*AIDA105 Machine Learning and Computer Vision*

**Unsupervised Machine Learning (Clustering)**

**Dimitrios Mitras aid24005, Konstantinos Bougioukas aid24011**

**1. Introduction**

In unsupervised learning, clustering is a type of machine learning where the algorithm is given unlabeled data and tasked with finding the underlying patterns or structure in the data. Choosing the appropriate clustering algorithm depends on the characteristics of the data and the desired outcomes. A good strategy is to try multiple algorithms and evaluate their performance based on specific criteria, such as cluster quality, interpretability, and computational efficiency. In this homework, our focus is on the case of clustering methods. Specifically, our aim is to compare and evaluate the performance of five different clustering methods combined with dimensionality reduction techniques.

**2. Problem description**

We want to apply a number of different clustering algorithms on raw and dimensionally reduced data of fashion images and evaluate their performance regarding specific metrics.

**3. Methods**

*3.1.* ***Dataset description***

Fashion-MNIST is a dataset of [fashion](https://en.wikipedia.org/wiki/Fashion) images consisting of a training set of 60,000 examples and a test set of 10,000 examples (**Fig.** 1) Each example is a 28x28 grayscale image, associated with a label from 10 classes.

**Labels**

Each training and test example is assigned to one of the following labels:

* 0 T-shirt/top
* 1 Trouser
* 2 Pullover
* 3 Dress
* 4 Coat
* 5 Sandal
* 6 Shirt
* 7 Sneaker
* 8 Bag
* 9 Ankle boot



**Figure 1.** Sample images from fashion MNIST test dataset

*3.2.* **Dimensionality reduction**

After scaling all measured features using the normalisation technique (values [0-1]), we applied the following dimensionality techniques to the dataset:

1. **Principal component analysis (PCA):** Principal component analysisis a statistical process that allows reducing the dimensionality of a dataset while preserving as much of the original variability as possible. This is achieved by transforming the data into a new coordinate system, where the axes are the principal components. However, interpreting the principal components may not always be straightforward, especially when dealing with a large number of features.
2. **Factor analysis (FA):** While PCA aims to capture the maximum variance in the data, Factor Analysis focuses on identifying the underlying factors that contribute to the observed variables. The factor analysis describes the covariance associations among many variables in terms of a few underlying, but unobservable, random quantities called factors. Factor analysis assumes that the variables can be grouped by their correlations.
3. **Stacked autoencoder (SAE):** It is a nonlinear generalisation of PCA that uses an adaptive, multilayer encoder network to transform the high-dimensional data into a low-dimensional code and a similar decoder network to recover the data from the code.
4. **Convolutional stacked autoencoder (CSAE):** A Convolutional Stacked Autoencoder (CSAE) is an extension of the traditional stacked autoencoder architecture, specifically designed for handling structured grid-like data such as images. The advantage of using convolutional layers is that they enable the autoencoder to capture local patterns and spatial dependencies, which is crucial for image data.
5. **Fast algorithm for Independent Component Analysis (Fast ICA):** ICA aim is similar in many aspects to principal component analysis (PCA) and factor analysis. The key difference of ICA from PCA and FA is that ICA aims for the statistical independence of the resulting components, while the components obtained using PCA and FA are only linearly independent, which in general does not imply statistical independence. A computationally very efficient method performing the actual estimation of ICA is given by the FastICA algorithm (ref.)

*3.3.* ***Classification methods used in this work***

1. **Minibatch K-Means:** The Mini-batch K-means partitioning-based algorithm is a variant of the traditional K-means clustering algorithm that is designed to work with large datasets more efficiently. While the standard K-means algorithm processes the entire dataset in each iteration, mini-batch K-means processes only a random subset or "mini-batch" of the data in each iteration.
2. **DBSCAN:** Density-Based Spatial Clustering of Applications with Noise is a density-based clustering algorithm for grouping together data points that are close to each other in high-dimensional space. Unlike K-means, DBSCAN does not require specifying the number of clusters beforehand and can find clusters of arbitrary shapes.
3. **Agglomerative Hierarchical Clustering:** It is a type of clustering algorithm that builds a hierarchy of clusters. It is a bottom-up approach, meaning that it starts with individual data points and progressively merges them into larger clusters. The result is a tree-like structure known as a dendrogram.
4. **Expectation-Maximization algorithm (EM) for Gaussian Mixture Model (EM-GMM):** It is a model-based approach for clustering when the data is assumed to be generated from a mixture of several Gaussian distributions (K-Means is a special case of GMM). In order to find the parameters of the Gaussian for each cluster (e.g the mean and standard deviation), Expectation–Maximization (EM) optimization algorithm is used.
5. **Bisecting K-Means:** Bisecting K-Means is a hybrid approach between partitional and hierarchical top down clustering. Instead of partitioning the data set into K clusters in each iteration, it splits one cluster into two sub clusters at each bisecting step (by using k-means) until k clusters are obtained.

*3.4.* ***Performance Metrics***

To validate the results of clustering methods, the following four indexes were used:

1. **Calinski–Harabasz index:** It provides a way to assess the compactness of clusters and the separation between clusters in a dataset. The index is calculated based on the ratio of the between-cluster variance to the within-cluster variance. The interpretation of the index is such that higher values indicate better-defined clusters.
2. **Davies–Bouldin index:** Similar to the Calinski–Harabasz index, the Davies–Bouldin index evaluates the compactness and separation between clusters in a dataset. The range of the Davies–Bouldin Index is theoretically between 0 and positive infinity. A lower DBI indicates more compact and well-separated clusters.
3. **Silhouette score:** The Silhouette Score is a metric used to calculate the goodness of a clustering technique, indicating how well-separated the clusters are. The Silhouette Score ranges from -1 to 1, where: A high Silhouette Score indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. A low Silhouette Score indicates that the object is poorly matched to its own cluster and well matched to neighboring clusters. A score around 0 indicates overlapping clusters.
4. **Adjusted Rand Index (ARI):** The Adjusted Rand Index (ARI) is a measure of the similarity between two data clusterings, adjusting for chance. The higher the ARI value, the closer the two clusterings are to each other. Even though ARI takes negative values, generally considered range for ARI is [0, 1]. ARI value 0 indicates real and modeled clustering do not agree on pairing, ARI value 1 indicates real and modeled clustering both represent the same clusters.

***3.5. Implementation in python***

In this section, the flow and a few details about the code will be explained briefly. Notably, some of the code chunks used in this work were found in the course notes.

Initially, the dataset was divided into training, validation, and test sets. Subsequently, the data was converted into 1D arrays and normalized to the range [0, 1]. The shape of the sets are:

X\_train.shape: (54000 , 784)

X\_valid.shape: (6000 , 784)

X\_test.shape: (10000 , 784)

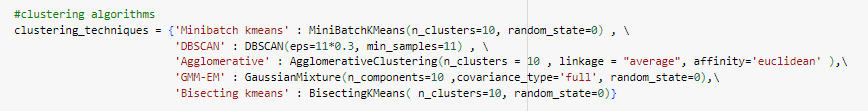
(**Normalize Data** cell in Notebook)

Afterwards, each dimensionality reduction technique was trained using the X\_train dataset, and the X\_test dataset was subsequently transformed into a new dataset with reduced dimensions. The following subsection provides a list of the parameters used, accompanied by a plot confirming the efficacy of the chosen technique.

(**PCA - FAST\_ICA** cells in Notebook)

The original X\_test, along with the datasets generated from the dimensionality reduction techniques, will be utilized for every clustering methods.

Given that we know the categories of the original dataset are 10, in the clustering methods (Bisecting KMeans, GMM-EM, Agglomerative, MiniBatchKMeans), the parameter n\_clusters was set to 10. For DBSCAN, attempts were made to configure the parameters appropriately to obtain clusters close to 10, but achieving this was challenging, as DBSCAN struggles with dense data. For Agglomerative the distance of the clusters compute as linkage=’average’



(**MAIN** cell in Notebook)

Metrics were calculated for the cluster\_labels generated by the techniques and subsequently stored in a data frame named 'results.' This data frame is exported at the end of the code.

(**Clustering Metrics** cell in Notebook)

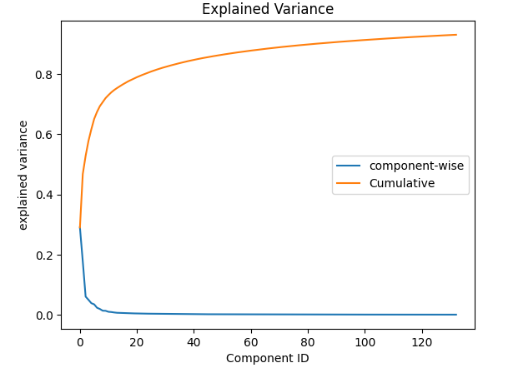
***3.5.1 Dimensionality Reduction Techniques***

For each dimensionality reduction technique, a new dataset was created. Below are presented the parameters used for each technique and a plot illustrating the performance of the technique.

*PCA*

n\_components = 0.93

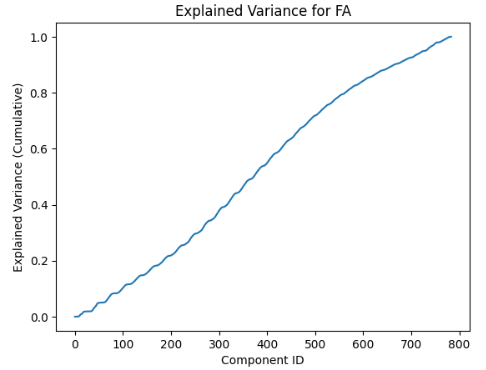
number of components selected: 133

time: 12.86s

*FA*

number of components selected: 120

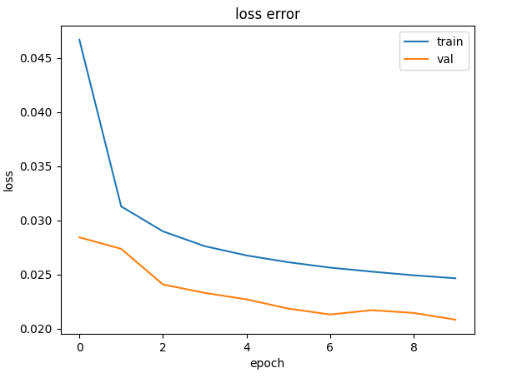
time: 74.3s



*SAE*

It was deemed preferable to use epoch=10 both in terms of time efficiency and performance.

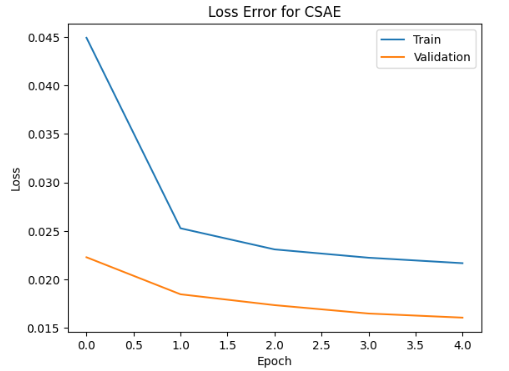
time: 265.1



*CSAE*

epoch = 5

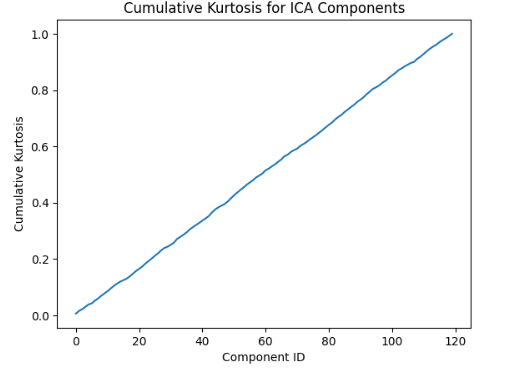
time: 504.1



*FAST\_ICA*

n\_components = 120

time: 43.7

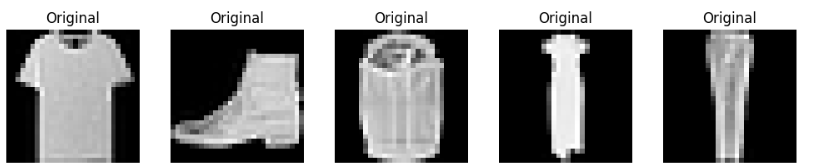


**4. Results**

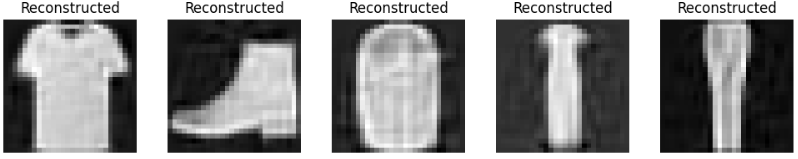
***4.1 Comparison of Original and Reconstructed for each Technique***

On the next two pages, a selection of random images from the X\_test dataset and the new images resulting from the dimensionality reduction techniques are presented. Visually, it is observed that the differences are very small, and at this point, no technique appears to be significantly better than the others.

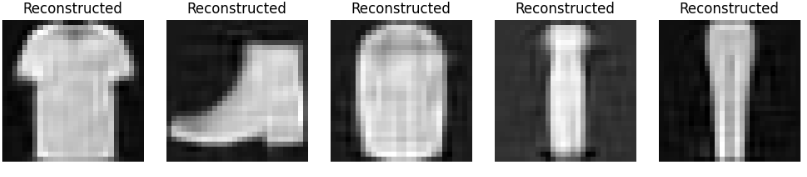
**(1)**



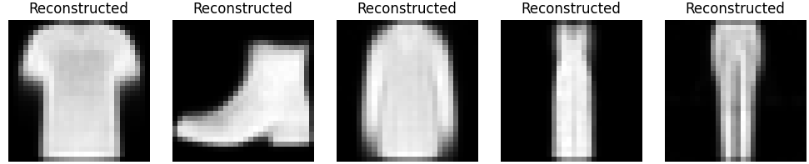
PCA



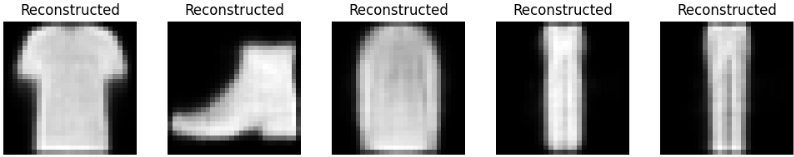
FA

**

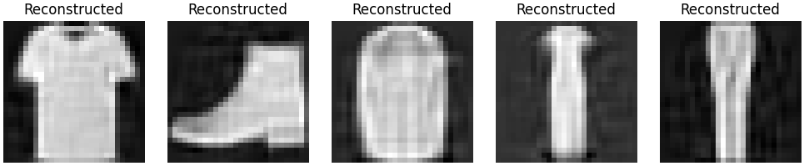
SAE

**

*CSAE*

**

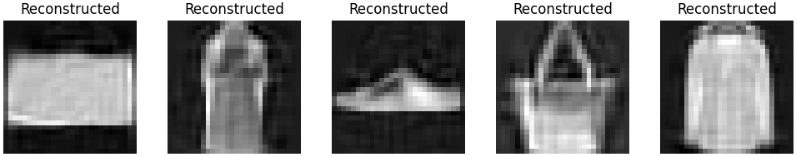
FAST\_ICA



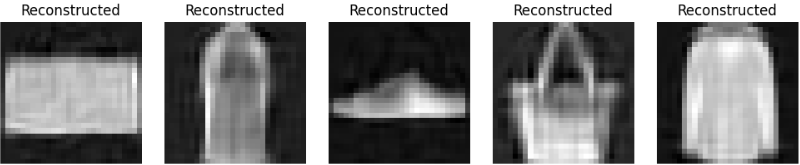
**(2)**



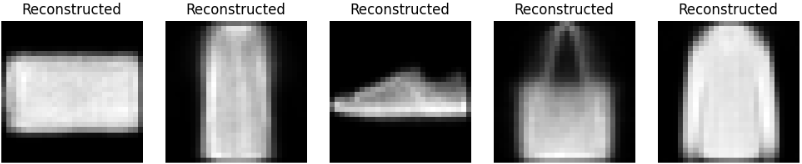
PCA



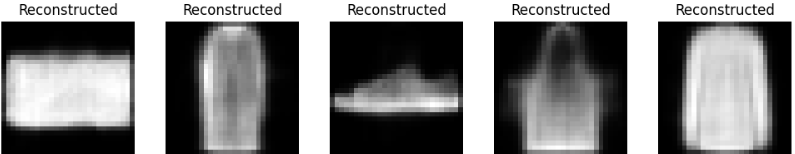
FA

**

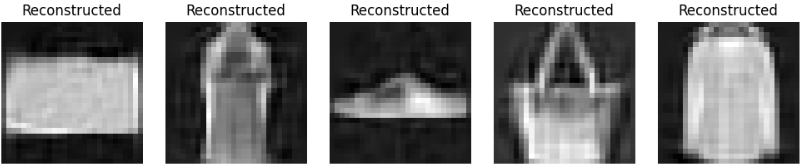
SAE

**

*CSAE*

**

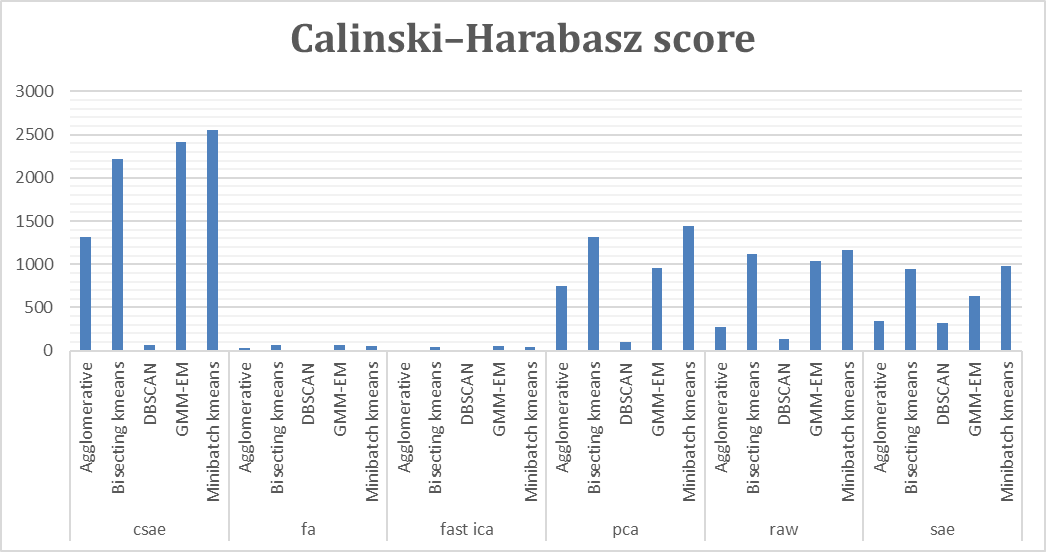
FAST\_ICA



***4.2.* *Comparison of performance scores of clustering methods***

***4.4.1.* Calinski–Harabasz score**

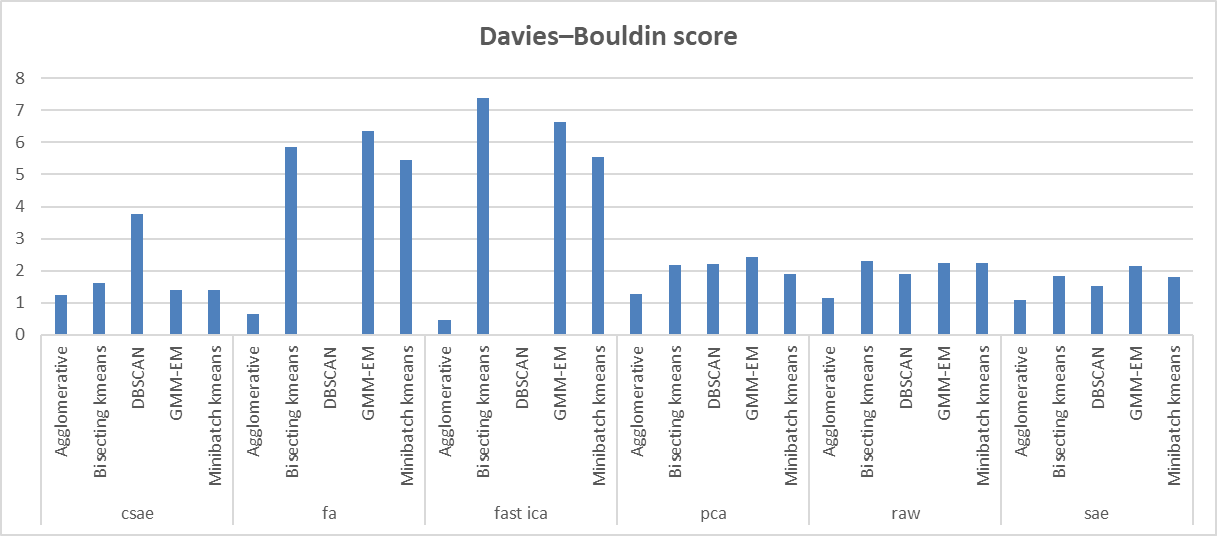
We observe that the application of Convolutional stacked autoencoder (CSAE) for dimensionality reduction compimned with Minibatch K-Means or GMM-EMalgorithms for clustering seem to perform better on Calinski–Harabasz score. Additionally, PCA and Stacked autoencoder (SAE) combined with clustering methods have similar performance to the raw data, while FA and Fast ICA have very low values for all clustering methods.



**Figure 2.** This bar plot shows the Calinski–Harabasz score for every dimensionality technique and clustering method.

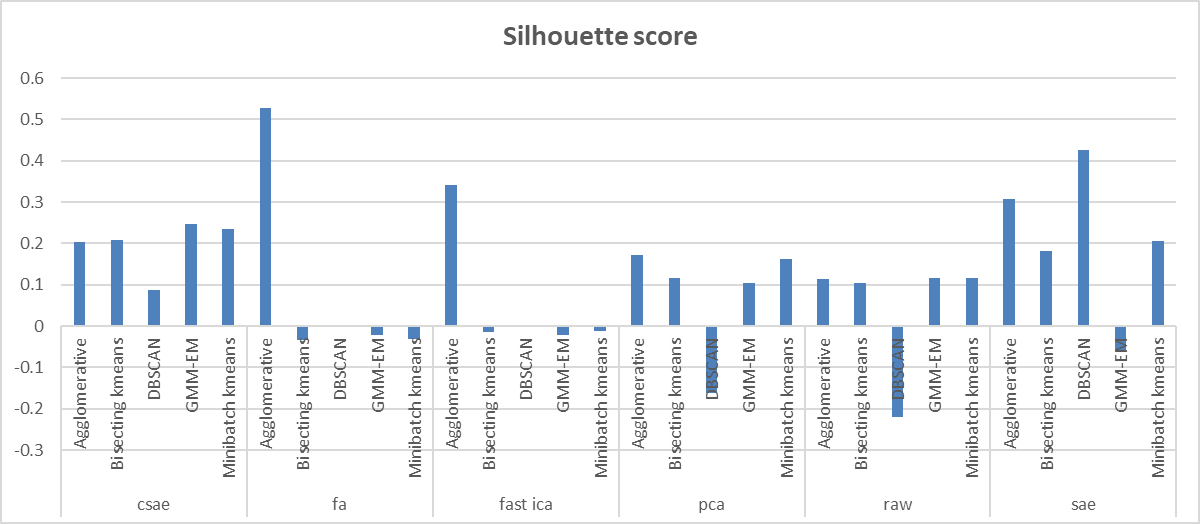
***4.4.2.* Davies–Bouldin score**

We observe that the application of Factor Analysis (FA) and Fast ICA for dimensionality reduction combined with the Agglomerative Hierarchical Clustering seem to perform better on Davies–Bouldin score (lower values). Note that the DBSCAN does not work well for these methods.



**Figure 3.** This bar plot shows the Davies–Bouldin score for every dimensionality technique and clustering method.

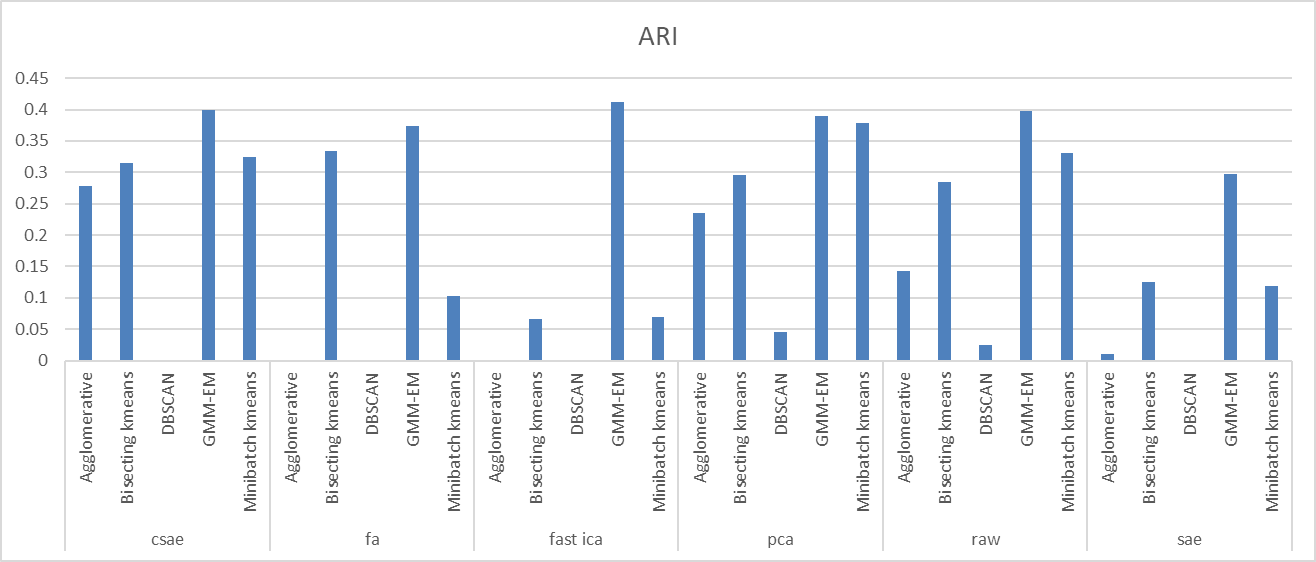
***4.4.3.* Silhouette Score**

We observe that the application of Factor Analysis (FA), Fast ICA, and Stacked autoencoder (SAE) for dimensionality reduction combined with Agglomerative Hierarchical Clustering, as well as DBSCAN with SAE, seem to perform better on Silhouette Score. 

**Figure 4.** This bar plot shows the Silhouette Score for every dimensionality technique and clustering method.

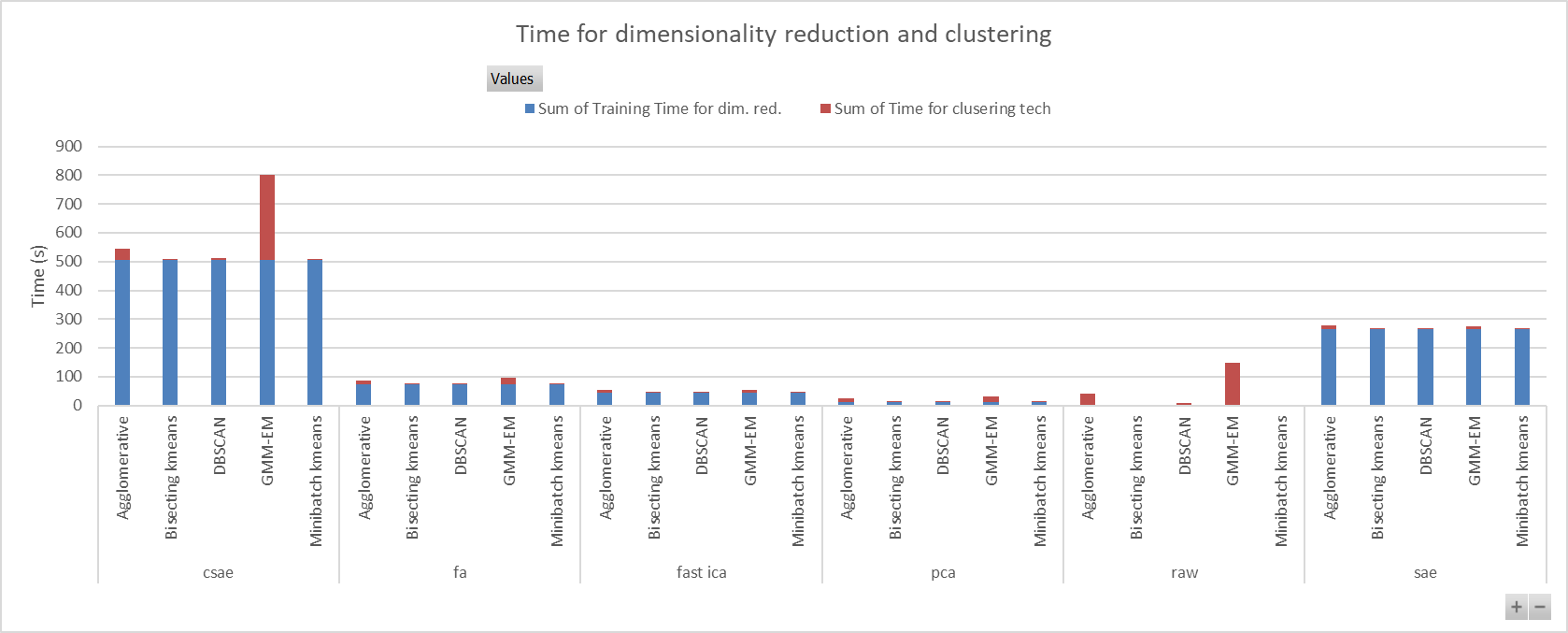
***4.4.4.* Adjusted Rand Index**

We observe that the application of GMM-EMalgorithms for clustering seem to perform better on Adjusted Rand Index for all dimensionality techniques.



**Figure 5.** This bar plot shows the Adjusted Rand Index Score for every dimensionality technique and clustering method.

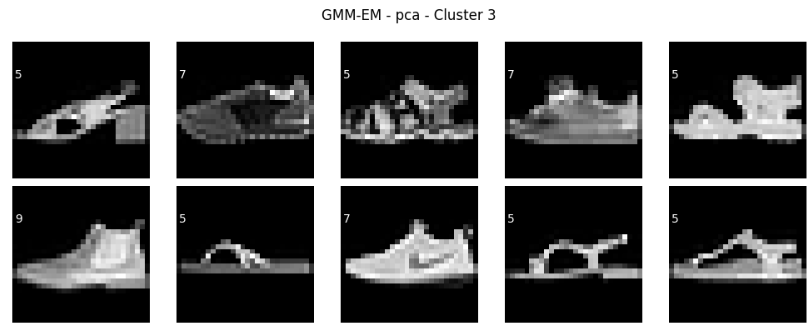
***4.4.5. T*raining time for dimensionality reduction and clustering**

We observe that PCA is the fastest method. On the other hand, the CSAE is the most time consuming (504 s) compared to the other methods (SAE 265 s, FA 74 s, Fast ICA 44 s, PCA 12s) regarding the time for dimensionality reduction. Additionally, the GMM-EM algorithm takes more time to run (295 s) with the CSAE method.

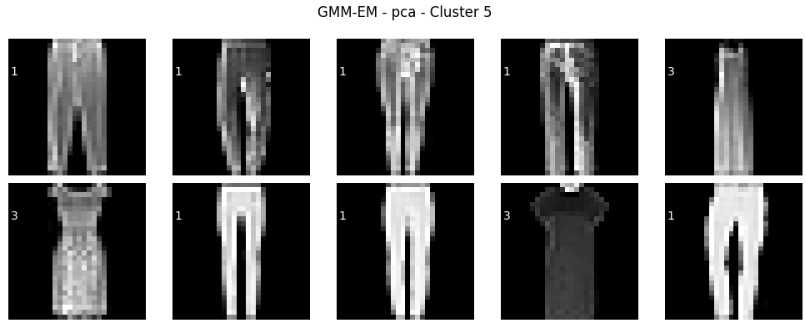
**Figure 6.** This bar plot shows the training time for dimensionality reduction (blue) and clustering (red).

***4.4.5. Sample examples form clusters produced by applying PCA and GMM-EM***

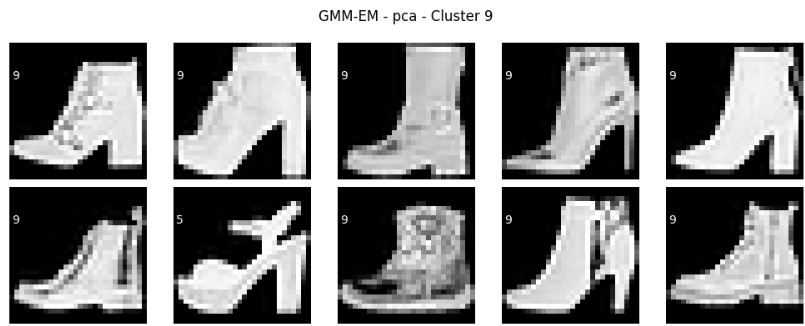


We sampled 10 images from the cluster 3 produced by applying PCA and GMM-EM algorithm which is fast and performs well. We observe that more than half of the images belong to the label 5 (Sandal), three images have the label 7 (Sneaker) and one the label 9 (Ankle boot).

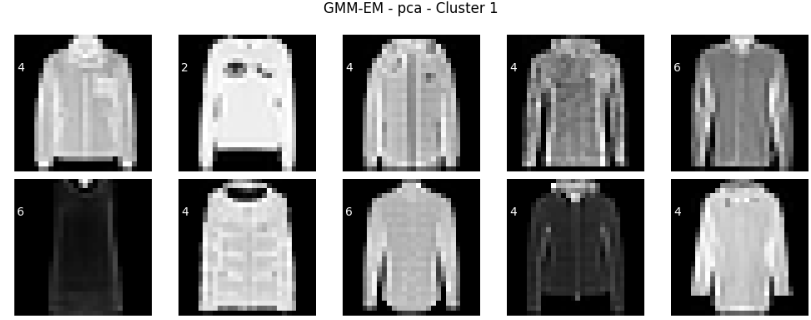
We also sampled 10 images from the cluster 5 produced by applying PCA and GMM-EM. We observe that seven of the images belong to the label 1 (Trouser), and three images have the label 3 (Dress).



Next, we have 10 sampled images from the cluster 9 produced by applying PCA and GMM-EM. We observe that nine of the images belong to the label 9 (Ankle boot), and only one has the label 5 (Sandal).



Next, we have 10 sampled images from the cluster 1 produced by applying PCA and GMM-EM. We observe that six of the images belong to the label 4 (Coat), three have the label 6 (Shirt) and one has the label 2 (Pullover).



**5. Conclusions**

We have presented a comparative study of five clustering methods using different techniques for dimensionality reduction. Using the Convolutional stacked autoencoder (CSAE), the optimal clustering algorithm regarding the Adjusted Rand Index (ARI) and Calinski–Harabasz score seems to be the GMM-EMalgorithm among the ones explored. However, the application of Factor Analysis (FA), Fast ICA for dimensionality reduction combined with Agglomerative Hierarchical Clustering seem to perform better on Silhouette Score and Davies–Bouldin score. Note that the DBSCAN does not work well for these methods.Furthermore, the CSAE method combined with the GMM-EM algorithm takes more time to run (overall 799 s) while the fastest method that performs well is the PCA (overall 31 s) and may be a preferred choice.