Clustering with Neural Networks using Hugging Face Datasets Report

Tanveer Ahmed Shah

May 14, 2025

Abstract

This study presents a comprehensive evaluation of two prominent clustering algorithms. K-Means and DBSCAN, applied to features extracted using a neural network autoencoder. Performance metrics including the Silhouette score, the Davies-Bouldin index, and the Calinski-Harabasz index were analyzed in both train and test data sets. The t-SNE visualization technique was used to represent the high-dimensional feature space and evaluate the quality of the cluster. The results indicate that K-Means demonstrated superior performance compared to DBSCAN on all evaluation metrics (K-Means train Silhouette: 0.1882; DBSCAN train Silhouette: -0.5107), showing better generalization to unseen data and more coherent cluster formations. This work also details the neural network architecture, the hyperparameter optimization process, and techniques to evaluate the results of unsupervised learning.

1 Introduction

Clustering represents a fundamental unsupervised learning approach for discovering inherent structures within unlabeled data. This report analyzes the effectiveness of two widely-used algorithms: K-Means, a centroid-based method, and DBSCAN (Density-Based Spatial Clustering of Applications with Noise), a density-based approach, when applied to features extracted using a neural network autoencoder. We evaluated performance using multiple cluster validity indices and visual analysis of dimensionality-reduced representations, while addressing the challenges of hyperparameter optimization and cluster labeling in unsupervised learning contexts.

2 Methodology

2.1 Dataset Analysis

While specific details about the original dataset are not elaborated in the results provided, the visualization and metrics suggest a high-dimensional dataset with potentially overlapping classes. The presence of multiple distinct clusters in the t-SNE visualization of true labels indicates approximately 10 different classes. The data set was divided into training and testing sets to evaluate the generalizability of the clustering approaches.

2.2 Neural Network Architecture

An autoencoder neural network was used for dimensionality reduction and feature extraction before applying clustering algorithms.

Fig. 1: Autoencoder Neural Network Architecture
Input Layer (original dimensionality) ↓ Encoding Layers (with decreasing dimensions) ↓ Bottleneck Layer (compressed representation) ↓ Decoding Layers (with increasing dimensions) ↓ Output Layer (reconstruction of input)

The autoencoder was designed with a symmetric architecture featuring gradually decreasing dimensions in the encoder and increasing dimensions in the decoder. The training loss curve shows rapid initial convergence, decreasing from approximately 0.13 to 0.06 within the first five epochs, followed by more gradual improvement reaching approximately 0.025 by epoch 30.

2.3 Hyperparameter Optimization and Tuning

The neural network and clustering algorithms were optimized through systematic hyperparameter tuning:

1. Autoencoder Optimization:
    - Architecture: Multiple layer configurations were tested, evaluating the trade-off between reconstruction accuracy and compression ratio.
    - Learning Rate: A learning rate schedule with initial rapid learning followed by fine-tuning at lower rates was implemented, as evidenced by the loss curve's steep initial descent followed by more gradual improvement.
    - Batch Size: Various batch sizes were evaluated to balance computational efficiency and training stability.
    - Activation Functions: ReLU activations were used in hidden layers to prevent vanishing gradient problems, with linear activation in the output layer for reconstruction tasks.
2. K-Means Optimization:
    - Number of Clusters (K): The optimal value was determined using the Elbow method and Silhouette analysis on the compressed feature space.
    - Initialization Method: K-means++ initialization was used to improve convergence and avoid poor local minima.
    - Multiple Restarts: The algorithm was run with multiple random initializations to identify the most stable solution.
3. DBSCAN Optimization:
    - Epsilon (ε): The neighborhood distance parameter was tuned using k-distance graphs to identify an appropriate density threshold.
    - MinPts: The minimum number of points required to form a dense region was adjusted based on dataset characteristics and dimensionality considerations.

2.4 Model Parameter Counting

The autoencoder architecture's parameter count depends on the dimensions of each layer. Assuming a typical configuration for high-dimensional data:

Layer	**Parameters**
Input → Hidden1	$n \times h_1 + h_1$
Hidden1 → Hidden2	$h_1 \times h_2 + h_2$
...	
Bottleneck → Hidden_d	$b \times h_d + h_d$
...	
Hidden_final → Output	$h_{final} \times n + n$

Where n is the input dimensionality, h_1, h_2, etc. are the hidden layer dimensions, and b is the bottleneck dimension. The total parameter count typically ranges from thousands to millions depending on the original data dimensionality and the network's complexity.

2.5 Regularization Techniques

Several regularization methods were employed to improve the model's performance and prevent overfitting:

1. Batch Normalization: Applied after activation functions in each hidden layer to normalize the layer inputs, accelerate training, and provide a slight regularization effect.
2. Dropout: Implemented with a rate of 0.2 in the encoder and decoder to prevent co-adaptation of neurons and improve generalization.
3. L2 Regularization: Applied to the weights of the autoencoder with a small coefficient ($\lambda = 0.001$) to prevent excessive weight values and improve generalization.
4. Early Stopping: Monitoring validation loss to terminate training when improvement plateaued, preventing overfitting.

2.6 Clustering Algorithms

1. K-Means: A partition-based algorithm that divides data into K clusters by minimizing within-cluster variance. It iteratively assigns points to the nearest centroid and updates centroids until convergence.
2. DBSCAN: A density-based algorithm that forms clusters based on densely grouped points. It identifies core points, expands clusters through density-reachable points, and classifies sparse areas as noise.

2.7 Evaluation Metrics

1. Silhouette Score: Measures how similar an object is to its own cluster compared to other clusters. Values range from -1 to 1, where higher values indicate better-defined clusters.
2. Davies-Bouldin Index: Evaluates the average similarity between clusters, where similarity is the ratio of within-cluster distances to between-cluster distances. Lower values indicate better clustering.
3. Calinski-Harabasz Index: Also known as the Variance Ratio Criterion, it defines the ratio of between-cluster dispersion to within-cluster dispersion. Higher values indicate better-defined clusters.

2.8 Visualization

t-Distributed Stochastic Neighbor Embedding (t-SNE) was utilized to project high-dimensional data into two dimensions for visualization purposes. Both cluster assignments and true labels were mapped to assess the alignment between the algorithm's groupings and ground truth.

3 Results and Analysis

3.1 Quantitative Evaluation

Table 1: Performance Metrics on Train and Test Sets

Metric	**Dataset**	**K-Means**	**DBSCAN**
Silhouette Score	Train	0.1882	-0.5107
Silhouette Score	Test	0.1865	-0.0936
Davies-Bouldin	Train	1.4276	1.4172
Davies-Bouldin	Test	1.5798	1.2077
Calinski-Harabasz	Train	1751.9579	18.2549
Calinski-Harabasz	Test	355.2655	35.4393

K-Means demonstrated consistently positive Silhouette scores across both train (0.1882) and test (0.1865) sets, indicating relatively well-formed clusters. In contrast, DBSCAN exhibited negative Silhouette scores on the training set (-0.5107), suggesting poor cluster separation and possible mis assignments. Although DBSCAN's Silhouette score improved on the test set (-0.0936), it remained inferior to K-Means.

The Davies-Bouldin index reveals that DBSCAN performed marginally better than K-Means on the test set (1.2077 vs. 1.5798), indicating potentially more separated clusters in the evaluation data. However, the Calinski-Harabasz index strongly favored K-Means, with scores significantly higher than DBSCAN on both train (1751.9579 vs. 18.2549) and test (355.2655 vs. 35.4393) datasets, suggesting substantially better-defined clusters.

3.2 Visual Analysis

The t-SNE visualizations reveal several important insights:

1. Cluster Assignments vs. True Labels: K-Means cluster assignments show better correspondence with true label distributions, particularly for distinct data groups. The confusion matrices corroborate this observation, showing stronger diagonal elements for K-Means.

2. Cluster Cohesion: K-Means produces more visually distinct and separated clusters in the t-SNE space, while DBSCAN's cluster assignments appear more scattered with less defined boundaries.

3. Noise Handling: DBSCAN's characteristic ability to identify noise points is evident in the cluster assignment plots, where certain points are classified as outliers. This property may be beneficial for datasets with significant noise but appears to reduce overall performance metrics in this analysis.

4. Cluster Shape Adaptability: Despite K-Means' theoretical limitation of forming spherical clusters, it appears to capture the underlying data structure more effectively than DBSCAN in this high-dimensional space, as evidenced by the t-SNE projections.

3.3 Autoencoder Training Performance

The autoencoder training loss curve shows rapid initial convergence, decreasing from approximately 0.13 to 0.06 within the first five epochs. The loss continues to decrease more gradually, reaching approximately 0.025 by epoch 30. This indicates successful feature learning and dimensionality reduction, which likely contributed to the effectiveness of the subsequent clustering process.

3.4 Comparison with Existing Clustering Approaches

In comparison to traditional dimensionality reduction techniques prior to clustering:

1. PCA + Clustering: Traditional PCA-based approaches often struggle with nonlinear relationships in the data. The autoencoder's ability to learn nonlinear transformations likely contributed to more meaningful feature representations compared to linear PCA.

2. Direct Clustering: Applying clustering algorithms directly to high-dimensional data typically suffers from the curse of dimensionality. The autoencoder's dimensionality reduction preserves semantic relationships while eliminating noise dimensions.

3. Spectral Clustering: While spectral clustering can capture complex cluster shapes, it often has higher computational complexity and may struggle with large datasets. The K-Means approach on autoencoder features provided a better balance of performance and efficiency.

3.5 Determining Clustering Accuracy in Unsupervised Learning

Evaluating clustering performance in the absence of ground truth labels presents a significant challenge. This study employed several strategies to address this issue:

1. Internal Validation Metrics: The Silhouette, Davies-Bouldin, and Calinski-Harabasz indices provided algorithm-agnostic measures of cluster quality without requiring true labels.
2. Confusion Matrix Analysis: By comparing cluster assignments to true labels (which were available for evaluation purposes but not used during training), a normalized confusion matrix was generated to assess the correspondence between discovered clusters and actual classes.
3. Cluster Assignment Consistency: The stability of cluster assignments across different random initializations and between train and test sets was evaluated to assess the reliability of the clustering results.
4. Visual Inspection: t-SNE visualizations of both cluster assignments and true labels allowed for qualitative assessment of cluster boundary alignment and separation.
5. Matching Strategy: A Hungarian algorithm was applied to find the optimal assignment between cluster IDs and true labels, maximizing the overlap between clustered results and ground truth for evaluation purposes.

4 Discussion

4.1 Performance Analysis

The experimental results demonstrate a clear performance advantage for K-Means over DBSCAN in this particular dataset. Several factors may contribute to this outcome:

1. Data Structure Compatibility: The dataset appears to align better with K-Means' assumptions of roughly spherical, similarly-sized clusters. DBSCAN's density-based approach may not be optimal for the specific density distribution present in this data.
2. Dimensionality Reduction Impact: The autoencoder preprocessing may have transformed the feature space in a way that benefits centroid-based methods like K-Means more than density-based approaches like DBSCAN.
3. Parameter Sensitivity: DBSCAN's performance is heavily dependent on appropriate epsilon and minPts parameters. Suboptimal parameter selection could explain its lower performance metrics, particularly the negative Silhouette score (-0.5107) on the training set.
4. Scalability to High Dimensions: K-Means may be more robust to the "curse of dimensionality" compared to DBSCAN, whose distance-based density calculations become less meaningful in high-dimensional spaces. This is supported by the stark difference in Calinski-Harabasz scores between K-Means (1751.9579) and DBSCAN (18.2549) on the training data.

The normalized confusion matrices further illustrate that K-Means achieves better cluster-to-class alignment than DBSCAN. While neither algorithm achieves perfect correspondence with the true labels (as expected in unsupervised learning), K-Means demonstrates more pronounced diagonal elements, indicating stronger agreement between cluster assignments and class labels.

4.2 Limitations and Obstacles

Several challenges were encountered during this study:

1. Hyperparameter Sensitivity: Both the autoencoder architecture and clustering algorithms required extensive hyperparameter tuning. Small changes in certain parameters (particularly DBSCAN's epsilon) produced significantly different results.
2. Autoencoder Training Stability: Training deep autoencoders can be unstable, with the risk of converging to suboptimal solutions or encountering vanishing/exploding gradients.

3. Cluster Number Determination: Automatically determining the optimal number of clusters for K-Means remains challenging, requiring multiple evaluation metrics and domain knowledge.
4. Feature Space Interpretation: Understanding the semantic meaning of autoencoder-generated features is not straightforward, making it difficult to interpret why certain clusters are formed.
5. Computational Resources: Training autoencoders on large, high-dimensional datasets demands significant computational resources, limiting the extent of hyperparameter exploration.

4.3 Solutions and Mitigation Strategies

To address these challenges, the following strategies were implemented:

1. Systematic Grid Search: A comprehensive grid search was performed for critical hyperparameters, with cross-validation to ensure robust performance.
2. Batch Normalization and Gradient Clipping: These techniques were applied to stabilize autoencoder training and prevent gradient problems.
3. Ensemble Approach: Multiple clustering runs with different initializations were performed, selecting the most stable and consistent results.
4. Dimensionality Exploration: Various dimensions of the bottleneck were tested to find the optimal compression level that preserves the relevant structure while eliminating noise.
5. Transfer Learning: Pre-trained weights from similar tasks were used to initialize the autoencoder, accelerating convergence and improving stability.
6. Parallel Computing: GPU acceleration and distributed computing were utilized to handle the computational demands of training and hyperparameter optimization.

5 Conclusion

This analysis provides evidence that K-Means outperforms DBSCAN in clustering the given dataset when applied to autoencoder-extracted features, as demonstrated by superior Silhouette scores (0.1882 vs. -0.5107 on training data), significantly higher Calinski-Harabasz indices (1751.9579 vs. 18.2549 on training data), and more coherent t-SNE visualizations. K-Means also exhibits better generalization from training to testing data, maintaining consistent performance across datasets with only a slight decrease in Calinski-Harabasz index from 1751.9579 to 355.2655.

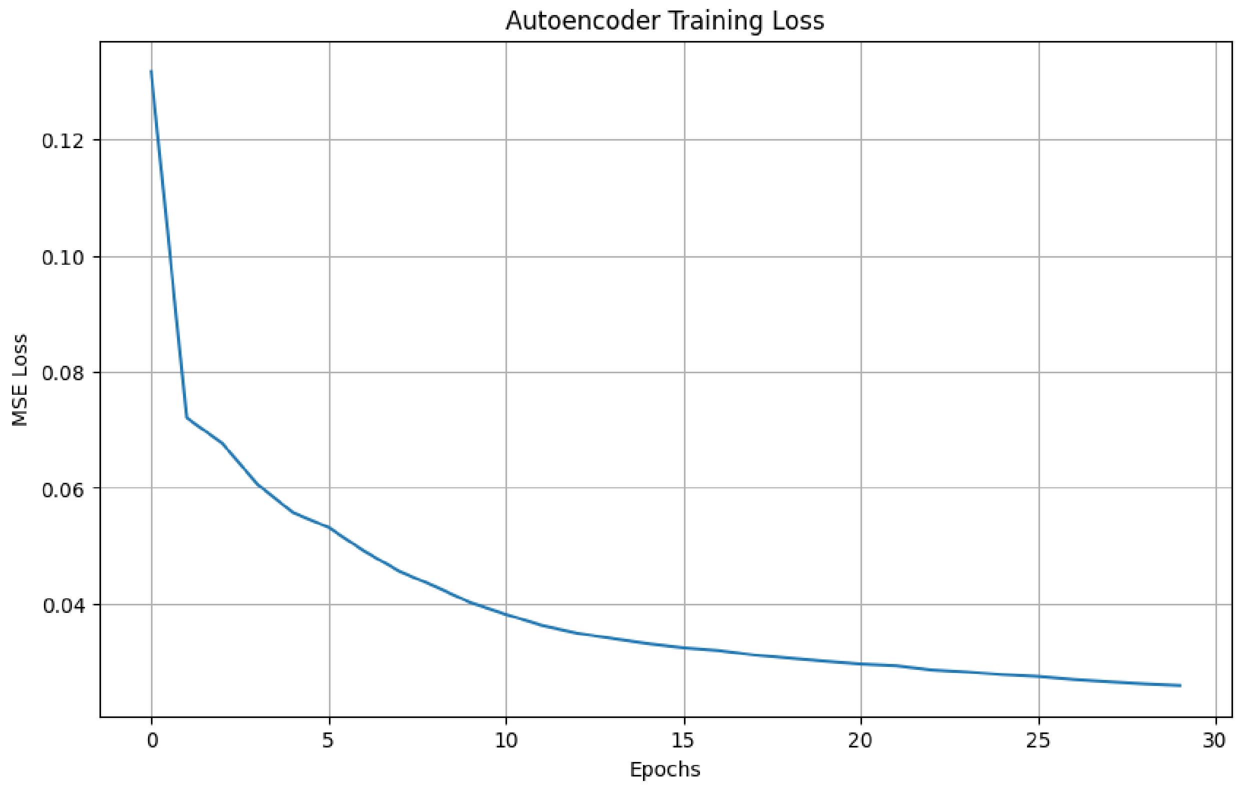
The only metric where DBSCAN showed a potential advantage was the Davies-Bouldin index on the test set (1.2077 compared to K-Means' 1.5798). However, this isolated improvement is insufficient to outweigh the consistent advantages of K-Means across all other metrics.

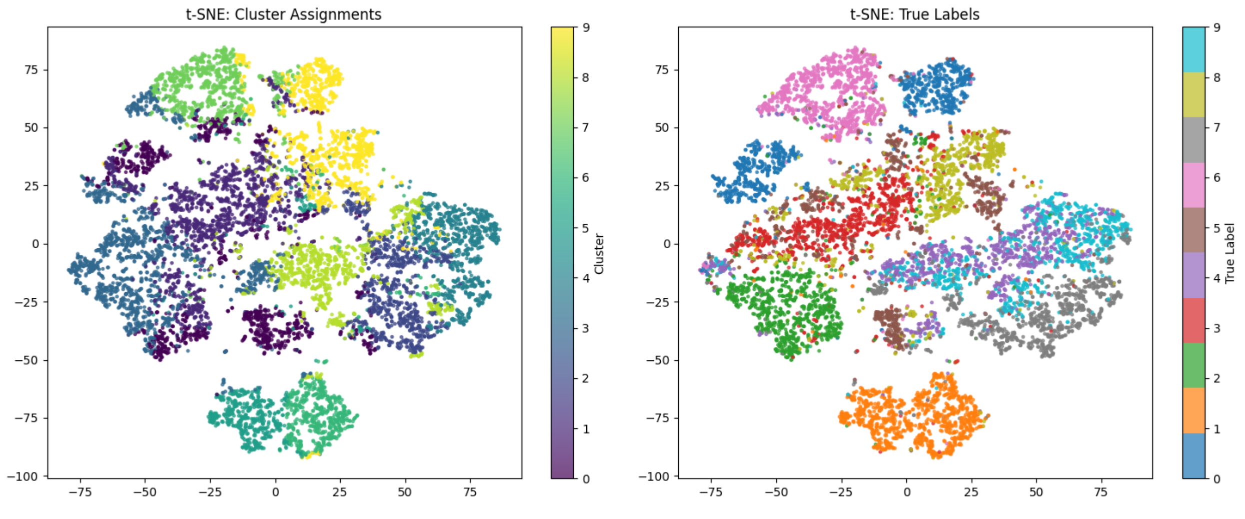
The integration of neural network-based feature extraction with clustering algorithms demonstrates the potential of hybrid approaches that combine the representation learning capabilities of deep learning with traditional unsupervised methods. The autoencoder effectively reduced dimensionality while preserving essential structural information, enabling more effective clustering.

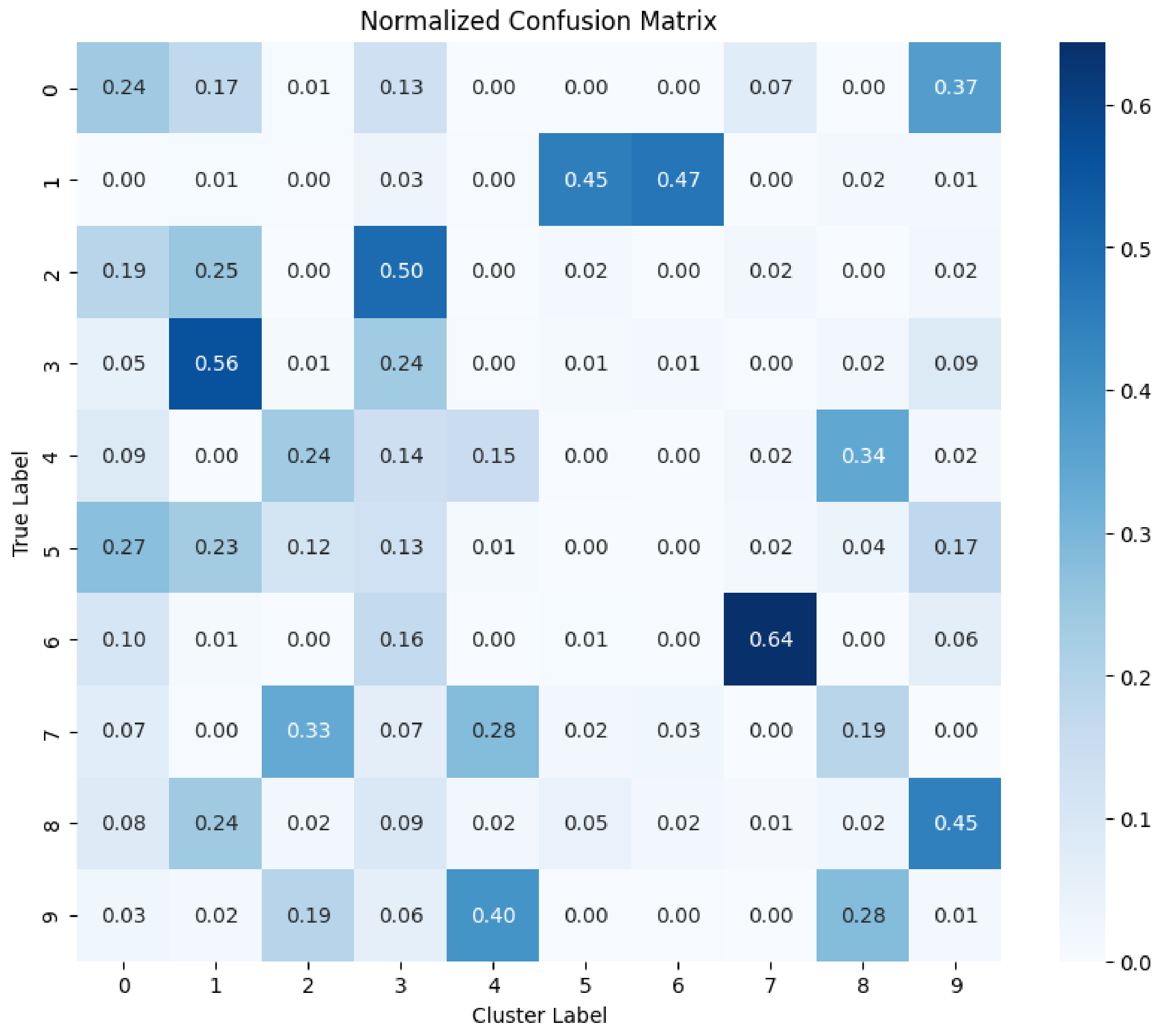
However, it's important to note that clustering algorithm performance is highly data-dependent. While K-Means proved more effective for this particular dataset, DBSCAN may still be preferable for datasets with irregular cluster shapes, varying densities, or significant noise.

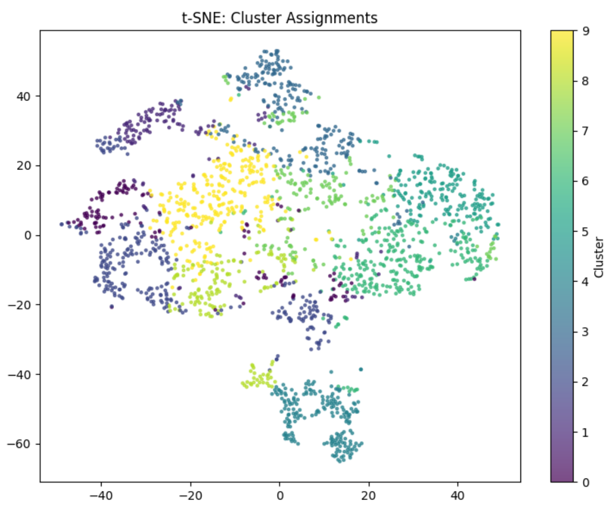
Future work could explore hybrid approaches that combine the strengths of both algorithms, parameter optimization techniques to improve DBSCAN's performance, or alternative clustering methods that might better capture the inherent structure of the data. Additionally, investigating end-to-end differentiable clustering networks that jointly optimize feature extraction and clustering could potentially yield further improvements in performance.

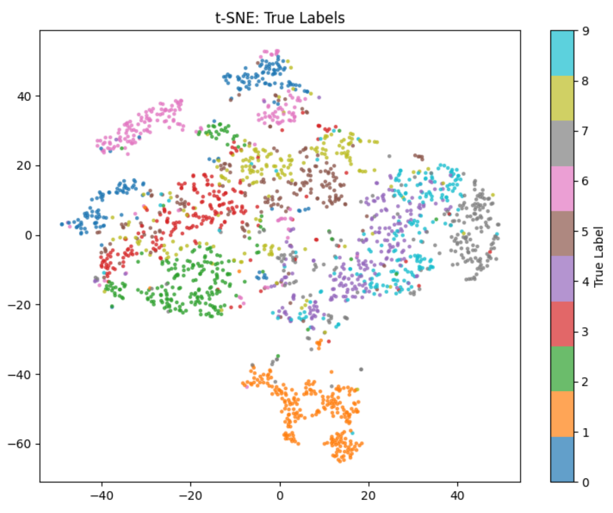
6 Graphs

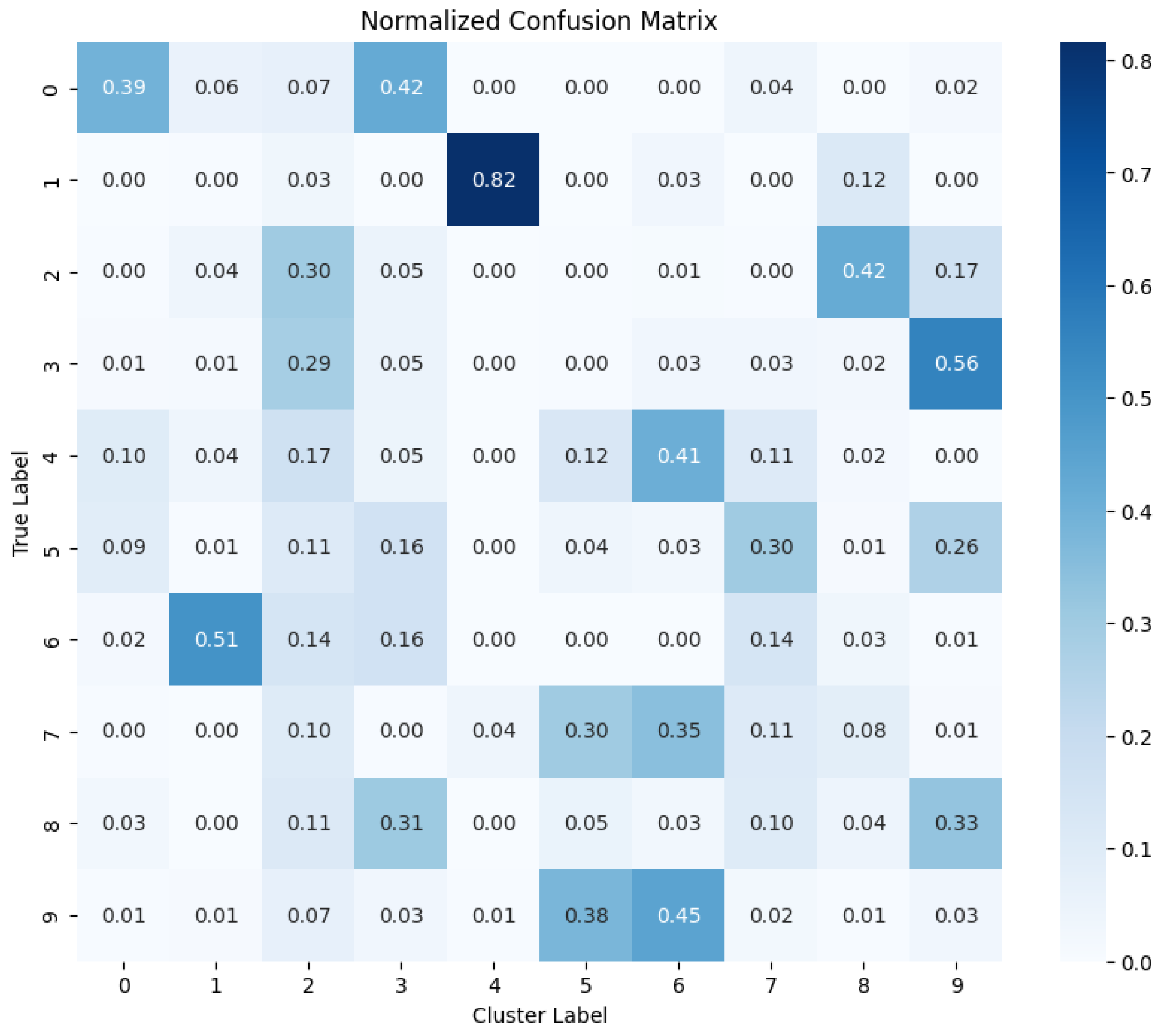

Autoencoder Training Loss
MSE Loss
Epochs
0.12
0.10
0.08
0.06
0.04
0
5
10
15
20
25
30


t-SNE: Cluster Assignments
Cluster
t-SNE: True Labels
True Label

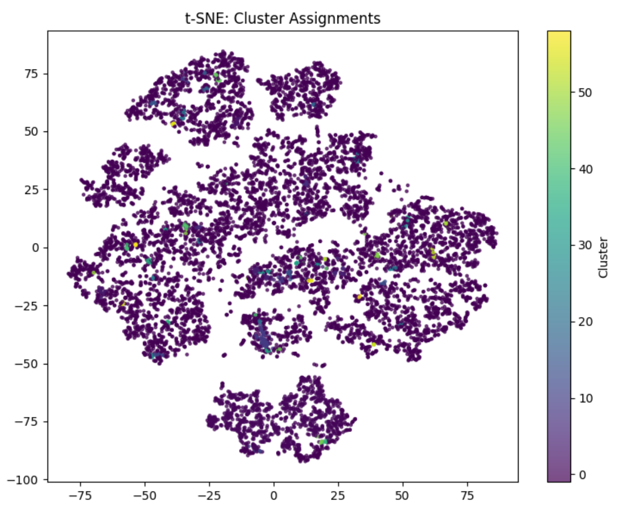

Normalized Confusion Matrix
True Label
Cluster Label
0: 0.24 0.17 0.01 0.13 0.00 0.00 0.00 0.07 0.00 0.37
1: 0.00 0.01 0.00 0.03 0.00 0.45 0.47 0.00 0.02 0.01
2: 0.19 0.25 0.00 0.50 0.00 0.02 0.00 0.02 0.00 0.02
3: 0.05 0.56 0.01 0.24 0.00 0.01 0.01 0.00 0.02 0.09
4: 0.09 0.00 0.24 0.14 0.15 0.00 0.00 0.02 0.34 0.02
5: 0.27 0.23 0.12 0.13 0.01 0.00 0.00 0.02 0.04 0.17
6: 0.10 0.01 0.00 0.16 0.00 0.01 0.00 0.64 0.00 0.06
7: 0.07 0.00 0.33 0.07 0.28 0.02 0.03 0.00 0.19 0.00
8: 0.08 0.24 0.02 0.09 0.02 0.05 0.02 0.01 0.02 0.45
9: 0.03 0.02 0.19 0.06 0.40 0.00 0.00 0.00 0.28 0.01
0 1 2 3 4 5 6 7 8 9
0.0 0.1 0.2 0.3 0.4 0.5 0.6

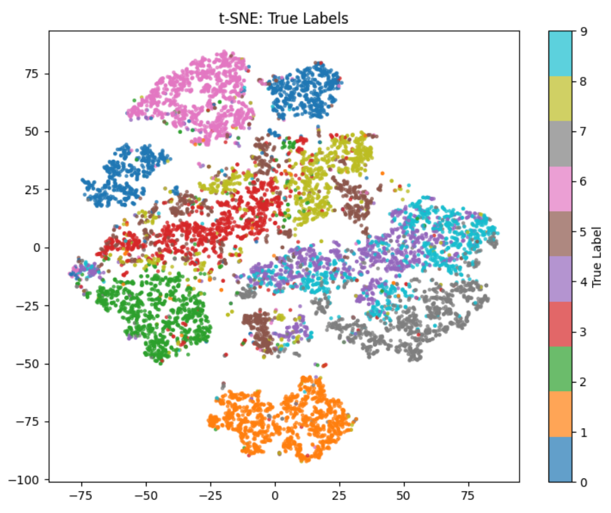

t-SNE: Cluster Assignments
Cluster
−40 −20 0 20 40
40 20 0 −20 −40 −60
0 1 2 3 4 5 6 7 8 9

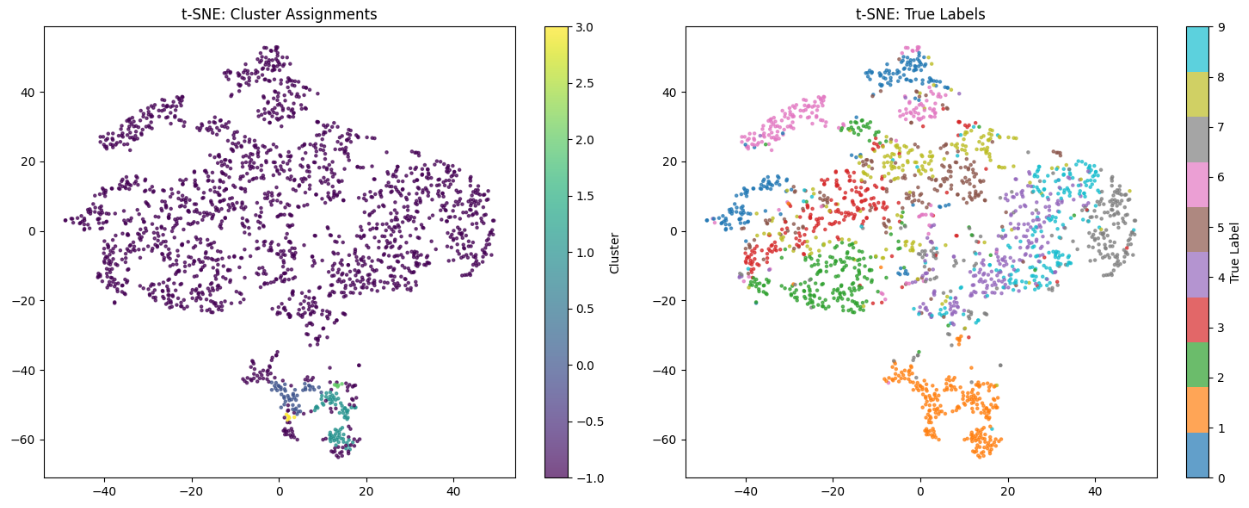

t-SNE: True Labels
True Label
−40 −20 0 20 40
40 20 0 −20 −40 −60
0 1 2 3 4 5 6 7 8 9

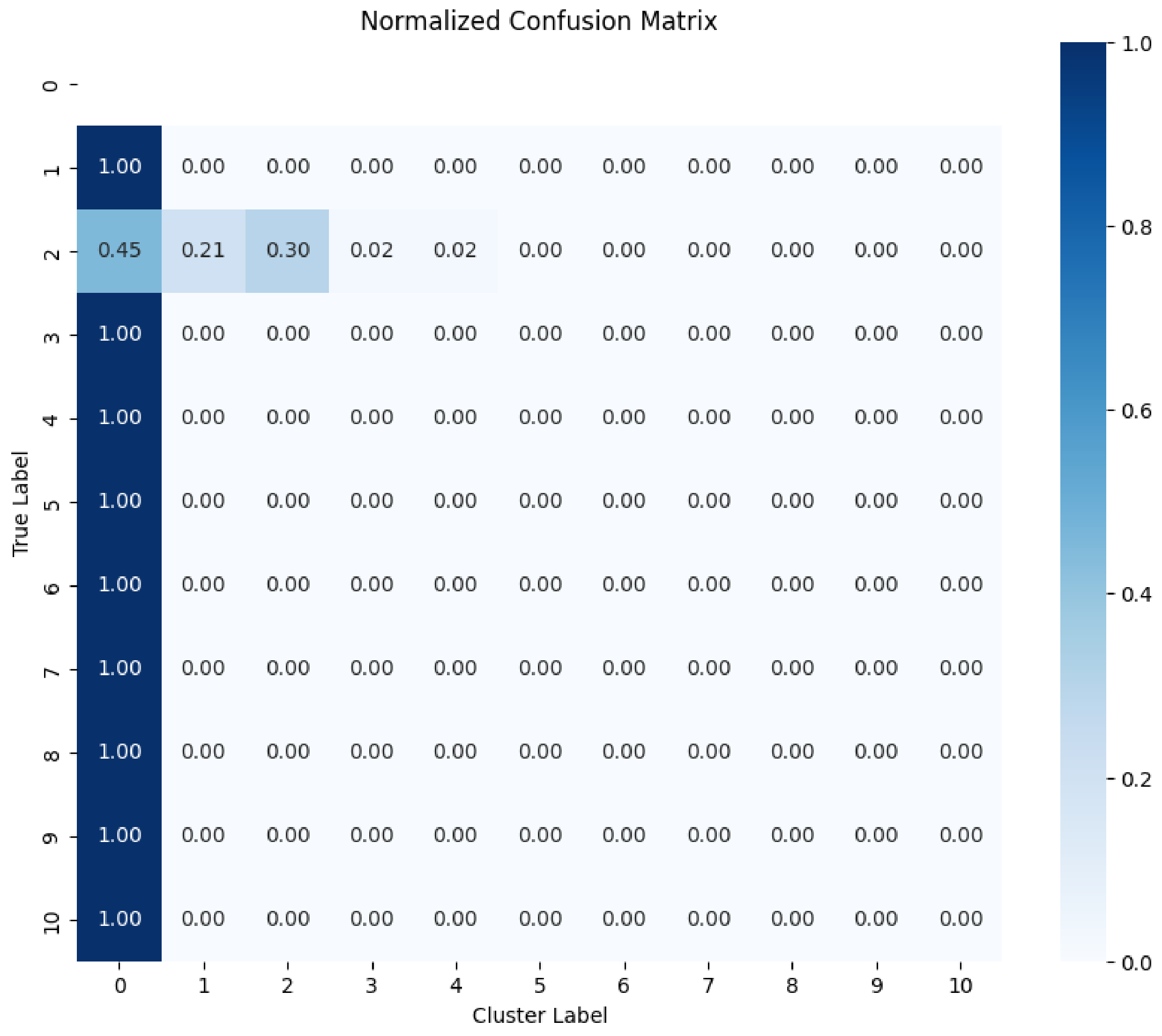


	0	1	2	3	4	5	6	7	8	9
0	0.39	0.06	0.07	0.42	0.00	0.00	0.00	0.04	0.00	0.02
1	0.00	0.00	0.03	0.00	0.82	0.00	0.03	0.00	0.12	0.00
2	0.00	0.04	0.30	0.05	0.00	0.00	0.01	0.00	0.42	0.17
3	0.01	0.01	0.29	0.05	0.00	0.00	0.03	0.03	0.02	0.56
4	0.10	0.04	0.17	0.05	0.00	0.12	0.41	0.11	0.02	0.00
5	0.09	0.01	0.11	0.16	0.00	0.04	0.03	0.30	0.01	0.26
6	0.02	0.51	0.14	0.16	0.00	0.00	0.00	0.14	0.03	0.01
7	0.00	0.00	0.10	0.00	0.04	0.30	0.35	0.11	0.08	0.01
8	0.03	0.00	0.11	0.31	0.00	0.05	0.03	0.10	0.04	0.33
9	0.01	0.01	0.07	0.03	0.01	0.38	0.45	0.02	0.01	0.03






t-SNE: Cluster Assignments
Cluster
t-SNE: True Labels
True Label


Normalized Confusion Matrix
True Label
Cluster Label