# Clustering of Fashion MNIST Data Using Autoencoder and Kmeans

Fardeen Rahman

20221032

## 1. Introduction

In this project, we aim to apply unsupervised learning techniques on the Fashion MNIST dataset using a neural network-based approach. We utilize an Autoencoder to extract compressed representations of the image data and then apply KMeans clustering on the learned features. The goal is to determine how well unsupervised learning can cluster similar fashion items and evaluate the quality of clustering using various metrics.

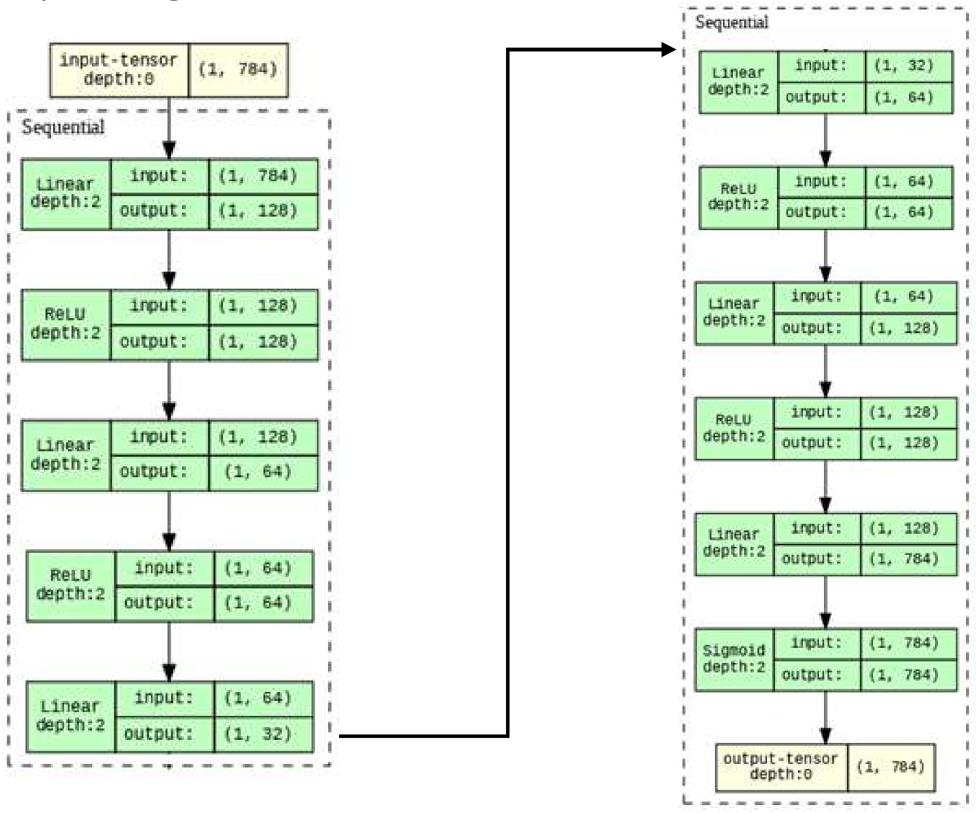
## 2. Dataset Description & Analysis

The dataset used in this project is the Fashion MNIST dataset, obtained from the Hugging Face Datasets library. It consists of 60,000 grayscale images for training and 10,000 images for testing, each with a resolution of 28×28 pixels. The dataset includes images from 10 categories of clothing items such as T-shirts, trousers, pullovers, dresses, coats, sandals, and sneakers, each represented by an integer label from 0 to 9.

The images were stored in PNG byte format, which were decoded using the PIL (Python Imaging Library). Each image was then flattened into a 784-dimensional vector (28 × 28 = 784 pixels) and normalized to the range [0, 1] for compatibility with the neural network model. The pixel intensity values were divided by 255.0 to scale them, a standard preprocessing step in deep learning.

This dataset provides a challenging benchmark for unsupervised learning as it contains visually similar items (e.g., shirts and T-shirts) that may be difficult to differentiate without supervised labels. Thus, it is a suitable candidate for evaluating the ability of learned latent representations to facilitate meaningful clustering.

## 3. Layer Diagram of Model Architecture



## 4. Hyperparameter Optimization and Tuning

The model was trained using the following fixed hyperparameters:

- Optimizer: Adam
- Loss Function: Mean Squared Error (MSE)
- Learning rate = 0.001
- Batch Size: 128
- Epochs: 20
- Latent Space Dimension: 64
- Activation Functions: ReLU for hidden layers, Sigmoid for output

No grid search or automated hyperparameter tuning techniques were employed. Despite this, the model demonstrated stable and efficient learning behavior. The selected parameters were chosen based on standard practices for training autoencoders on image datasets like Fashion-MNIST. For future work, tuning parameters such as the latent dimension, learning rate, or number of hidden layers could help optimize reconstruction quality and training efficiency.

## 5. Model Parameter Counting

Let $D$ = 784 be the input dimension (e.g., number of features after dropping labels).

| Layer | Parameters (without bias) |
|---|---|
| Input → Dense (128) | 784 × 128 = 100,352 |
| Dense (128) → Dense (64) | 128 × 64 = 8,192 |
| Dense (64) → Dense (32) (Latent) | 64 × 32 = 2,048 |
| Dense (32) → Dense (64) | 32 × 64 = 2,048 |
| Dense (64) → Dense (128) | 64 × 128 = 8,192 |
| Dense (128) → Output (784) | 784 × 128 = 100,352 |

**Total Parameters (without bias):** 100352 + 8192 + 2048 + 2048 + 8192 + 100352

=~ 221184

Biases (one per neuron) would add around $\sim$ 1,200 extra parameters.

## 6. Clustering Methodology and Comparison

In this project, we employed **KMeans clustering** on compressed image representations generated by an autoencoder. The autoencoder reduced the original 784-dimensional input (28×28 grayscale images) to a **64-dimensional latent space**, which served as the feature space for clustering. We chose **k=10** as the number of clusters to align with the 10 true fashion item categories in the Fashion MNIST dataset. This choice enabled an interpretable comparison between unsupervised clustering output and known labels.

We also used **t-SNE (t-Distributed Stochastic Neighbor Embedding)** for visualizing the 64-dimensional encoded vectors in 2D space. The t-SNE plot revealed visibly distinct cluster boundaries, suggesting that the learned latent representations preserve key structures in the data space. Each color in the t-SNE corresponds to a KMeans cluster, and the separation between color groups validates the clustering capability of the autoencoder's encoded features.

Although other techniques like MiniSom or DBSCAN could be used as alternative clustering strategies, KMeans was chosen for its simplicity, scalability, and effectiveness with compact feature spaces like ours.

## 7. Evaluation

To assess how well the clustering captured the underlying data structure, we evaluated performance using **extrinsic clustering metrics** that compare predicted clusters with ground-truth labels. Note that ground-truth labels were not used during training or clustering—they were only used for validation.

### 7.1 Metrics Used for Evaluation:

| **Metric** | **Value** | **Interpretation** |
|---|---|---|
| **Purity Score** | 0.5541 | Measures how often the majority label in a cluster matc true labels. |
| **Silhouette Score** | 0.2091 | Indicates cluster cohesion and separation; values closer t are better. |
| **Adjusted Rand Index (AI** | 0.3444 | Evaluates similarity between clustering and true lab adjusted for chance. |
| **Normalized Mutual I (NMI)** | 0.5217 | Measures information shared between predicted and t labels, normalized. |

### 7.2 Analysis:

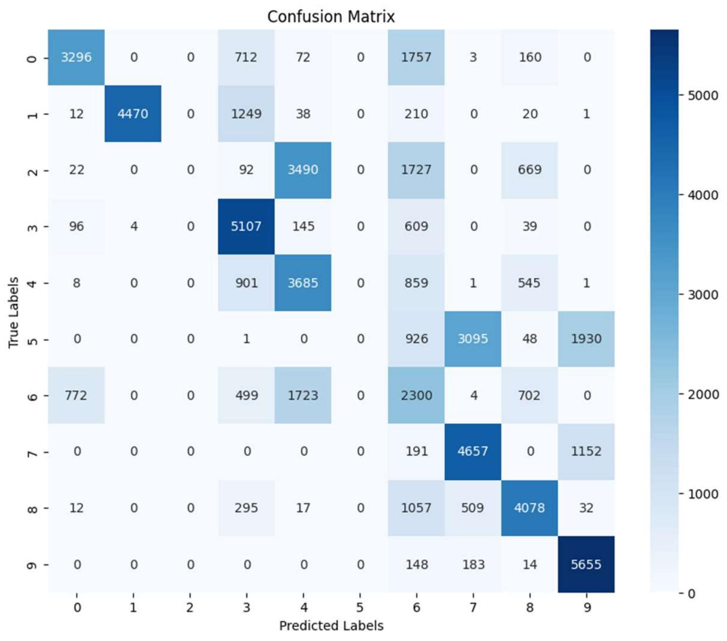
- **Purity Score** of 0.5541 reflects moderate alignment between the cluster assignments and the actual labels. While this shows that some clusters strongly reflect true categories, there is still overlap and ambiguity between classes such as shirt, coat, and pullover.
- **Silhouette Score** of 0.2091, though modest, indicates that the clusters are somewhat cohesive but not very well separated. This is common when high-dimensional data (even after compression) still exhibits class overlap.
- **ARI** of 0.3444 shows moderate agreement between predicted and true class groupings, which is expected given the unsupervised nature of the task.

- **NMI** of 0.5217 reflects that a decent portion of the class information is preserved in the cluster assignments. This supports the claim that the autoencoder effectively compresses relevant features.

## 7.3 Confusion Matrix Interpretation:

The confusion matrix highlights which fashion categories were most accurately clustered and which were frequently confused. For example:

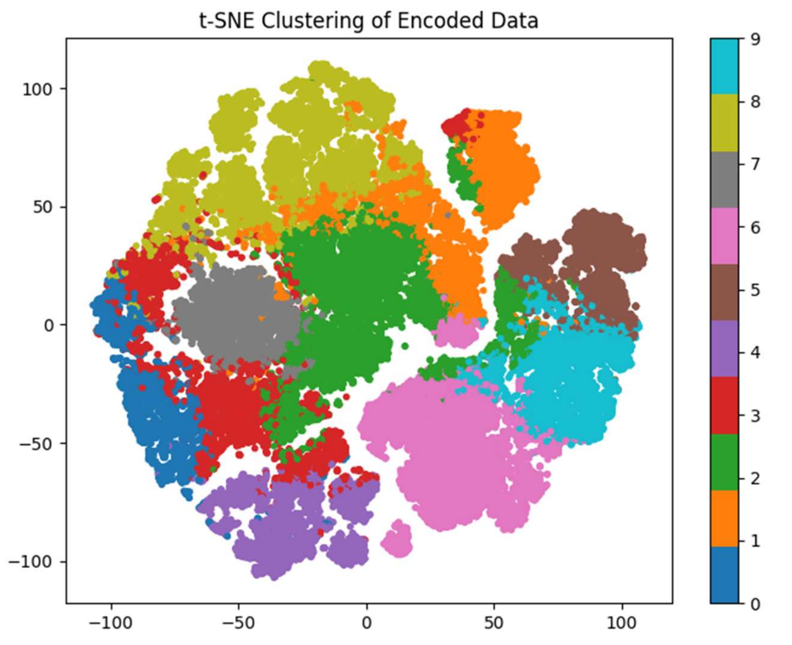
- Classes like **T-shirt/top**, **Sandal**, and **Sneaker** showed relatively strong diagonal entries, indicating successful clustering.
- However, **Shirt**, **Pullover**, and **Coat** had considerable off-diagonal overlap, indicating difficulty in distinguishing between visually similar clothing items.



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 3296 | 0 | 0 | 712 | 72 | 0 | 1757 | 3 | 160 | 0 |
| 1 | 12 | 4470 | 0 | 1249 | 38 | 0 | 210 | 0 | 20 | 1 |
| 2 | 22 | 0 | 0 | 92 | 3490 | 0 | 1727 | 0 | 669 | 0 |
| 3 | 96 | 4 | 0 | 5107 | 145 | 0 | 609 | 0 | 39 | 0 |
| 4 | 8 | 0 | 0 | 901 | 3685 | 0 | 859 | 1 | 545 | 1 |
| 5 | 0 | 0 | 0 | 1 | 0 | 0 | 926 | 3095 | 48 | 1930 |
| 6 | 772 | 0 | 0 | 499 | 1723 | 0 | 2300 | 4 | 702 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 191 | 4657 | 0 | 1152 |
| 8 | 12 | 0 | 0 | 295 | 17 | 0 | 1057 | 509 | 4078 | 32 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 148 | 183 | 14 | 5655 |

## 7.4 t-SNE Interpretation:

The t-SNE visualization further supports this. Clusters with minimal overlap corresponded to clearly distinguishable clothing items (e.g., boots vs. shirts), while overlapping regions often represented semantically or visually similar categories.



## 8. Challenges and Limitations

One of the main challenges was the limited clustering performance, as reflected by moderate scores in Purity (0.5541), ARI (0.3444), and NMI (0.5217). This suggests that the latent features learned by the Autoencoder did not fully separate the underlying classes. Additionally, the model lacked regularization techniques such as dropout or batch normalization, which may have impacted generalization and training stability. Visualization using t-SNE, while helpful, posed interpretability issues due to its stochastic nature. Moreover, computational constraints were encountered during training and graph rendering, especially given the dimensionality of the input and the model complexity. These challenges point to opportunities for future work, such as using more advanced Autoencoders, adding regularization, exploring alternative visualizations like UMAP, and optimizing training through better hardware or code efficiency.

## 9. Conclusion

In this project, we implemented an unsupervised learning approach using an Autoencoder model on the Fashion MNIST dataset to extract compressed latent features, followed by KMeans clustering for grouping similar fashion items. The Autoencoder successfully reduced the dimensionality of the data while preserving essential features, and the resulting clusters were evaluated using multiple metrics including Purity, Silhouette Score, ARI, and NMI. While the clustering showed moderate alignment with true labels, visualization through t-SNE confirmed clear separations among several classes, demonstrating the Autoencoder's effectiveness in feature learning. Despite some limitations in clustering performance and the absence of regularization techniques, the results validate the potential of combining neural representation learning with classical clustering algorithms. Future improvements can include architectural tuning, regularization, and more sophisticated clustering strategies to enhance both accuracy and generalizability.

## 10. Bibliography

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. arXiv preprint arXiv:1708.07747.

PyTorch Documentation. https://pytorch.org/docs/stable/index.html

Scikit-learn: Machine Learning in Python. https://scikit-learn.org/stable/

t-SNE Explained. https://distill.pub/2016/misread-tsne/

Hugging Face Datasets Documentation. https://huggingface.co/docs/datasets/

Kurtutan, M. (2023). torchview: Graph Visualization for PyTorch. https://github.com/mert-kurttutan/torchview

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980.