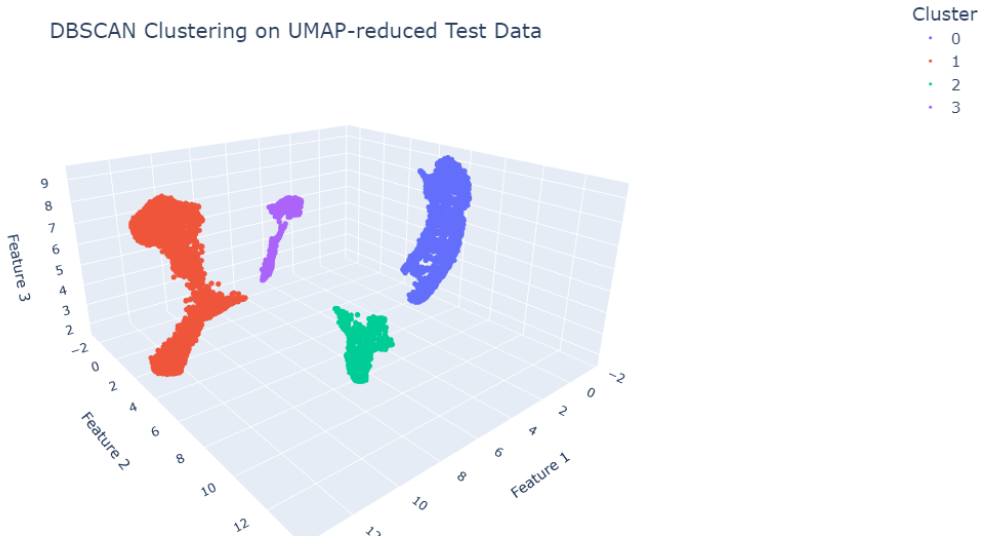


Figure 16: DBSCAN Clustering Results After UMAP Dimensionality Reduction

4.2.3 Gaussian Mixture Models

Gaussian Mixture Models (GMM) was employed to accommodate the possibility of overlapping clusters and non-spherical distributions in the data. GMM is a probabilistic clustering model with parameters related to the mixture components:

- **n_components**: The number of mixture components (clusters) in the GMM.
- **covariance_type**: The type of covariance parameters to use. The default option for this paeameter is "full".
- **max_iter**: The maximum number of iterations for the expectation-maximization algorithm.

The number of components was again set to the number of classes, and the "full" covariance type was

used to allow each cluster to define its own elliptical shape in the feature space. This is suitable for modeling clusters with arbitrary shapes.

The default value for "max_iter" is 100 and was used in this implementation. Setting "max_iter" equal to 100 for Gaussian Mixture Model (GMM) means that the Expectation-Maximization (EM) algorithm, used for fitting the GMM, will iterate at most 100 times to optimize the model parameters.

The table below shows the results of GMM on the metrics, for both the raw data and the reduced data, with each dimensionality reduction technique.

Technique	**Clusters**	**Calinski-Harabasz**	**Davies-Bouldin**	**Silhouette Score**	**ARI**
RAW	10	1032.226	2.117	0.113	0.382
PCA	10	1343.837	2.288	0.151	0.370
SAE	10	1619.272	1.788	0.171	0.369
CNN-SAE	10	1079.055	2.212	0.108	0.381
LDA	10	6514.250	1.568	0.358	0.470
UMAP	10	36738.522	0.707	0.489	0.418

Table 4: Clustering Performance Metrics for Gaussian Mixture Model Clustering

The analysis of performance metrics provides valuable insights into the efficacy of various dimensionality reduction techniques combined with the GMM clustering algorithm. UMAP consistently emerges as the top-performing technique, showcasing superior results across Calinski-Harabasz, Davies-Bouldin, and Silhouette metrics. Its ability to capture complex local structures in high-dimensional data is evident in the significantly higher scores.

LDA also demonstrates strong performance, particularly excelling in the Adjusted Rand Index. This suggests that LDA provides a compact representation of the data that aligns well with the ground truth labels.

The following diagram serves as an example of GMM clustering results on the reduced dataset by the UMAP technique. It can be observed that 10 clusters were created from the GMM model, as indicated from the initialization parameters.

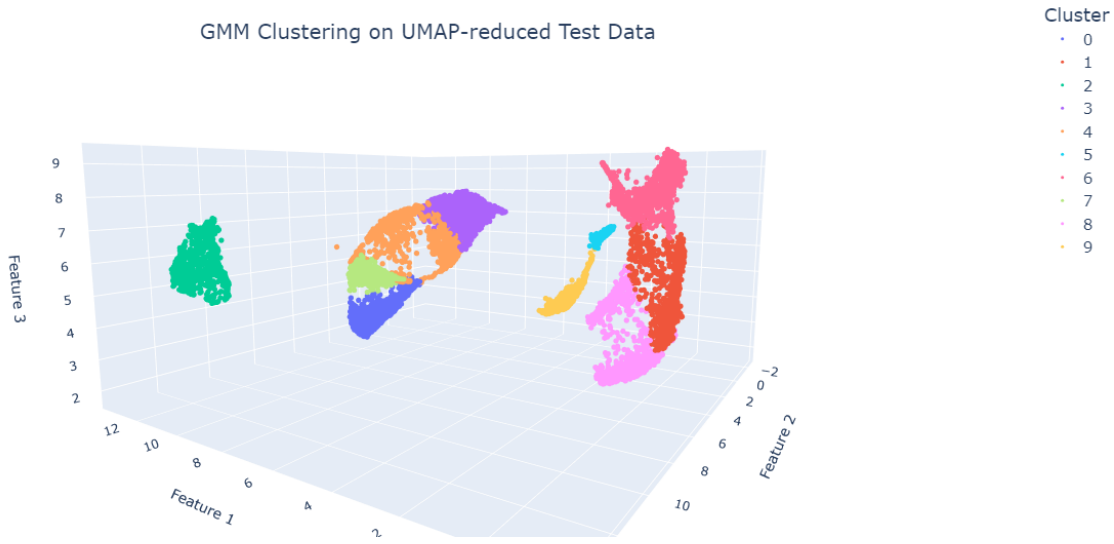


Figure 17: Visualization of Gaussian Mixture Model Clusters

4.2.4 Hierarchical Clustering

Hierarchical Clustering was applied to explore the dataset's inherent hierarchical structure. Hierarchical Clustering involves parameters related to the linkage method and the tree structure:

- **n_clusters**: The number of clusters to be formed. This parameter helps determine the granularity of the clustering solution. Choosing an appropriate number of clusters is a crucial step, as it affects the interpretability and utility of the clustering results.
- **linkage**: The linkage criterion, determining how to measure the distance between clusters. Different linkage methods can lead to different cluster structures.
- **metric**: The metric parameter defines the distance metric used for calculating the linkage between clusters. The choice of distance metric affects how the algorithm measures the dissimilarity between data points or clusters. It can influence the shape and size of the resulting clusters.

The number of clusters was predefined to be equal to 10, as the dataset includes 10 classes.

In our code, "ward" is chosen as linkage criterion, which corresponds to Ward's linkage method. Ward's method minimizes the variance within each cluster and is known for producing compact, spherical clusters. Ward's linkage is often a good choice as it tends to produce compact clusters, but it may not be optimal for all datasets.

The metric was specified as "euclidean", which means the Euclidean distance is used to measure the dissimilarity between clusters.

Technique	Clusters	Calinski-Harabasz	Davies-Bouldin	Silhouette Score	ARI
RAW	10	1116.133	1.933	0.117	0.348
PCA	10	2040.495	1.389	0.223	0.392
SAE	10	1713.311	1.605	0.193	0.427
CNN-SAE	10	1272.122	1.902	0.137	0.348
LDA	10	8612.240	1.039	0.339	0.474
UMAP	10	42328.090	0.633	0.541	0.471

Table 5: Clustering Performance Metrics for Hierarchical Clustering

The examination of performance metrics reveals distinctive patterns in the effectiveness of various dimensionality reduction techniques combined with Hierarchical Clustering. UMAP consistently outperforms other techniques across all metrics, showcasing a remarkable capability to preserve intricate local structures and providing well-defined clusters. The significantly higher scores in Calinski-Harabasz, Davies-Bouldin, Silhouette, and Adjusted Rand Index emphasize its suitability for revealing hierarchical relationships within the data.

Additionally, LDA exhibits strong performance, especially excelling in the Adjusted Rand Index, indicating its proficiency in preserving discriminative features relevant to the ground truth labels. The nuanced results highlight the importance of tailored technique selection, with UMAP standing out as a robust choice for revealing hierarchical structures within the dataset.

The results of Hierarchical CLustering are plotted in 3D space, after UMAP is used for dimensionality reduction. The diagram can be seen below:

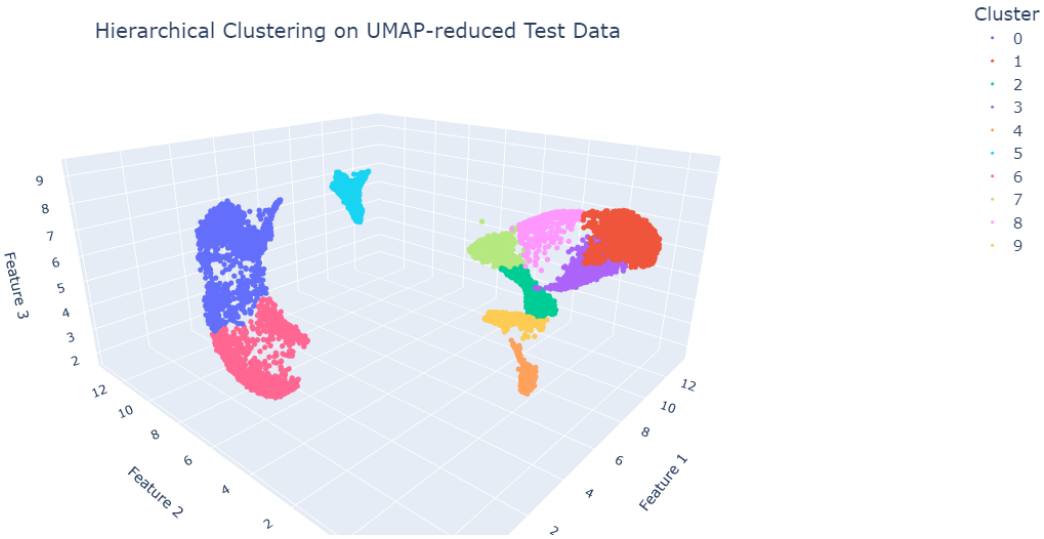


Figure 18: Visualization of Hierarchical Clustering Results After UMAP

In addition, the resulting dendrogram from the application of Hierarchical Clustering, after UMAP, is presented below:

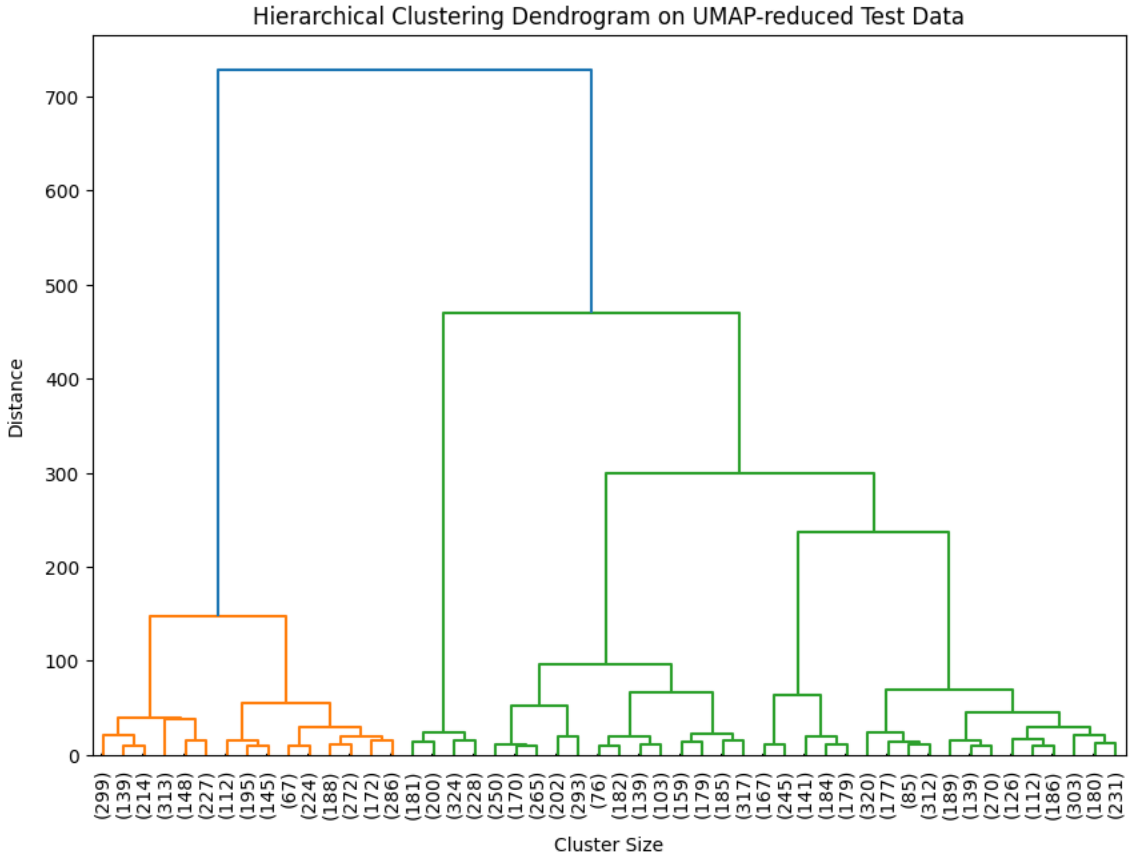


Figure 19: Resulting Dendrogram from Hierarchical Clustering After UMAP

4.2.5 Spectral Clustering

Spectral Clustering relies on parameters related to the graph and affinity matrix:

- **n_clusters**: The number of clusters to form.
- **affinity**: The affinity matrix used to capture the relationship between data points.
- **n_neighbors**: The number of neighbors to consider when constructing the affinity matrix, applicable when using methods like "nearest_neighbors". A higher number of neighbors may increase the computational cost, especially for large datasets, as the algorithm needs to compute pairwise distances between data points.

The "n_clusters" was again set to 10.

The affinity chosen is the nearest neighbors' affinity matrix, which is suitable when there is a clear local structure in the data.

The number of neighbors was set to its default value. In the Spectral Clustering implementation from scikit-learn, the default value for the "n_neighbors" is typically set to 10. This means that each data point is connected to its 10 nearest neighbors in the input space.

The results of the application of Spectral Clustering for each dimensionality technique are presented above:

Technique	Clusters	Calinski-Harabasz	Davies-Bouldin	Silhouette Score	ARI
RAW	10	1180.747	1.890	0.149	0.399
PCA	10	2134.304	1.348	0.231	0.374
SAE	10	1736.543	1.606	0.200	0.401
CNN-SAE	10	1324.107	1.777	0.163	0.392
LDA	10	8433.658	0.970	0.336	0.436
UMAP	10	10147.766	1.001	0.401	0.404

Table 6: Clustering Performance Metrics for Spectral Clustering

It can be observed that the UMAP technique exhibited the best overall performance, as indicated by its highest Calinski-Harabasz score of 10147.766, signifying dense and well-separated clusters. Additionally, UMAP achieved the highest Silhouette Score (0.401), suggesting better-defined cluster structures.

The following diagram is an example of Spectral Clustering, after the application of UMAP. The results are again presented in 3-dimensional space.

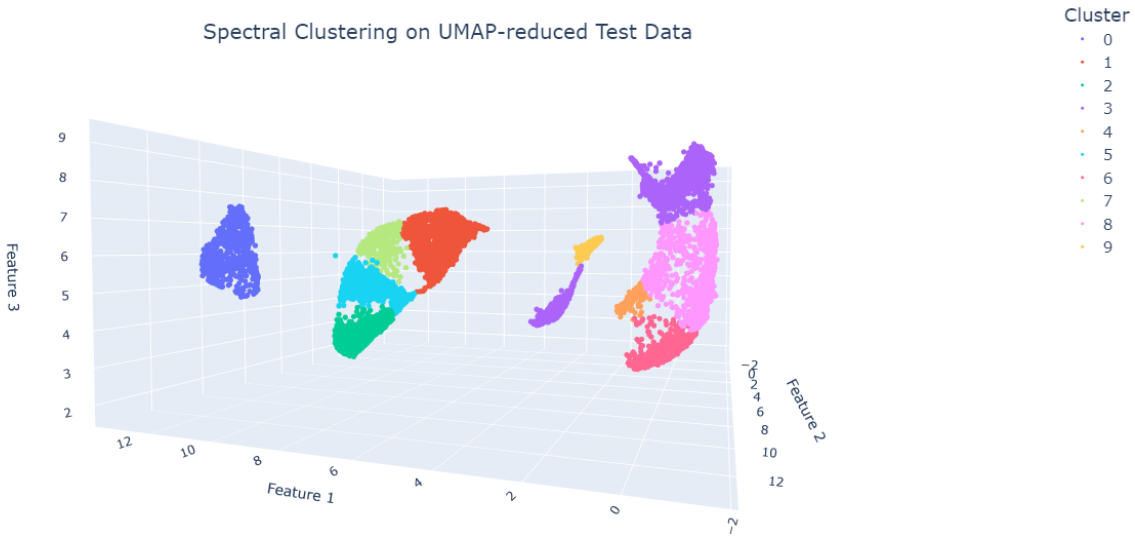
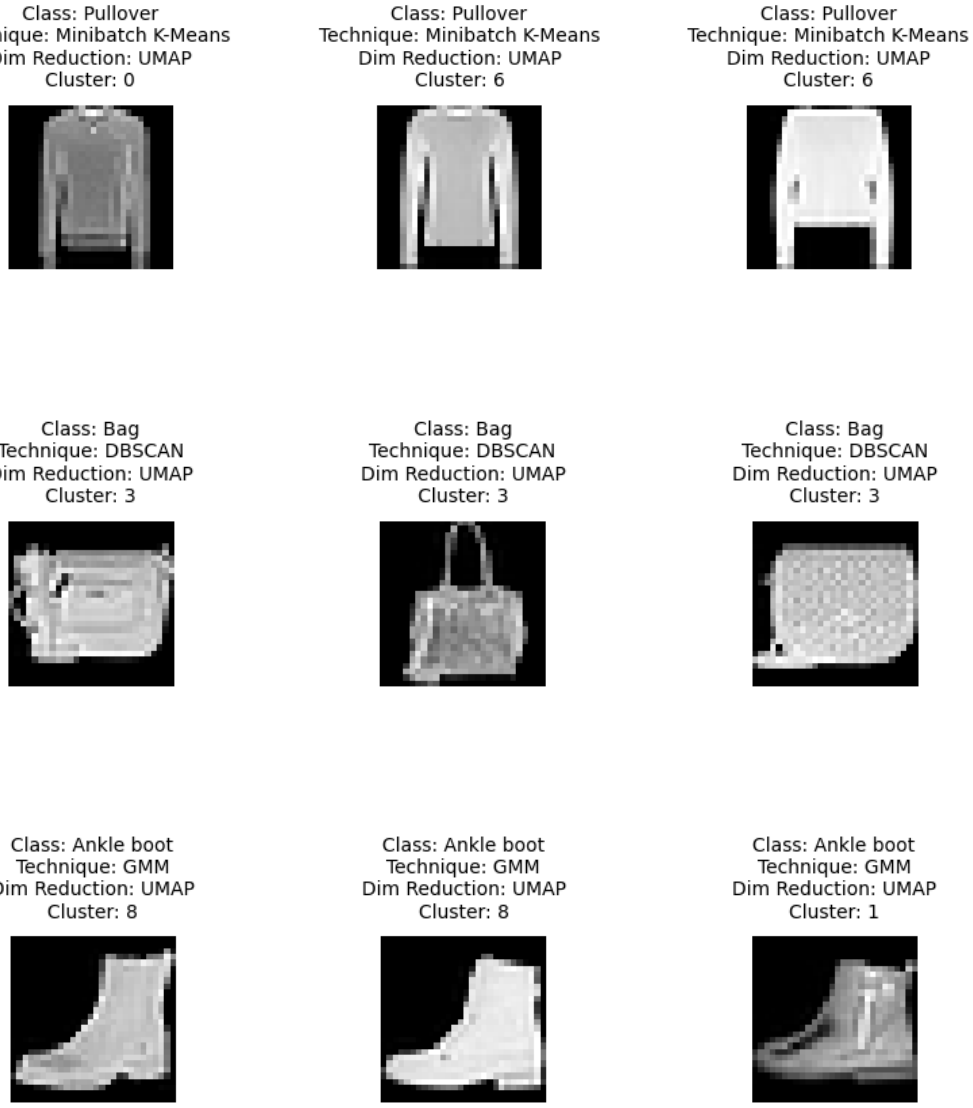


Figure 20: Visualization of Spectral Clustering Results After UMAP

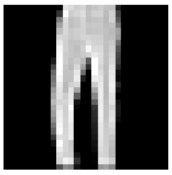
4.3 Sample Clustering Results

For the purpose of visual representation, we have undertaken an examination of illustrative clustering outcomes. In light of its superior performance in dimensionality reduction, we have opted to explore the clustering results subsequent to the application of UMAP. To facilitate a comprehensive analysis, we have chosen to explore the results of all five clustering algorithms. For each algorithm, three representative examples from each of the ten classes are presented. These examples are accompanied by the corresponding original (ground truth) label and the cluster assignment of each image.

The result of this visualization is 60 images, along with the original labels and assigned clusters. Some of them (one for each clustering algorithm) are presented below:

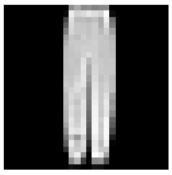


Class: Trouser
Technique: Hierarchical Clustering
Dim Reduction: UMAP
Cluster: 5

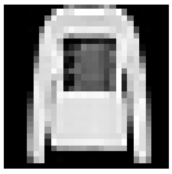


Class: Trouser
Technique: Hierarchical Clustering
Dim Reduction: UMAP
Cluster: 5

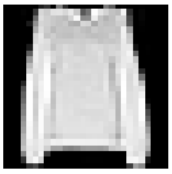
Class: Trouser
Technique: Hierarchical Clustering
Dim Reduction: UMAP
Cluster: 5



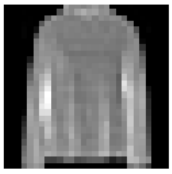
Class: Pullover
Technique: Spectral Clustering
Dim Reduction: UMAP
Cluster: 1



Class: Pullover
Technique: Spectral Clustering
Dim Reduction: UMAP
Cluster: 1



Class: Pullover
Technique: Spectral Clustering
Dim Reduction: UMAP
Cluster: 1



5 Comparative Analysis

5.1 Effect of Dimensionality Reduction on Clustering Performance

In the realm of unsupervised machine learning, the impact of dimensionality reduction techniques on clustering analysis is paramount. This reduction not only enhances computational efficiency but also facilitates a clearer understanding of the underlying relationships among data points.

In the following section, we conduct an in-depth analysis to ascertain the extent to which implementing dimensionality reduction techniques beforehand has impacted the performance of our clustering algorithms.

Within the scope of this investigation, a set of four illustrative bar plots has been developed, each dedicated to a distinct performance metric. Each diagram portrays the metric's average values from all five clustering algorithms across the five dimensionality reduction techniques as well as the unprocessed raw data. The plots for Calinski-Harabasz Index, Davies - Bouldin Index, Sillouette Score and Adjusted Rand Index are shown below.

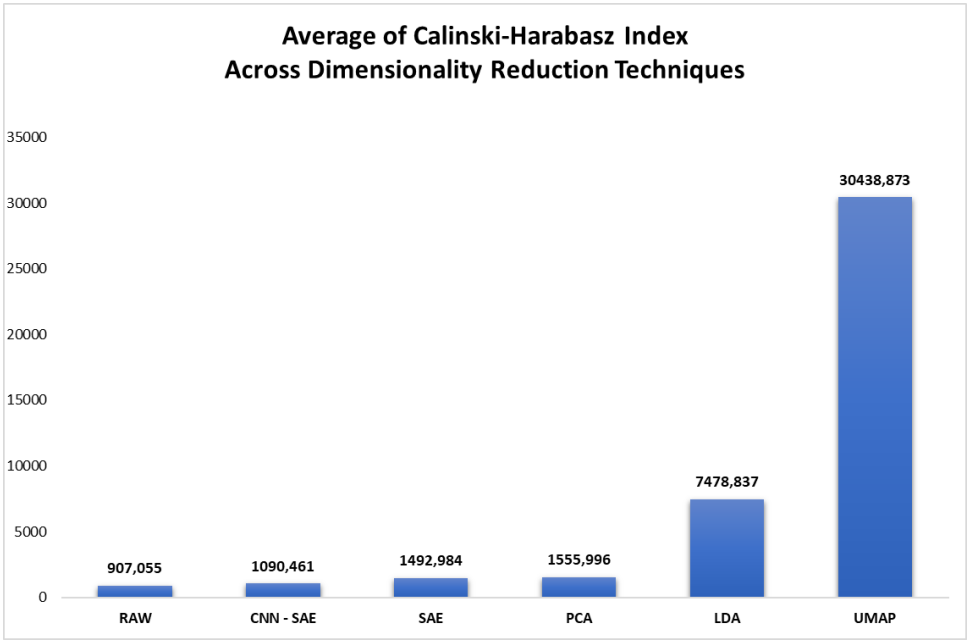


Figure 21: Visualization of Calinski-Harabasz Index Average Values

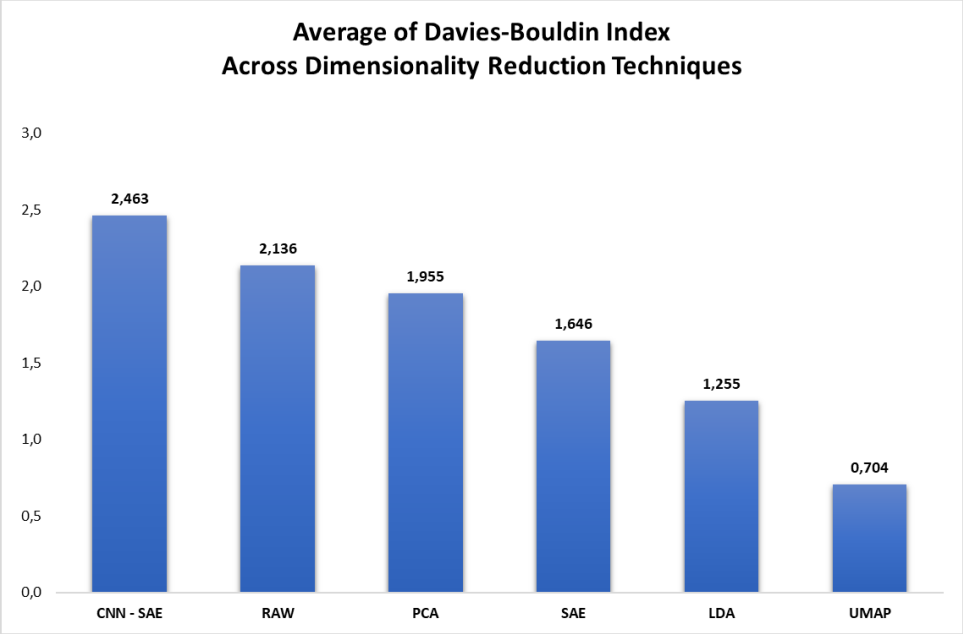


Figure 22: Visualization of Davies-Bouldin Index Average Values

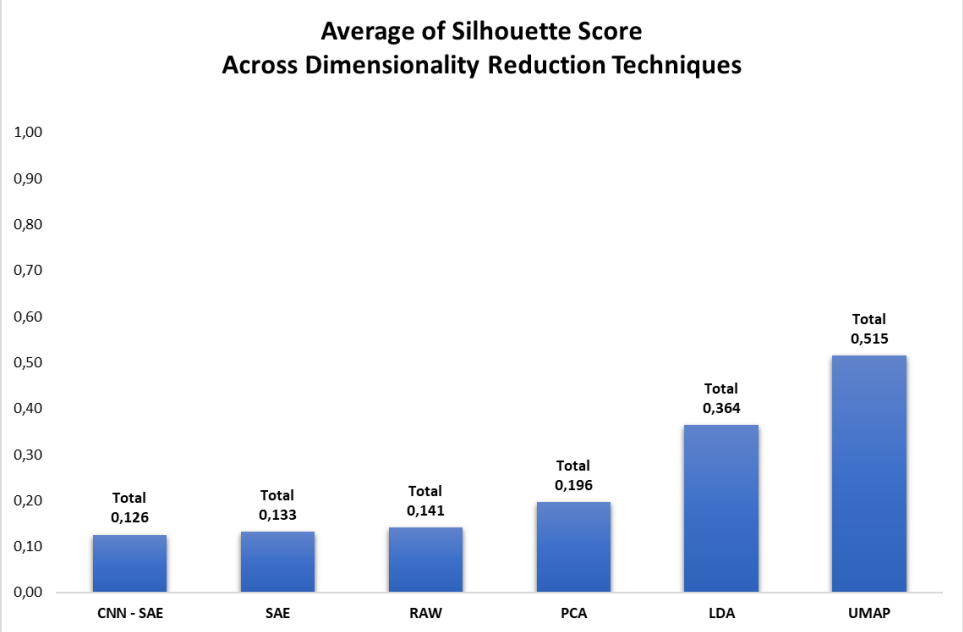


Figure 23: Visualization of Sillouette Score Average Values

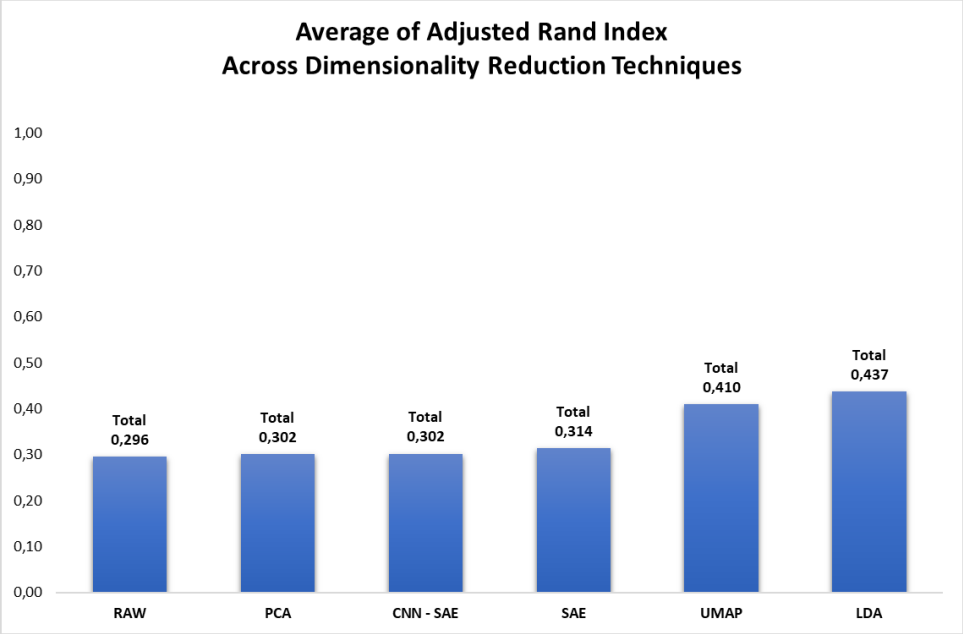


Figure 24: Visualization of Adjusted Rand Index Average Values

Based on the charts provided, it can be observed that there is a general trend where the performance of clustering, as measured by the average metric values, improves with the application of dimensionality reduction techniques compared to using raw data.

The unprocessed data score considerably low in all four metrics. The overall trend indicates that dimensionality reduction is beneficial for clustering tasks. It suggests that reducing noise and focusing on the most informative features in the data can lead to more distinct and well-separated clusters, while, without any dimensionality reduction, the clustering algorithms struggle to find a clear structure within the data.

SAE and CNN-SAE demonstrate a noticeable pattern of performance that is relatively moderate in the context of clustering analysis. Their impact on clustering enhancement, when compared to the raw data, demonstrates variability, with occasional improvements. It's worth noting that these methods do not consistently deliver significant improvements in clustering quality. While they may show positive impacts in specific instances, they do not consistently match the performance achieved by more effective alternatives like PCA, LDA and UMAP.

PCA demonstrates a moderate yet consistent enhancement in clustering performance compared to raw data across all metrics, highlighting its reliability as a dimensionality reduction technique.

UMAP has the best average values for Calinski-Harabasz Index, Davies-Bouldin Index and Silhouette Score by a substantial margin, as it scores the lowest on Davies-Bouldin and highest on the rest. In addition to that, UMAP secures the second-highest ranking in the Adjusted Rand Index, narrowly trailing the performance of the LDA method, albeit with a marginal discrepancy between them.

Subsequent to UMAP, LDA consistently attains the second position across all metrics and even surpasses UMAP in the Adjusted Rand Index. The Adjusted Rand Index stands out as the singular metric where LDA demonstrates superior performance over UMAP. This distinct outcome can be attributed to the intrinsic qualities of LDA, which are particularly aligned with the evaluative criteria of ARI. The underlying reasons for this are explained as follows:

- LDA is a supervised learning technique, which means it takes into account the class labels during the dimensionality reduction process. It aims to maximize the distance between classes while minimizing the variance within each class. This characteristic of LDA is particularly aligned with the Adjusted Rand Index, which measures the similarity between the true class labels and the

clustering outcome. By using class labels to guide the dimensionality reduction, LDA creates a feature space where the classes are more distinctly separable, potentially leading to better clustering performance as judged by ARI.

- LDA focuses on finding a projection that maximizes the separability between different classes. This is achieved by creating a new axis (or axes) that maximize the ratio of between-class variance to within-class variance. As a result, the resulting lower-dimensional space is more conducive to forming clusters that align well with the actual class labels, which is precisely what ARI measures.

- While UMAP is excellent at preserving both local and global structures within the data and is generally better for capturing the intrinsic geometric structure of the data, it does not specifically aim to separate different classes as LDA does. UMAP's primary goal is to maintain the manifold structure of the data, which might lead to clusters that are coherent in terms of data geometry but not necessarily in alignment with the predefined class labels.

To enhance our understanding of the data, we provide additional boxplots for each performance metric. These boxplots offer a visual exploration of the performance variability of dimensionality reduction techniques when paired with clustering algorithms. The boxplots displayed below reveal not only the average performance but also the distribution range and outlier occurrences, providing deeper insights into the robustness and reliability of each technique.

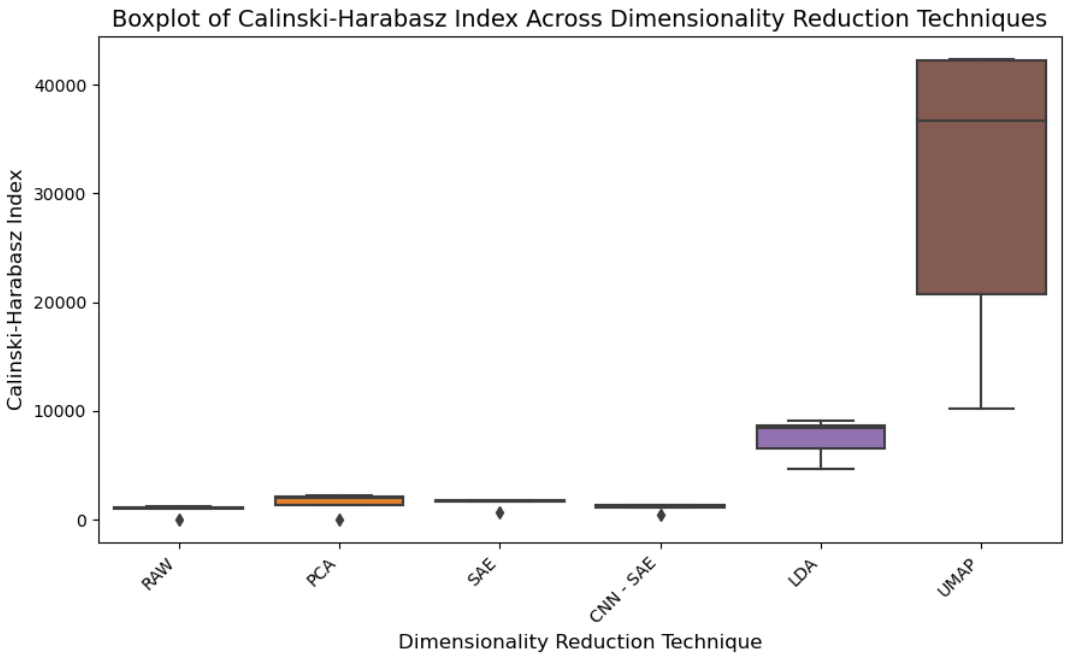


Figure 25: Boxplot of Calinski-Harabasz Index Values

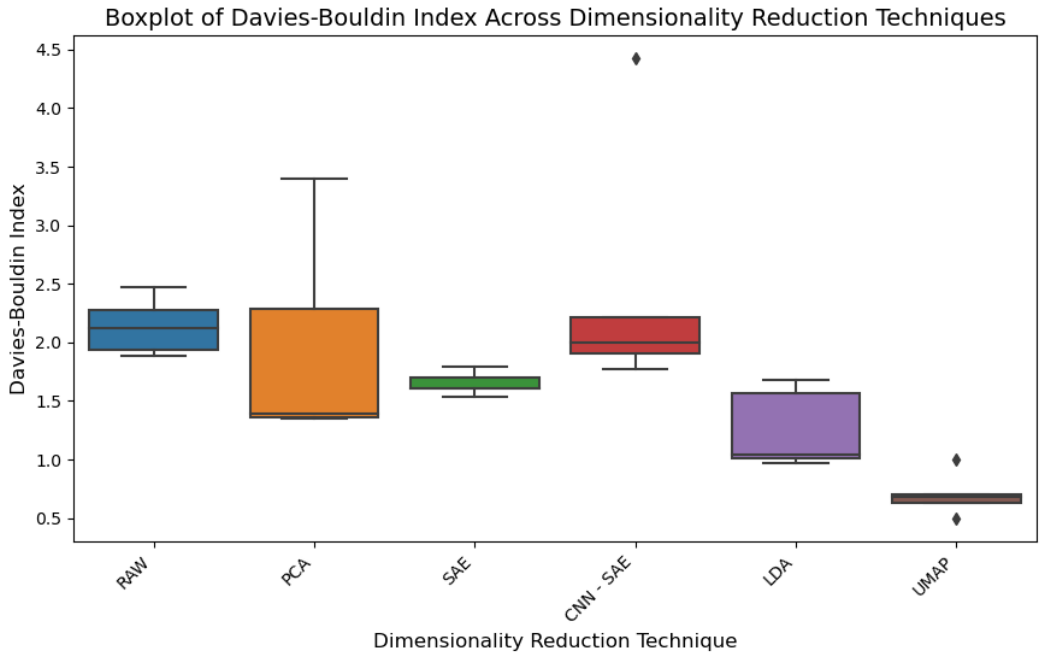


Figure 26: Boxplot of Davies-Bouldin Index Values

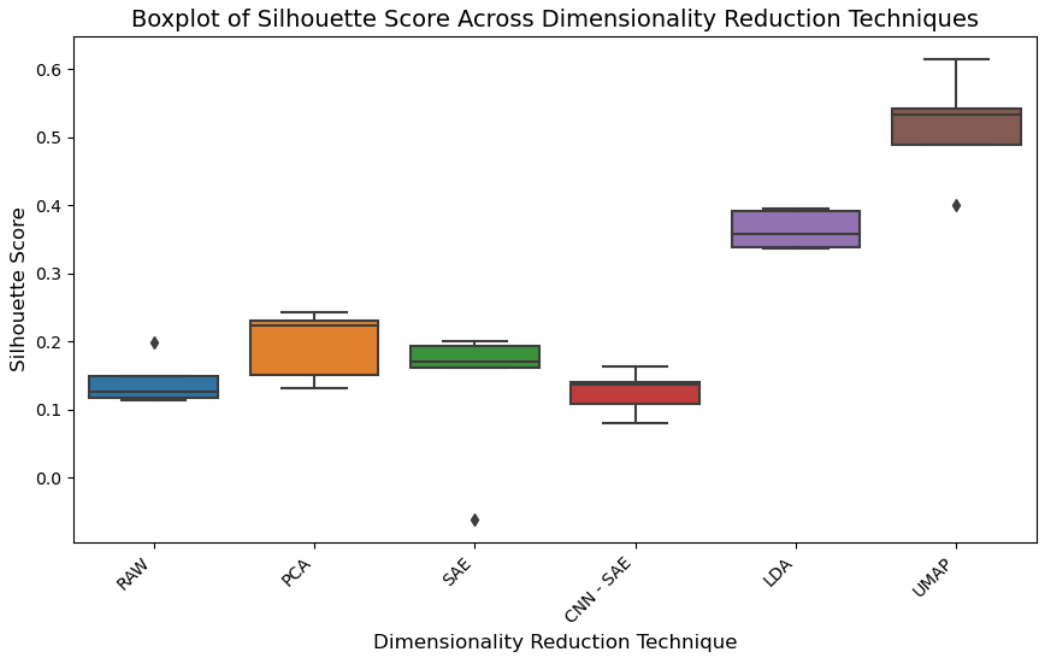


Figure 27: Boxplot of Sillouette Score Values

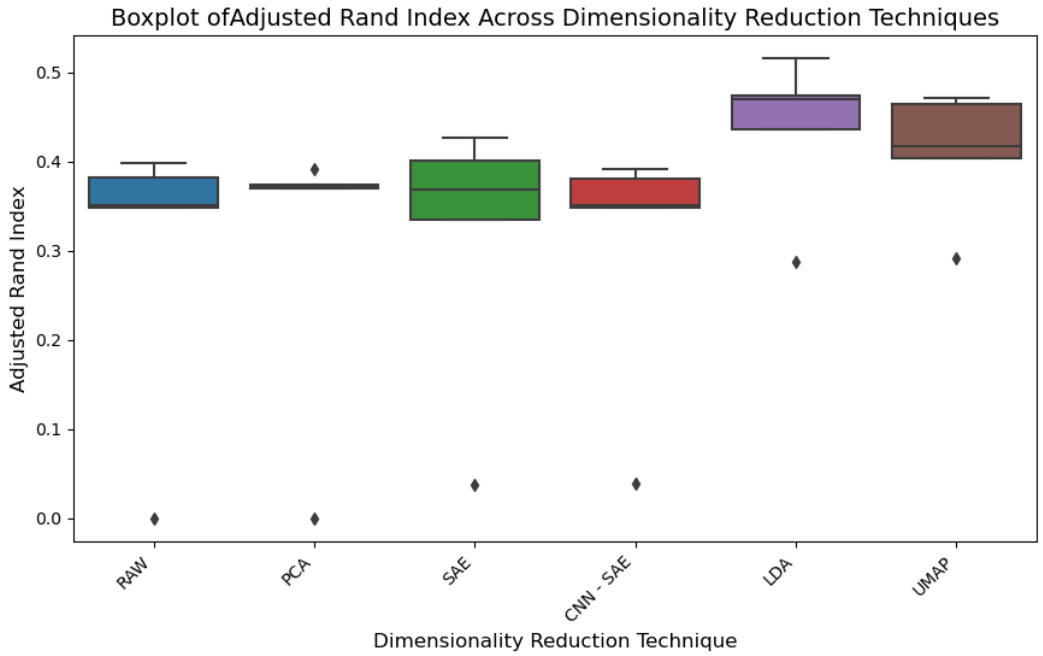


Figure 28: Boxplot of Adjusted Rand Index Values

The boxplots provided show the distribution of all four performance metrics for different dimensionality reduction techniques applied before clustering. A synthesis of insights is presented for each metric:

Calinski-Harabasz Index: UMAP outperforms other techniques with a notably high median value, indicative of well-defined and separated clusters. Although the expansive interquartile range (IQR) points to performance variability, UMAP reliably exceeds the results of other techniques.

Davies-Bouldin Index: Contrary to the Calinski-Harabasz Index, lower values of the Davies-Bouldin Index are preferred, and UMAP again shows superior performance with the lowest median value, implying compact and well-separated clusters. Although the boxplot reveals two outliers in UMAP's performance, it consistently outshines the other techniques in comparison.

Silhouette Score: UMAP demonstrates the highest median Silhouette Score, which is consistent with its performance in the Calinski-Harabasz Index, reinforcing the notion of its robust clustering. Other methods, such as LDA, also exhibit respectable medians. The remaining techniques consistently exhibit significantly lower values for all clustering methods.

Adjusted Rand Index: UMAP and LDA both show strong performance with high median values. LDA's median value is slightly higher than UMAP's, indicating a slight edge in achieving consistency between clusterings by these techniques. The tight clustering of values for RAW, PCA, SAE, and CNN-SAE indicates lower performance. The detection of outliers at the lower end of the value spectrum for all dimensionality reduction techniques indicates the existence of a clustering algorithm that persistently yields sub-optimal performance.

Summary

Overall, the general pattern suggests that employing dimensionality reduction provides advantages for clustering tasks. UMAP and LDA emerge as the most effective techniques.

UMAP outperforms its counterparts significantly, indicating that this technique is particularly effective at preserving the global data structure and enhancing clustering performance for this dataset. Unlike linear methods like PCA and LDA, UMAP adeptly captures non-linear relationships within the data, crucial for the complex range of fashion items in the dataset. Its robustness to varied data distributions and noise enhances its versatility. Most importantly, UMAP effectively balances the preservation of both local and global data structures, ensuring a comprehensive understanding of individual item characteristics and broader category relationships, key for effective clustering in the Fashion-MNIST dataset. The synergy of these features makes UMAP a highly effective dimensionality reduction technique for the Fashion-MNIST dataset, as evidenced by its outstanding performance in our analysis.

The consistency of LDA across different scenarios is also notable. The RAW data and other dimensionality reduction techniques, such as PCA, SAE, and CNN-SAE, tend to underperform across all metrics, reinforcing the value of sophisticated dimensionality reduction in clustering tasks.

5.2 Analysis of Training Time for Dimensionality Reduction Techniques

The following table shows the average training time for each dimensionality reduction technique.

Dimensionality Reduction Technique	Training Time in Seconds
PCA	7,753
LDA	18,245
UMAP	173,603
SAE	232,242
CNN - SAE	4893,747

Table 7: Training Time for Each Dimensionality Reduction Technique

The above results are also plotted diagrammatically.

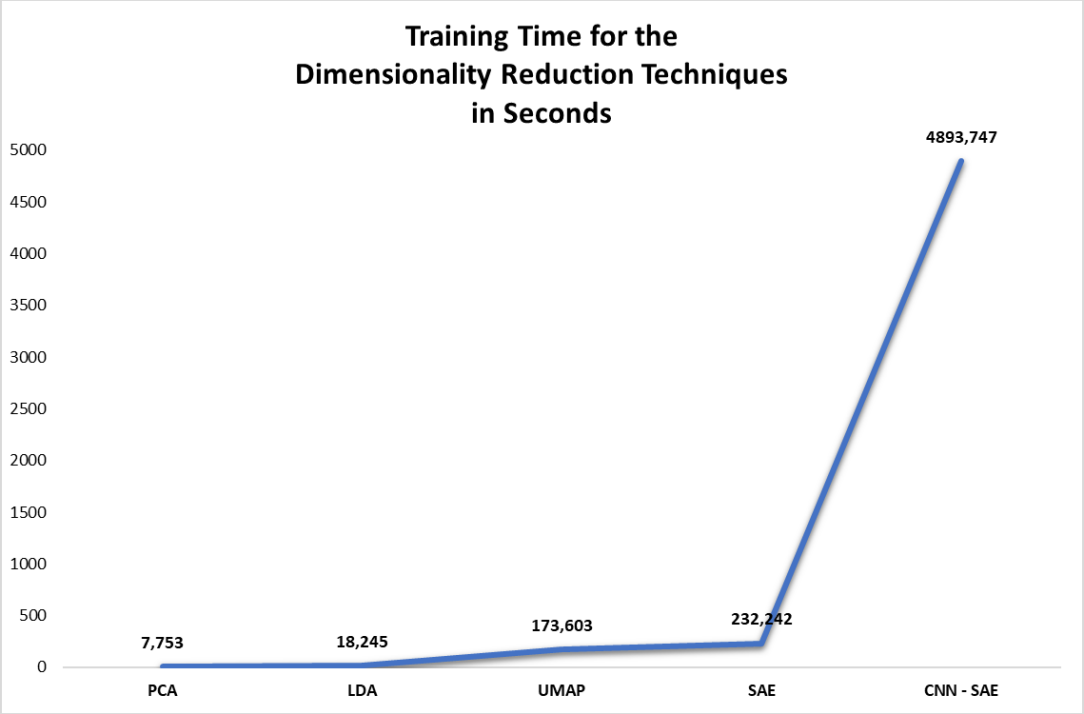


Figure 29: Training Time of Dimensionality Reduction Techniques

Based on the above information, it can be observed that PCA and LDA showcase a relatively quick training time, exemplifying efficiency in dimensionality reduction with durations of 7.7 and 18.2 seconds, respectively.

UMAP demands a more prolonged training period, underscoring the computational overhead inherent in implementing this nonlinear dimensionality reduction method. Despite its computational intensity, the training time for UMAP remains reasonably manageable, recorded at 173.6 seconds. The enhanced clustering outcomes achieved by UMAP demonstrate its substantial value, reinforcing that the additional computational effort is a worthwhile exchange for the quality of results obtained.

SAE exhibits a notable increase in training time, implying a more computationally intensive process for achieving sparsity in the learned representation, with a duration of 232.2 seconds.

CNN-SAE logs the longer training time among the techniques, signaling the intricacies associated with employing convolutional neural networks for dimensionality reduction, as reflected in the training time of 4893.7 seconds.

5.3 Balancing Performance and Efficiency in Dimensionality Reduction Techniques

Here, we aim to synthesize the insights derived from the comparative analysis of dimensionality reduction techniques in terms of their impact on clustering performance and the associated training times. The findings are summarized as follows:

PCA: PCA displays a moderate yet consistent improvement in clustering performance compared to raw data across all metrics. Given its minimal computational time, this indicates that PCA is a practical and effective choice for dimensionality reduction in our clustering task.

SAE and CNN-SAE: Both SAE and CNN-SAE do not consistently deliver significant improvements in clustering quality. Additionally, they exhibit considerably high training times, with CNN-SAE being the most computationally intensive. This might indicate that these techniques, particularly CNN-SAE,

are less suitable for the task at hand due to the disproportionate balance between performance gains and computational demands.

LDA: LDA not only requires relatively low computational time but also consistently offers significant improvements in clustering performance when compared to raw data. This underscores LDA's efficiency and effectiveness in enhancing the quality of clustering results.

UMAP: UMAP stands out with stark improvements in clustering performance, requiring a moderately high, yet manageable computational time. The substantial gains in clustering quality justify the additional computational effort, making UMAP an excellent choice for our task, where preserving the global data structure and enhancing clustering performance are paramount.

In summary, while PCA and LDA present themselves as time-efficient options with considerable performance benefits, UMAP's superior clustering outcomes, despite its higher computational cost, make it a compelling choice for the task at hand. Conversely, the high computational demands of SAE and CNN-SAE, combined with their inconsistent performance improvements, indicate that they may not be the most advantageous choices for dimensionality reduction in our clustering workflow.

5.4 Performance Evaluation of Clustering Algorithms

This section is dedicated to evaluating the performance of the various clustering algorithms both on the raw dataset and after the implementation of the dimensionality reduction techniques. We leverage the four selected performance metrics to benchmark the algorithms' effectiveness in capturing the true structure of the Fashion-MNIST dataset.

5.4.1 Calinski-Harabasz Index Analysis

We start by exploring the Calinski-Harabasz Index, which quantifies cluster validity based on intra-cluster dispersion and inter-cluster separation. The following chart displays a heatmap representing the Calinski-Harabasz Index values across all clustering algorithms and dimensionality reduction techniques:

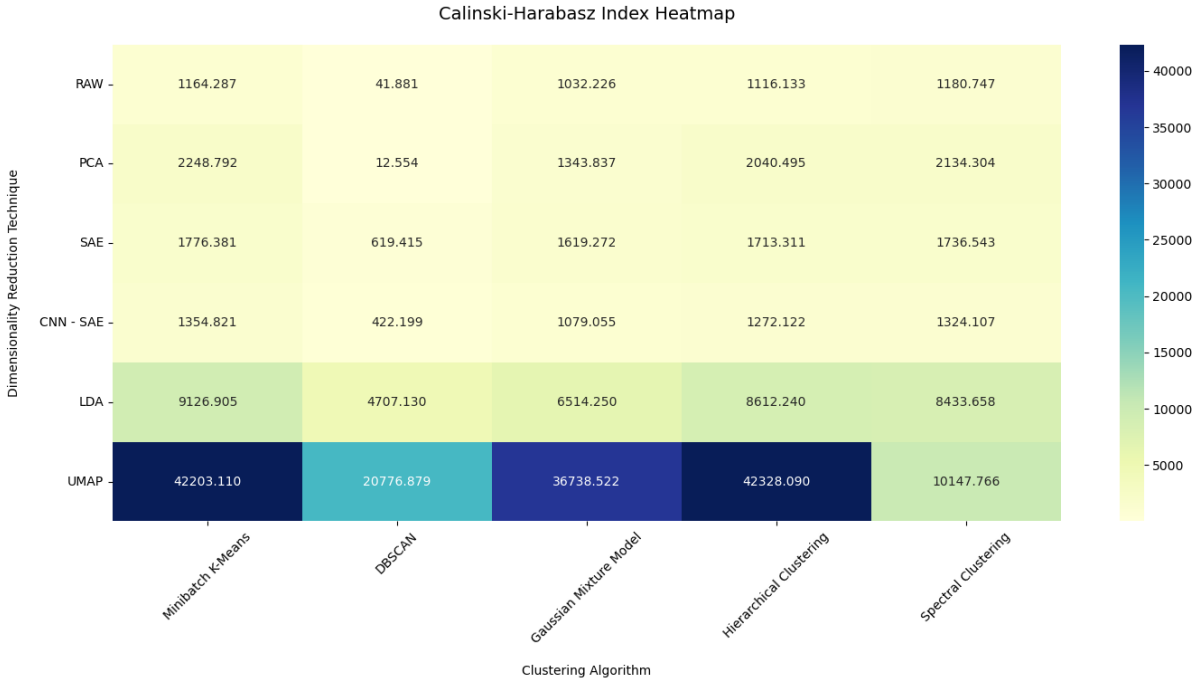


Figure 30: Performance Evaluation of Clustering Algorithms Based on the Calinski-Harabasz Index

The heatmap reaffirms the standout performance of the UMAP dimensionality reduction technique.

Hierarchical Clustering upon applying UMAP emerges as the top performer, achieving a Calinski-Harabasz Index of 42,328.090. This is closely followed by Minibatch K-Means, which, in combination with UMAP, attains a Calinski-Harabasz Index of 42,203.110. The Gaussian Mixture Model after UMAP also shows commendable performance with a resulting index of 36,738.522. These results underscore the effectiveness of these algorithmic combinations in producing distinct and cohesive clusters.

LDA exhibits consistently high performance across various clustering algorithms. Notably, Minibatch K-Means and Hierarchical Clustering achieve the best results when paired with LDA in terms of the Calinski-Harabasz Index, with values of 9,126.905 and 8,612.240, respectively.

It is noteworthy that even the lowest performing clustering algorithm when paired with UMAP, which is Spectral Clustering with a Calinski-Harabasz Index of 10,147.766, surpasses the highest performance of this index achieved by any clustering algorithm combined with other dimensionality reduction techniques.

The DBSCAN algorithm, which determines the number of clusters based on density, shows a significantly lower index value across all dimensionality reduction techniques. This might be due to the density distribution of the dataset, which may not align well with the algorithm's assumptions.

5.4.2 Davies-Bouldin Index Analysis

The Davies-Bouldin Index is a metric for evaluating clustering algorithms where a lower value indicates better clustering performance, with a preference for smaller within-cluster scattering and greater between-cluster separation. Figure 31 shows the heatmap with respect to the Davies-Bouldin Index:

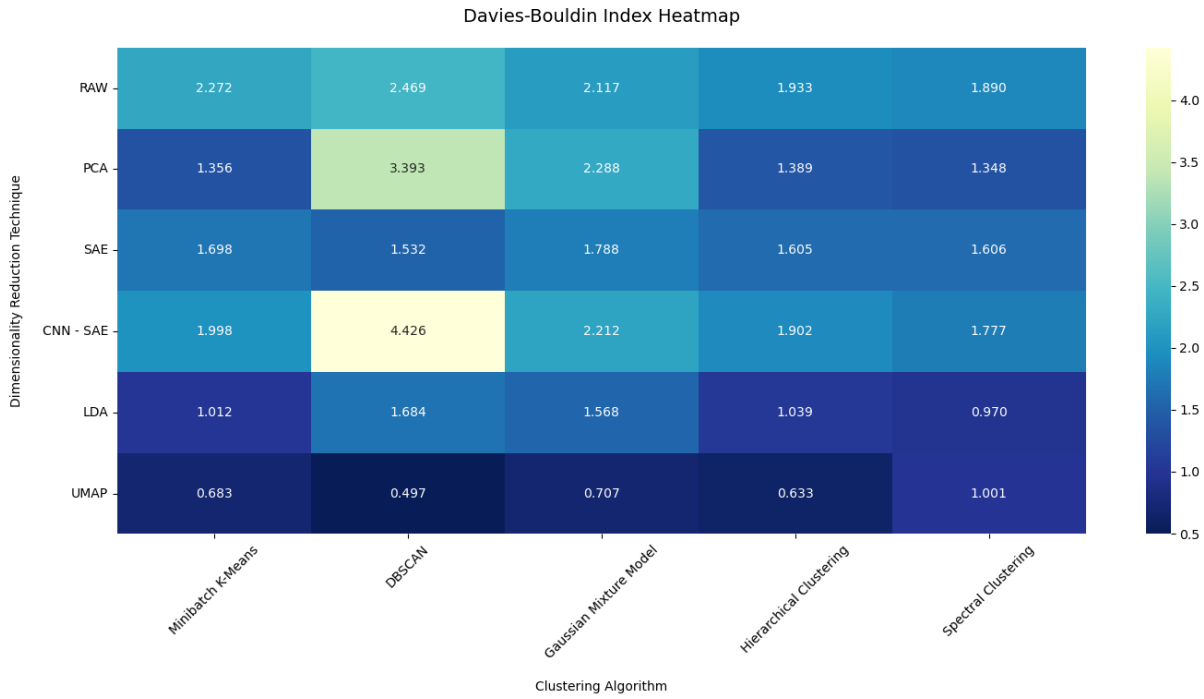


Figure 31: Performance Evaluation of Clustering Algorithms Based on the Davies-Bouldin Index

The Davies-Bouldin Index further solidifies UMAP's superiority in delineating clusters within the Fashion-MNIST dataset.

DBSCAN, when combined with UMAP, emerges as the top-performing algorithm, achieving a Davies-Bouldin Index of 0.497. Following closely is Hierarchical Clustering used after UMAP, which attains a Davies-Bouldin Index of 0.633. Minibatch K-Means, also in combination with UMAP, ranks third with a Davies-Bouldin Index of 0.683.

It is critical to acknowledge that DBSCAN, uniquely among the algorithms tested, cannot incorporate a predefined number of clusters. In our analysis, DBSCAN autonomously determined that four clusters were optimal, which likely contributed to its notably low Davies-Bouldin Index. However, this result doesn't necessarily provide a complete picture of clustering efficacy. Since DBSCAN's cluster count diverges from the expected ten, as predetermined for the other algorithms, its superior performance in this metric may not be directly comparable with those algorithms that adhered to the ten-cluster specification.

5.4.3 Silhouette Score Analysis

We now turn our attention to the Silhouette Score, which integrates both cohesion and separation to judge the proximity of data points within a cluster relative to other clusters. A higher Silhouette Score indicates better-defined clusters. The heatmap provided in Figure 32 presents the Silhouette Scores for all combinations of dimensionality reduction techniques and clustering algorithms.

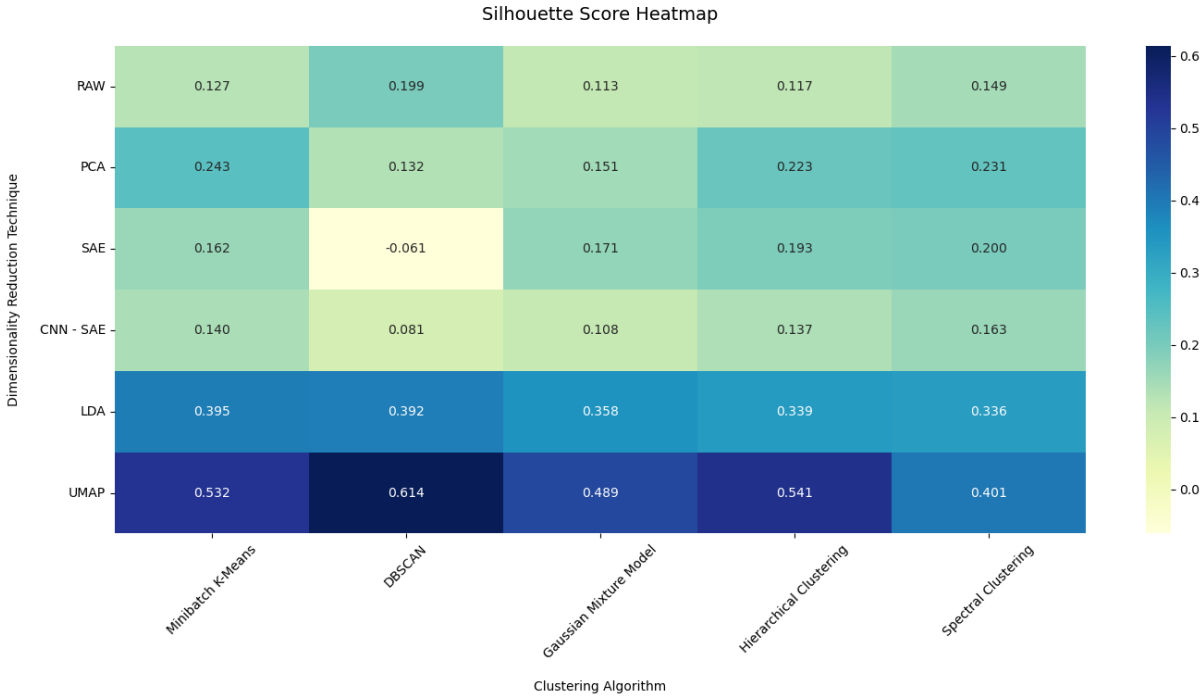


Figure 32: Performance Evaluation of Clustering Algorithms Based on the Silhouette Score

Post-implementation of UMAP as the dimensionality reduction technique, DBSCAN, Hierarchical Clustering, and Minibatch K-Means stand out as the only algorithms with a Silhouette Score exceeding 0.5. Their respective scores are 0.614 for DBSCAN, 0.541 for Hierarchical Clustering, and 0.532 for Minibatch K-Means.

It is important to reiterate, as previously noted for the Davies-Bouldin Index values, that the high Silhouette Score of DBSCAN might be similarly influenced by its identification of merely four clusters. This is in contrast to the ten clusters predefined for the other algorithms. Such a difference in cluster count can potentially inflate DBSCAN's score, as the task of delineating fewer clusters generally presents reduced complexity.

5.4.4 Adjusted Rand Index Analysis

Lastly, we turn our attention to the Adjusted Rand Index (ARI), a metric that stands apart from the others we have considered. Unlike the Calinski-Harabasz Index, Davies-Bouldin Index, or Silhouette Score, the ARI is unique in that it assesses clustering performance by comparing the clustering results

directly with the ground truth labels. Continuing with our established approach, we present the heatmap illustrating the Adjusted Rand Index results:

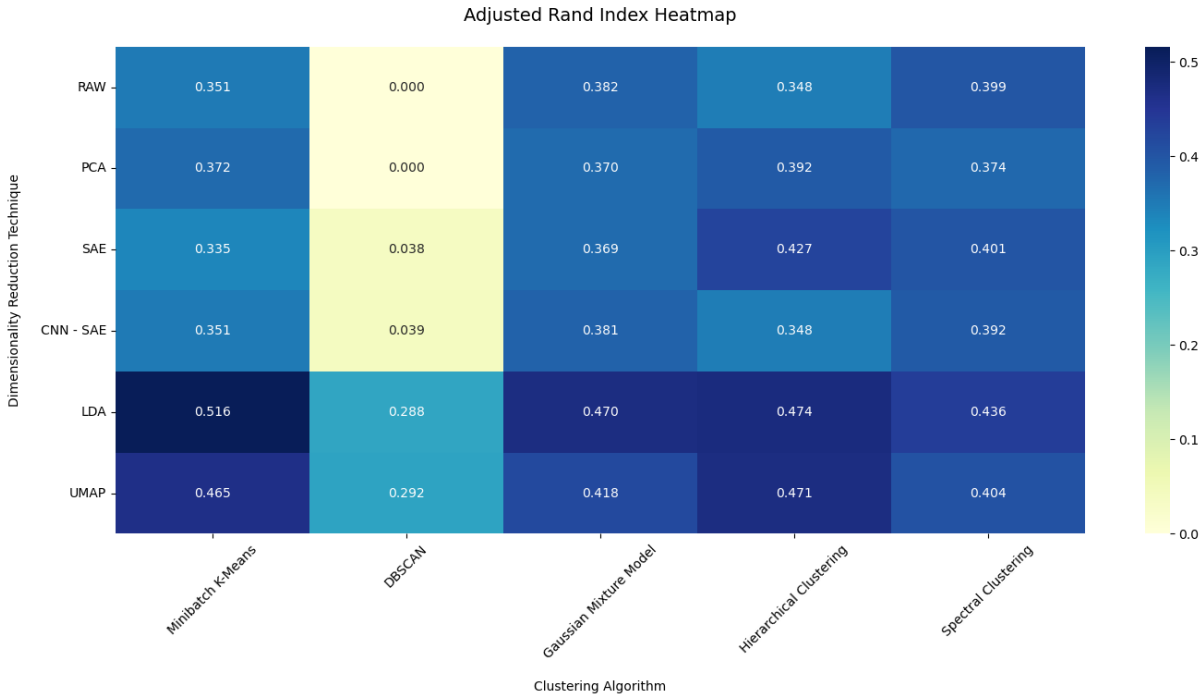


Figure 33: Performance Evaluation of Clustering Algorithms Based on the Adjusted Rand Index

As illustrated in Figure 33, the highest ARI is observed when LDA is used for dimensionality reduction, followed by Minibatch K-Means clustering, resulting in an ARI of 0.516. This indicates a strong agreement between the obtained clusters and the true labels. Hierarchical Clustering after LDA also performs very well, with an ARI of 0.474.

UMAP maintains its consistent performance with an ARI of 0.471 when combined with Hierarchical Clustering and 0.465 with Minibatch K-Means, reaffirming its robustness in clustering tasks when benchmarked against true labels.

Similar to its performance with the Calinski-Harabasz Index, DBSCAN exhibits underwhelming results when paired with all dimensionality reduction techniques in the Adjusted Rand Index metric. This may be attributed to DBSCAN's inherent mechanism of determining cluster count based on data density, which led to the prediction of only four clusters, rather than the ten present in the Fashion-MNIST dataset. Such a significant discrepancy in the number of clusters can adversely affect the ARI, since it compares the clustering results with the actual ground truth where the number of clusters is known to be ten.

5.4.5 Summary

Our comprehensive analysis reveals that UMAP, combined with Hierarchical Clustering emerges as the most effective combination in terms of the Calinski-Harabasz Index, indicating well-defined, distinct clusters.

Concerning the Davies-Bouldin Index and Silhouette Score, DBSCAN stands out as the best-performing algorithm. However, as previously discussed, this result may not fully reflect the overall clustering effectiveness. DBSCAN doesn't conform to the expected ten clusters specified for other algorithms, making its superior performance in these metrics not directly comparable to algorithms following the ten-cluster specification. Hierarchical Clustering combined with UMAP ranks as the second-best performer in these metrics, while adhering to the specified configuration of 10 clusters. The minimal disparity in

performances between DBSCAN and Hierarchical Clustering, coupled with the ability of Hierarchical Clustering to identify 10 clusters, renders it a suitable choice for the task.

Minibatch K-Means combined with UMAP, trailing closely behind Hierarchical Clustering, demonstrates commendable results with respect to all three previous metrics. This underscores the validity of Minibatch K-Means as a reliable clustering technique for our dataset.

Regarding the Adjusted Rand Index, Minibatch K-Means, after dimensionality reduction with LDA, showed the strongest performance. Hierarchical Clustering, following dimensionality reduction with LDA, also exhibits commendable performance on the Adjusted Rand Index, highlighting the robust and stable nature of its clustering outcomes.

The superiority of LDA over UMAP in terms of ARI can be primarily attributed to LDA's supervised nature and its focus on maximizing class separability, which makes it particularly effective for achieving clustering outcomes that closely match the true class labels in datasets like Fashion-MNIST.

Lastly, it is important to emphasize that for the Fashion-MNIST dataset, both the number and labels of the clusters were predetermined. Consequently, to accurately reflect the dataset's characteristics, we set the number of clusters to ten for all algorithms, with the exception of DBSCAN. This decision played a significant role in influencing the performance of the algorithms and the resulting metric values. In scenarios where the true number of clusters is unknown, the performance of these algorithms might differ, highlighting the importance of context in evaluating clustering efficacy. Therefore, while our findings provide valuable insights, they are specifically tailored to scenarios where cluster labels and counts are pre-established, a condition that may not always be present in real-world applications.

5.5 Analysis of Execution Time for Clustering Algorithms

The execution time of clustering algorithms is a crucial factor in assessing their practicality and efficiency. This section delves into the time performance of various clustering algorithms, with and without the application of dimensionality reduction techniques.

Figure 34 illustrates the average execution time of each clustering algorithm. This provides a direct comparison of the algorithms' computational demands.

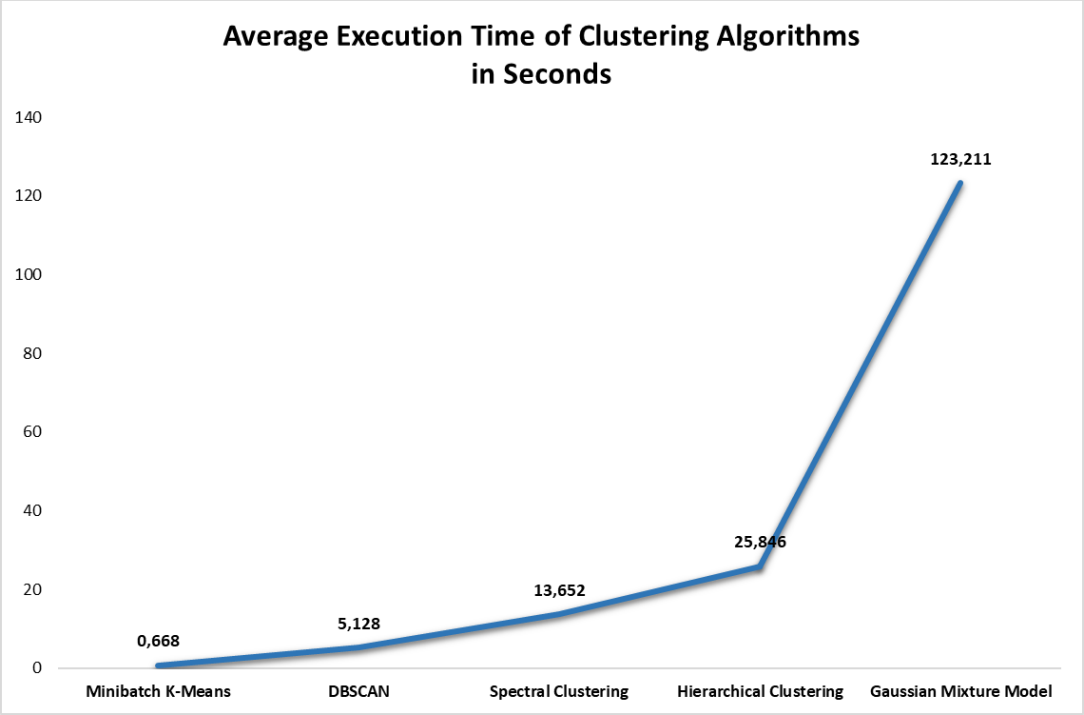


Figure 34: Average Execution Time of Clustering Algorithms

The chart highlights that Gaussian Mixture Models demand considerably more time, averaging around 123 seconds for execution. Hierarchical Clustering follows, yet with a substantial margin, necessitating an average execution time of approximately 26 seconds. Remarkably, Minibatch K-means operates with an impressive efficiency, requiring less than one second on average for execution.

The next figure presents the execution time for each combination of dimensionality reduction technique and clustering algorithm. This comprehensive overview allows us to assess the impact of dimensionality reduction on the time efficiency of clustering.

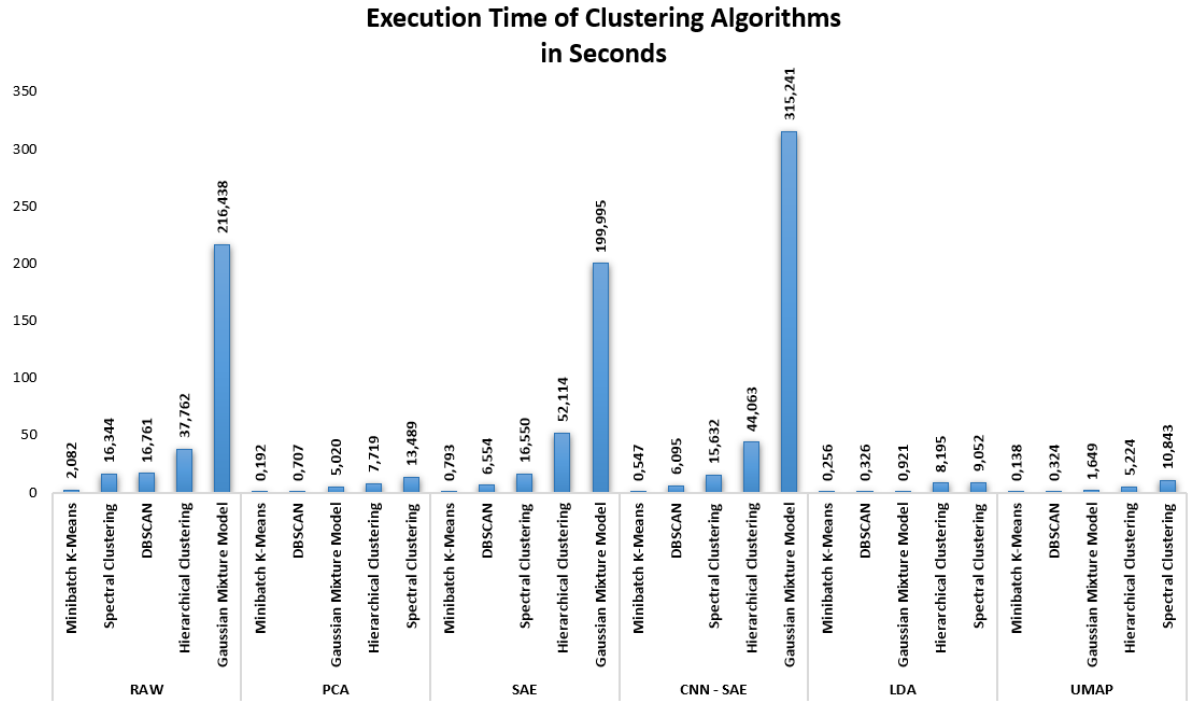


Figure 35: Execution Time for Dimensionality Reduction and Clustering Combinations

The data clearly shows that the choice of dimensionality reduction technique can significantly impact computational time. Techniques such as UMAP, LDA, and PCA, when applied before clustering, markedly enhance execution efficiency. This is particularly noticeable with time-intensive algorithms like Gaussian Mixture Models and Hierarchical Clustering, where the preliminary use of these dimensionality reduction methods leads to a notable reduction in the time required for computation.

Lastly, figure 36 depicts the average execution time of the clustering algorithms after applying each dimensionality reduction technique, as well as on the raw data. This allows us to evaluate the efficiency of clustering post-dimensionality reduction.

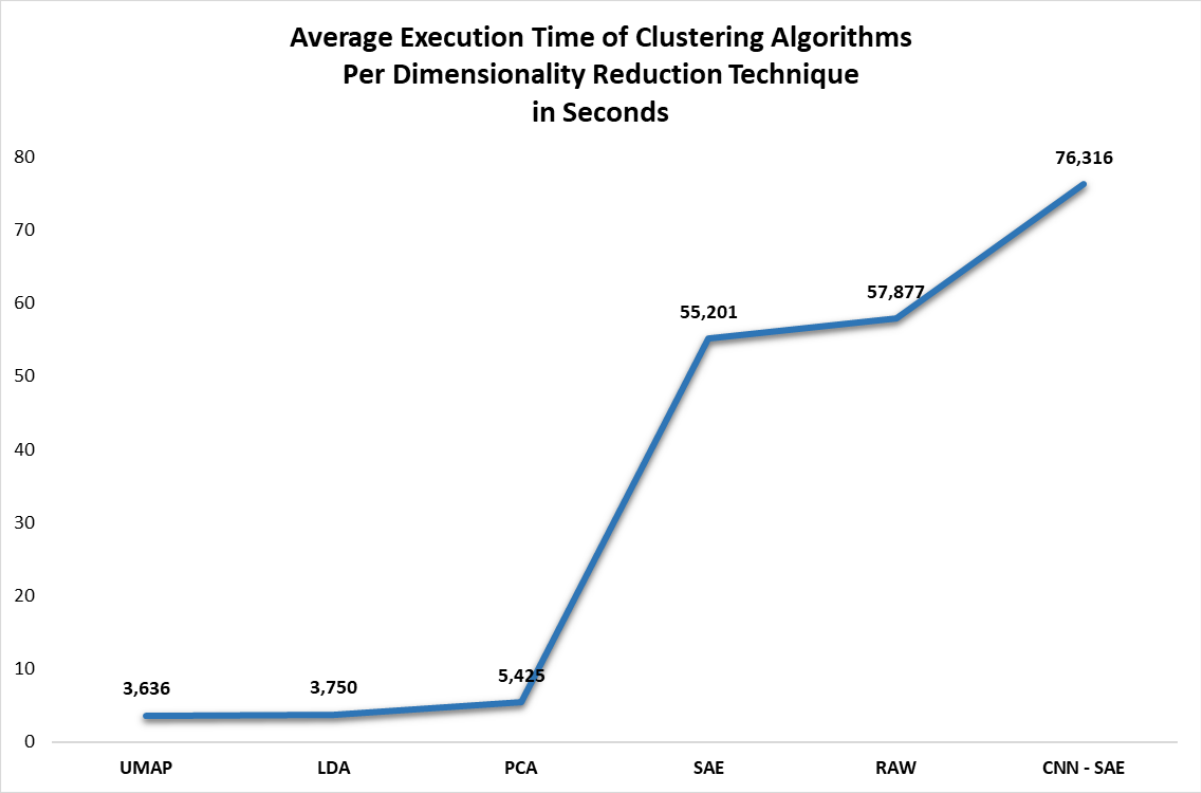


Figure 36: Average Execution Time of Clustering Algorithms Per Dimensionality Reduction Technique

The chart reinforces the observation that employing dimensionality reduction techniques such as UMAP, LDA, and PCA significantly enhances the computational efficiency of subsequent clustering algorithms, as opposed to clustering on the raw dataset directly. While SAE offers a modest improvement in computational time, the use of CNN-SAE appears to extend the duration required, indicating a potential increase in complexity.

6 Conclusion

This report aimed to evaluate the effectiveness of various dimensionality reduction techniques coupled with clustering algorithms, and to assess the performance of these clustering algorithms on both the dimensionally reduced and raw Fashion-MNIST dataset. Our analysis encompassed five dimensionality reduction techniques (PCA, SAE, CNN-SAE, LDA, and UMAP) applied in conjunction with five clustering algorithms (MiniBatch K-Means, DBSCAN, Gaussian Mixture Models, Hierarchical Clustering, and Spectral Clustering). Furthermore, we directly implemented these clustering algorithms on the raw dataset to establish a baseline for comparison. The evaluation was based on four key metrics: Calinski-Harabasz Index, Davies-Bouldin Index, Silhouette Score, and Adjusted Rand Index.

6.1 Best Dimensionality Reduction Technique

UMAP consistently demonstrated superior performance, significantly elevating clustering outcomes by achieving the highest scores in three out of the four examined metrics. Its capability to efficiently reduce dimensionality while preserving essential data structures makes it particularly suitable for the Fashion-MNIST dataset. Furthermore, UMAP stands out as a time-efficient methodology. While its training time may be relatively extended, the implementation of UMAP for dimensionality reduction effectively minimizes the execution time of subsequent clustering algorithms, thereby enhancing overall computational efficiency.

Conversely, LDA emerged as the superior technique specifically in the Adjusted Rand Index, a metric that evaluates the similarity of the clustering results to the true class labels. This distinct advantage suggests that LDA is particularly adept at tasks where the primary objective extends beyond mere clustering to accurately identifying the actual labels of the data. This characteristic renders LDA more suitable than UMAP for scenarios where precise label identification is paramount. Additionally, it is noteworthy that LDA not only demonstrates a brief training duration but also significantly reduces the execution time of subsequent clustering algorithms.

6.2 Best Combination of Dimensionality Reduction and Clustering

The varied performance of the clustering algorithms across different metrics underscores the importance of choosing the right combination of dimensionality reduction technique and clustering algorithm based on the specific requirements of the task.

In the context of the Fashion-MNIST dataset, the combination of UMAP with Hierarchical Clustering stands out as a particularly effective choice. This pairing excels at producing well-separated and cohesive clusters, demonstrating a superior performance in revealing the inherent structure of the data. Moreover, our analysis revealed that this combination is not only high-performing but also time-efficient in terms of implementation. This blend of superior clustering quality and computational efficiency makes it an optimal choice for analyzing the Fashion-MNIST dataset.

The combination of Minibatch K-Means with UMAP presents a compelling alternative, achieving comparable results, with the added advantage of the shortest implementation time. This makes it a compelling option for tasks involving large datasets or scenarios where time efficiency is paramount.

If the primary objective is to attain clusters closely aligned with the true labels, the optimal choice is to employ LDA in tandem with MiniBatch K-Means. This is evident in the superior performance of this combination on the Adjusted Rand Index, a metric precisely designed to gauge the similarity between predicted and true cluster labels. Additionally, this combination stands out for its efficiency in terms of computational time.

6.3 Final Insights

The synergy between dimensionality reduction techniques and clustering algorithms is critical. It allows for the extraction of meaningful patterns from complex datasets such as Fashion-MNIST, ensuring that the performance of clustering algorithms is not only maintained but significantly improved, striking an optimal balance between effective clustering and computational efficiency. In essence, dimensionality reduction is not merely a data preprocessing step but a transformative process that profoundly influences the quality of clustering.

Our analysis reveals that UMAP, combined with Hierarchical Clustering, stands out as the most effective strategy for producing distinct and cohesive clusters within the Fashion-MNIST dataset. Nevertheless, our findings underscore the importance of a customized approach in the selection of dimensionality reduction techniques and clustering algorithms, affirming that their effectiveness is highly dependent on the dataset and the objectives of the analytical task at hand. It is also imperative to weigh the trade-offs between computational demand and performance gain. This balance is crucial for optimizing the efficiency of unsupervised learning tasks, particularly when dealing with large and complex datasets or in situations where computational resources are limited.

While our analysis provides valuable insights, it is essential to acknowledge its limitations, such as the specific focus on the Fashion-MNIST dataset and the range of techniques evaluated. Future research could explore these techniques across more diverse datasets and incorporate emerging dimensionality reduction methods to further our understanding. The practical implications of our findings are significant, offering data scientists and machine learning practitioners guidance in selecting the most suitable techniques for their specific data and objectives.