Fashion-MNIST Unsupervised Clustering

Task Overview

The goal of this task is to cluster images from the Fashion-MNIST dataset into distinguishable clusters. The main goal is to cluster them into 10 clusters that closely align to their labels which will not be used for the training process. This task will also test 3 different clustering methods being, K-Means, Gaussian Mixture and Agglomerative Clustering. With this being a image dataset, various dimensionality reductions will be evaluated being: PCA, Autoencoder and UMAP. The dataset used can be found on Kaggle.

Initial Data Exploration

The data was loaded into a Pandas DataFrame and resulted in the following findings:

- The data consists of 60000 Rows and 785 Columns.
- The data contains pixel values between 0 and 255 for each of the 784 grayscale pictures.
- 10 labels for the images in the dataset.

The dataset was chosen due to its variety of small images that make the clustering task not too demanding on a personal computer, with runtimes that are manageable for a task of this scale.
The 10 different classes for this set can be seen in Figure 1 with a sample image from each class.

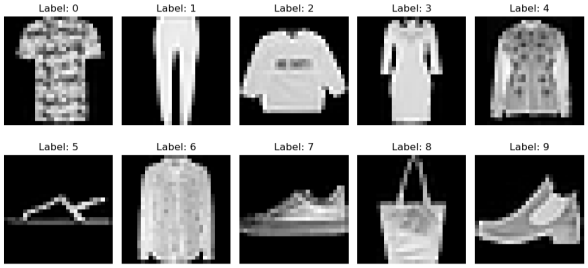

Figure 1: Sample Images from the Fashion-MNIST dataset

The 10 labels for these classes correspond to the following items: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag and Ankle boot. As some of these are very similar, I expect there to not be many distinct clusters due to the similarity of many of these such as boots and sneakers or the t-shirts and pullovers.

Model Selection

This section will discuss the performance of each individual model algorithm with various dimensionality reduction methods applied. Each model will be measured through the use of 3 metrics: **silhouette score**, measuring the overlap between multiple clusters, **ARI**, which measures the agreement between the clustering results and the ground-truth labels while correcting for chance and **NMI**, which measures how much information is shared between the predicted clusters and the true labels.

K-Means

The K-Means models were all created from the Sklearn implementation of K-Means. The results of the K-Means experiments can be seen in Figure 1

Metric	No Dim Red	PCA 90	PCA 95	Autoencoder	UMAP
Silhouette Score	0.1433	0.1730	0.1679	0.1564	0.5099
ARI	0.3750	0.3750	0.3489	0.2840	0.4873
NMI	0.5131	0.5131	0.5128	0.4847	0.6489
Time (s)	189	21.8	40.2	4.17	14

Table 1: K-Means clustering performance under different dimensionality reduction methods

The time included in the table only contains times for the algorithm to run and not the time required for dimensionality reduction. The time to perform the dimensionality reduction for each technique can be seen in Figure 4.
Overall K-Means shows quite strong performance on this dataset given how similar some items from different classes are. The silhouette score stays around 0.15 which shows some strong overlap with some areas of clusters being fairly strongly separated when compared to others.
The dimensionality reduction methods greatly reduce the runtime of the clustering algorithms and are very useful to apply. The runtimes of each algorithm are as expected with the algorithms that use a greater amount of variables taking longer to finish.
The ARI and NMI scores are consistently high across all methods, meaning the clustering successfully captures meaningful patterns in the data. The differences between scores show how the choice of representation affects clustering quality.
From the K-Means experiments, the UMAP dimensionality reduction achieves the best results due it having the largest ARI, NMI and silhouette score, showing it creates the most separated clusters from the data with high quality clusters. This can be further seen in Figure 19 which shows clear separation between some clusters while others are more closely grouped. It is also the second fastest running algorithm, when including the dimensionality reduction process, making it the clear best performer in the K-Means category.

Gasussian Mixture

Gaussian Mixture models were created using 10 componenets, a covariance type of full with k-means++ as the initial parameters. The results for these models can be seen in Figure 2

Metric	PCA 90	PCA 95	Autoencoder	UMAP
Silhouette Score	0.0640	0.0663	0.0798	0.4701
ARI	0.3181	0.3673	0.4293	0.4868
NMI	0.5144	0.5515	0.5864	0.6463
Time (s)	277	1406	165	9.55

Table 2: Gaussian Mixture clustering performance under different dimensionality reduction methods

Overall, the Gaussian Mixture models show moderate but consistent clustering performance, with results highly dependent on the dimensionality reduction method used. Compared to K-Means, GMMs demonstrate greater sensitivity to both feature dimensionality and runtime, particularly under higher-dimensional PCA representations.
The silhouette scores for PCA-based representations remain low ($\approx$0.06–0.07), indicating substantial cluster overlap when assuming Gaussian components in linear subspaces. The Autoencoder representation slightly improves geometric separation, while UMAP produces a substantially higher silhouette score, suggesting that nonlinear manifold learning enables more compact and well-separated Gaussian

components.
The ARI and NMI scores follow a clear trend: performance improves as representations move from linear PCA spaces to learned or nonlinear embeddings. UMAP achieves the highest ARI and NMI values, closely matching the best K-Means performance, while the Autoencoder representation also shows a notable improvement over PCA. This indicates that GMMs benefit from representations that better disentangle class structure, rather than relying on raw variance preservation.
Runtime varies dramatically across representations. PCA-based GMMs are computationally expensive, especially at higher retained variance, due to full covariance estimation in high-dimensional spaces. In contrast, UMAP and Autoencoder representations significantly reduce computational cost, with UMAP achieving both strong clustering performance and the fastest runtime among GMM experiments. This makes UMAP the most effective representation for Gaussian Mixtures in this study.

Agglomerative Clustering

Due to the high memory usage of Agglomerative Clustering, a subset of 20,000 samples was used, selected through stratified sampling to maintain the original class distributions. These models were ran to create 10 clusters using the ward linkage. The runtimes for the dimensionality reduction on the subsets can be found in Figure 5 The results for the Agglomerative Clustering experiments can be seen in Figure 3.

Metric	PCA 90	PCA 95	Autoencoder	UMAP
Silhouette Score	0.1483	0.1343	0.1131	0.3086
ARI	0.4311	0.4311	0.3185	0.4048
NMI	0.5928	0.5928	0.5271	0.5508
Time (s)	31.2	30.3	26.9	166

Table 3: Agglomerative clustering performance under different dimensionality reduction methods

Agglomerative Clustering shows robust and stable performance across most representations, particularly in terms of ARI and NMI, despite being evaluated on a reduced dataset due to memory constraints. This suggests that hierarchical methods are capable of capturing a meaningful structure even with fewer samples.
Silhouette scores indicate moderate geometric separation for PCA-based representations, with PCA-90 outperforming PCA-95 and the Autoencoder. UMAP again yields the highest silhouette score, although the improvement is less pronounced than in the GMM results, reflecting the method's tendency to distort global distances despite improving local separation.
The ARI and NMI scores are strongest for PCA-based representations, particularly PCA-90 and PCA-95, which achieve the highest and most stable alignment with ground truth labels. This suggests that agglomerative clustering with Ward linkage benefits from distance-preserving, linear representations, where hierarchical merges more closely reflect class structure. The Autoencoder representation underperforms in this setting, indicating that its latent space may not be well suited for hierarchical distance-based clustering.
While UMAP improves silhouette score, its ARI and NMI are slightly lower than those achieved with PCA, highlighting a trade-off between geometric separation and semantic alignment for hierarchical methods. Additionally, UMAP incurs a significantly higher runtime for agglomerative clustering, making it less practical in this context.

Key Findings, Insights and Next Steps

Overall K-Means with UMAP dimensionality reduction performs best for this task and produces clusters that are high quality and well separated. This experiment shows that there are still some problems with clustering these fashion images as the best sillouhette score that is achieved is around 0.5. This issue stems from the low quality of the images which limits the models ability to accurately distinguish between similar items such as t-shirts and shirts.

Further work could include trying to create less clusters with more broad structures such as having all shoes in a single category and all trousers in another. Another improvement that could be made is to use higher quality images. This will increase computational expensiveness but can result in more accurate models that create more distinct clusters.
This work shows that UMAP can achieve very good results when compared to autoencoders and PCA for low quality image data specifically. It also highlights the struggles that Gaussian Mixtures face when dealing with image data as that is the model that performed the worst overall.

Appendix

This section contains all cluster visualisations and cluster sample grids for every test as well as time taken for each dimensionality reduction technique.

Cluster Visualisations

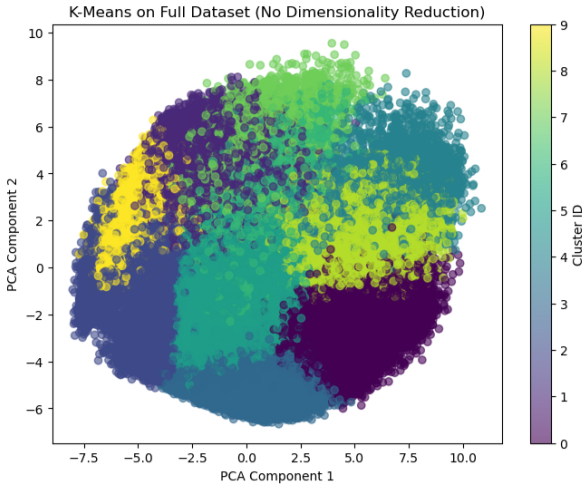

Figure 2: K-means clustering with no dimensionality reduction (full feature space).

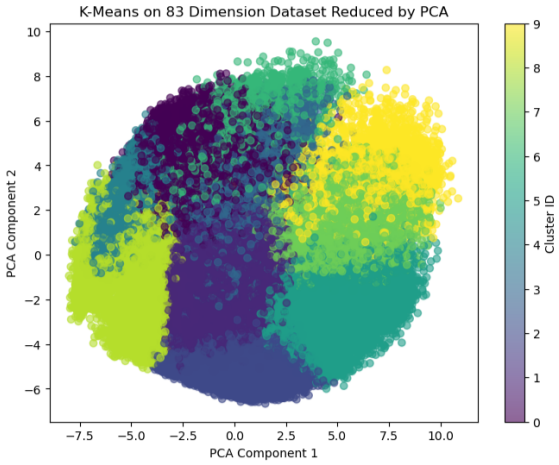

Figure 3: *
K-means with PCA reduced to 90% explained variance.

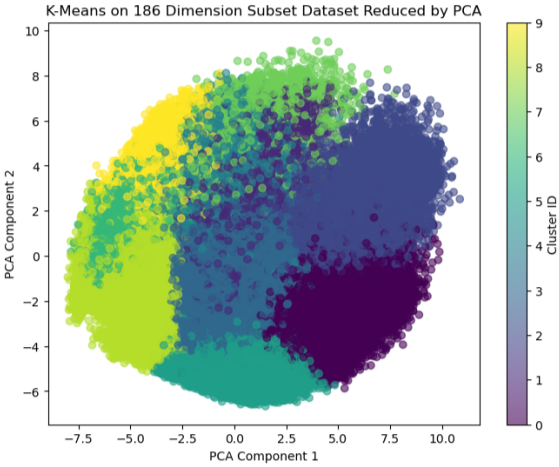

Figure 4: *
K-means with PCA reduced to 95% explained variance.

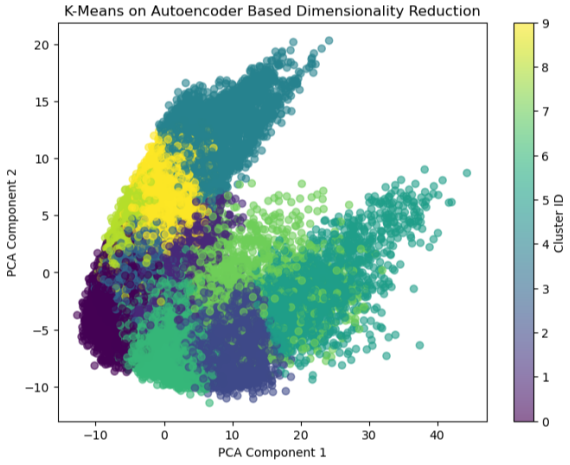

Figure 5: *
K-means with an autoencoder reduced to 32 latent dimensions.

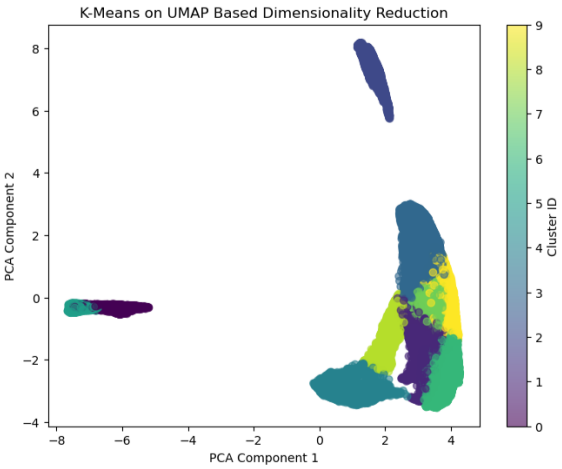

Figure 6: *
K-means with UMAP dimensionality reduction.

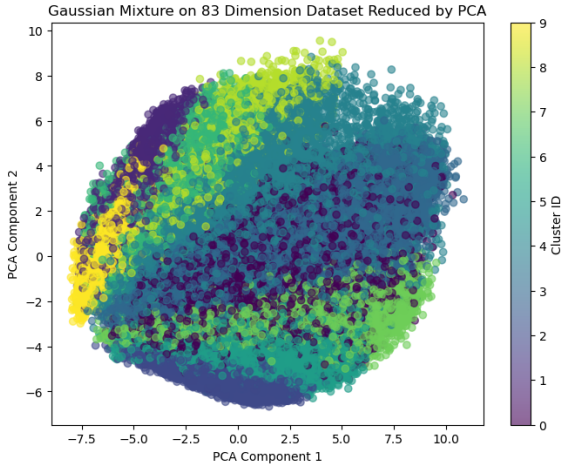

Figure 7: *
Gaussian Mixture Model with PCA reduced to 90% explained variance.

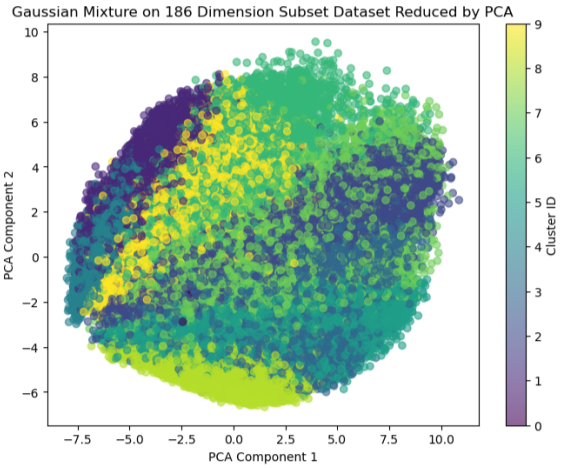

Figure 8: *
Gaussian Mixture Model with PCA reduced to 95% explained variance.

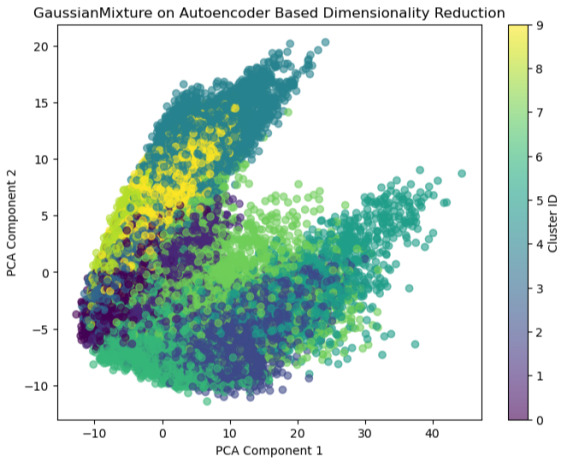

Figure 9: *
Gaussian Mixture Model with an autoencoder reduced to 32 latent dimensions.

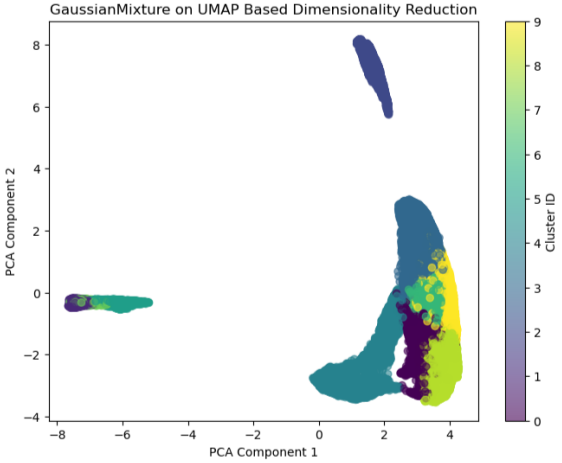

Figure 10: *
Gaussian Mixture Model with UMAP dimensionality reduction.

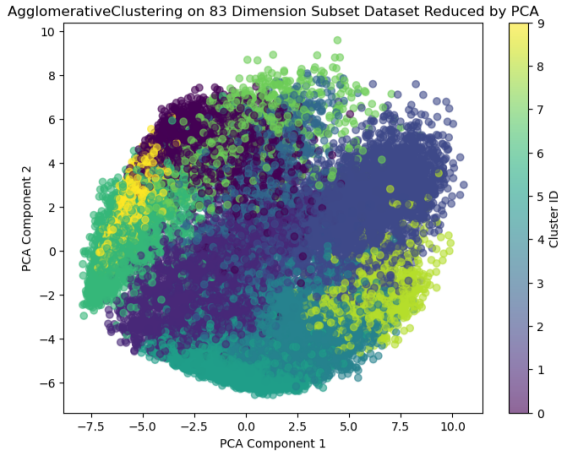

Figure 11: *
Agglomerative clustering with PCA reduced to 90% explained variance.

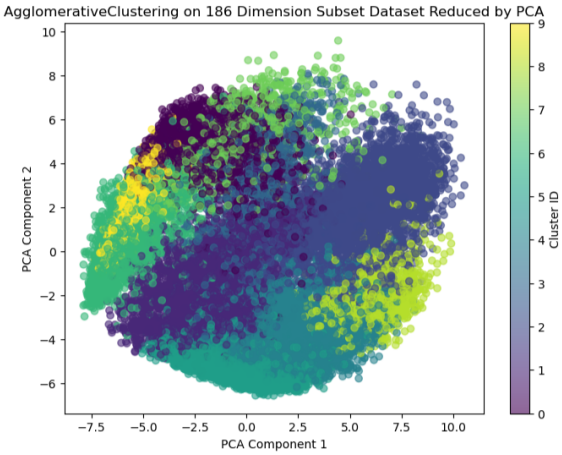

Figure 12: *
Agglomerative clustering with PCA reduced to 95% explained variance.

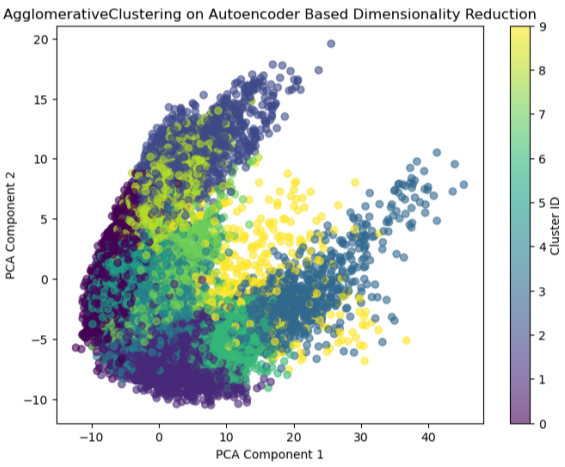

Figure 13: *
Agglomerative clustering with an autoencoder reduced to 32 latent dimensions.

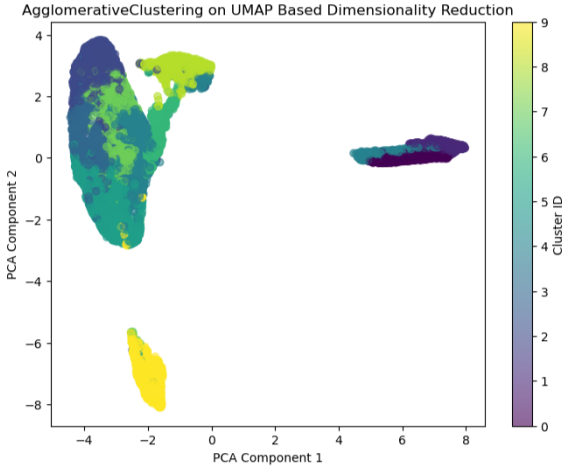

Figure 14: *
Agglomerative clustering with UMAP dimensionality reduction.

Cluster Sample Images

Figure 15: K-means clustering with no dimensionality reduction (full feature space).

Figure 16: *
K-means with PCA reduced to 90% explained variance.

Figure 17: *
K-means with PCA reduced to 95% explained variance.

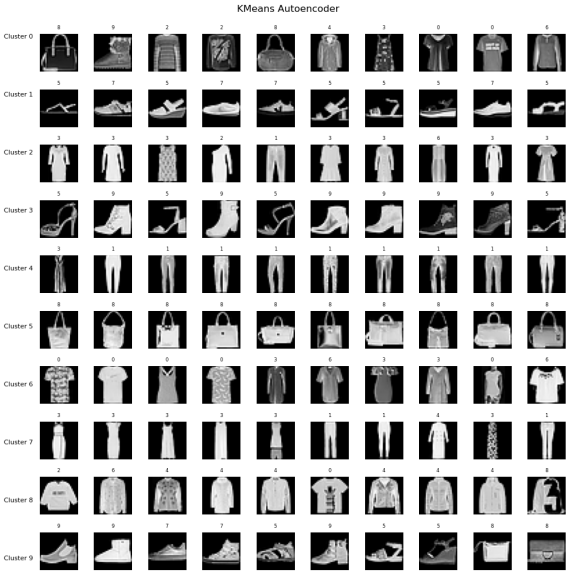

Figure 18: *
K-means with an autoencoder reduced to 32 latent dimensions.

Figure 19: *
K-means with UMAP dimensionality reduction.

Figure 20: *
Gaussian Mixture Model with PCA reduced to 90% explained variance.

Figure 21: *
Gaussian Mixture Model with PCA reduced to 95% explained variance.

Figure 22: *
Gaussian Mixture Model with an autoencoder reduced to 32 latent dimensions.

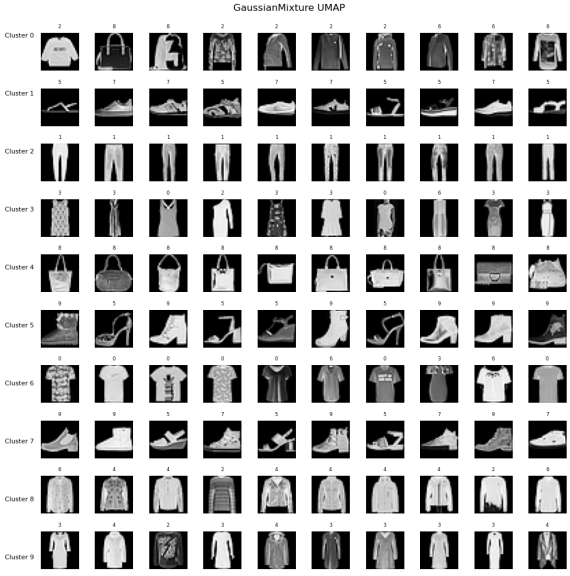

Figure 23: *
Gaussian Mixture Model with UMAP dimensionality reduction.

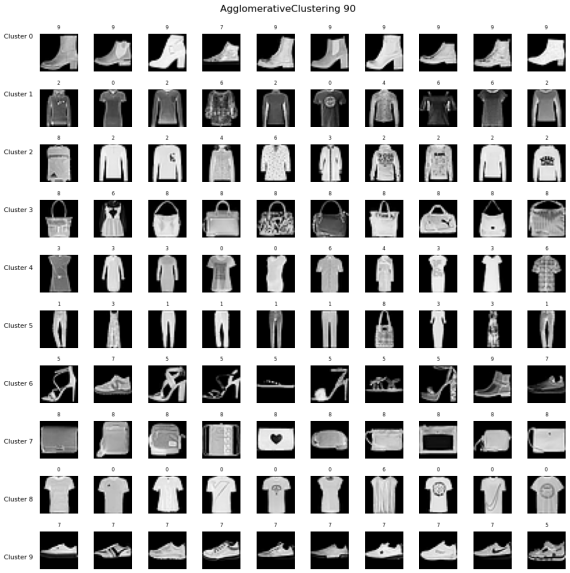

Figure 24: *
Agglomerative clustering with PCA reduced to 90% explained variance.

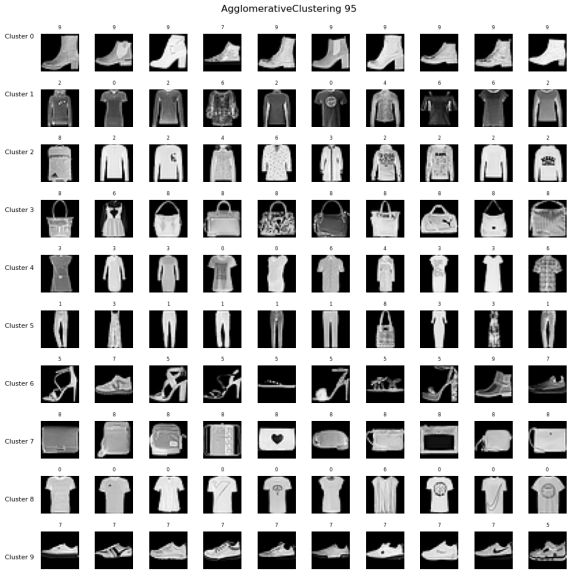

Figure 25: *
Agglomerative clustering with PCA reduced to 95% explained variance.

Figure 26: *
Agglomerative clustering with an autoencoder reduced to 32 latent dimensions.

Figure 27: *
Agglomerative clustering with UMAP dimensionality reduction.

Dimensionality Reduction

PCA was used to reduced to 83 and 186 components to respectively show 90% and 95% of explained variance. Autoencoders reduced to 32 latent variables and UMAP was reduced to 32 components.

Dimensionality Reduction Method	**Time Taken (s)**
PCA 90%	5.25
PCA 95%	5.83
Autoencoder	342
UMAP	170

Table 4: Time for Dimensionality Reduction on Full Set

Dimensionality Reduction Method	**Time Taken (s)**
PCA 90%	2.67
PCA 95%	2.42
Autoencoder	95
UMAP	39.4

Table 5: Time for Dimensionality Reduction on Subset