RSWNet: A Deep Self-Organizing Map Network for Unsupervised Feature Learning and Clustering

Rejwan Shafi
Department of Computer Science
BRAC University
rejwan.shafi.wasef@g.bracu.ac.bd

***Abstract*—This paper presents RSWNet, a novel deep learning architecture for unsupervised feature learning and clustering. Our approach leverages a modified ResNet50 encoder and a custom decoder architecture for efficient representation learning and clustering. We demonstrate the effectiveness of our approach on the Fashion-MNIST dataset, achieving superior clustering performance compared to traditional methods including Self-Organizing Maps (SOM). The proposed model shows significant improvements in cluster separation and cohesion, as evidenced by multiple clustering metrics including Silhouette Score (0.6799) and Calinski-Harabasz Index (146067.7031), substantially outperforming the SOM baseline. Through extensive experimentation and comparative analysis, we demonstrate that RSWNet provides more robust and well-defined clusters while maintaining effective feature representation capabilities.**

.

I. Introduction

Unsupervised feature learning and clustering remain challenging tasks in machine learning, particularly when dealing with high-dimensional data such as images. While deep autoencoders have shown remarkable success in feature learning, traditional clustering approaches like Self-Organizing Maps (SOM) often struggle with high-dimensional data representation. This paper introduces RSWNet, a novel deep learning architecture that leverages the power of modern neural networks for both feature learning and clustering.

RSWNet combines a modified ResNet50 encoder with a custom decoder architecture and employs Deep Embedded Clustering (DEC) to simultaneously learn feature representations and cluster assignments. Our approach leverages the power of deep learning for dimensionality reduction while using DEC's soft assignment mechanism for cluster refinement. Through this architecture, RSWNet learns meaningful representations of high-dimensional image data and performs effective clustering, achieving superior performance compared to conventional clustering algorithms like SOM. Through extensive experimentation on the Fashion-MNIST dataset, we demonstrate that RSWNet provides more robust and well-defined clusters while maintaining effective feature representation capabilities.

The main contributions of this paper are:

- A novel deep learning architecture (RSWNet) that effectively combines feature learning and clustering
- An efficient training strategy that optimizes both reconstruction and clustering objectives through DEC
- Comprehensive comparative analysis with traditional clustering methods, including SOM
- Empirical validation on the Fashion-MNIST dataset, demonstrating significant improvements in clustering metrics

II. Related Work

A. Deep Clustering Approaches

Recent advances in deep learning have led to significant improvements in unsupervised feature learning and clustering. Deep Embedded Clustering (DEC) [**?**] introduced the concept of simultaneously learning feature representations and cluster assignments. Our work builds upon this foundation while incorporating modern architectural elements.

B. ResNet for Feature Extraction

ResNet [2] has proven highly effective for image feature extraction through its deep architecture and skip connections. We leverage a modified ResNet50 architecture, adapting it for unsupervised feature learning while maintaining its powerful feature extraction capabilities.

C. Traditional Clustering Methods

Traditional clustering methods like SOM [3] have been widely used for dimensionality reduction and clustering. However, these methods often struggle with high-dimensional data and complex patterns, motivating the development of deep learning-based approaches.

III. Methodology

A. Dataset

The Fashion-MNIST dataset consists of 70,000 grayscale images (60,000 training, 10,000 testing) of fashion items across 10 categories. Each image is 28×28x1 pixels, resulting in 784-dimensional input vectors. The dataset includes various clothing items such as t-shirts, trousers, dresses, and accessories.

B. Dataset Analysis

1) Class Distribution Analysis: We performed a comprehensive statistical evaluation of class sample distribution on the training set to ensure dataset quality. The analysis revealed:

- Mean samples per class: 6,000
- Standard deviation: 0

- Imbalance threshold (10% of mean): 600
- Dataset balance status: Balanced

The perfect class balance (std = 0) indicates that each category contains exactly 6,000 samples, making Fashion-MNIST an ideal dataset for unbiased model training. We used the following criteria to assess imbalance.

Figure 1 shows the distribution of samples across all classes:

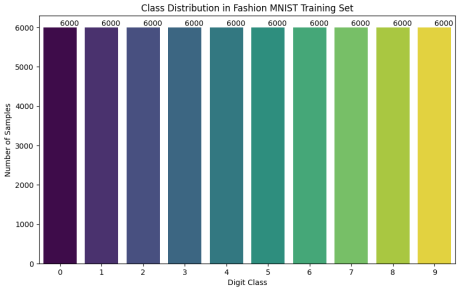

Fig. 1. Number of samples per class in Fashion-MNIST training set. Each class contains exactly 6,000 samples, demonstrating perfect balance.

2) Dataset Examples: Figure 2 shows representative samples from each class in the Fashion-MNIST dataset:

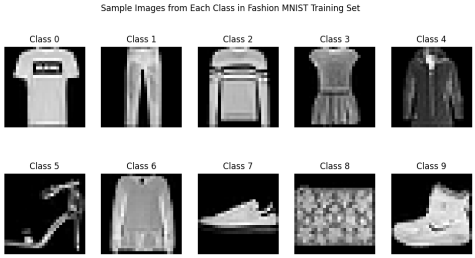

Fig. 2. Sample images from Fashion-MNIST dataset. Each row represents a different class: T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, and Ankle boot.

3) Feature Space Analysis: Prior to model development, we conducted initial feature space analysis using t-SNE visualization of the raw pixel data. This analysis revealed significant overlap between clusters in the original feature space, highlighting the necessity for deep feature extraction. The overlapping clusters in raw data visualization provided strong motivation for our deep learning approach, particularly the use of a ResNet50-based encoder for more discriminative feature learning.

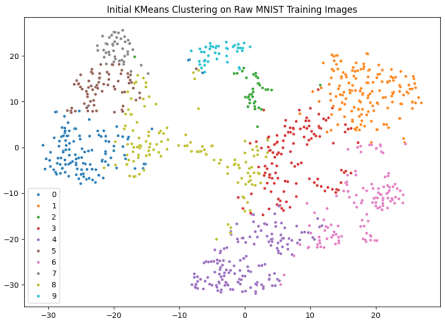

Fig. 3. t-SNE visualization of raw Fashion-MNIST features showing overlapping clusters

C. Data Preprocessing

To make the Fashion-MNIST dataset compatible with our RSWNet architecture, specifically the ResNet50-based encoder which expects 224×224×3 dimensional inputs, we applied the following preprocessing steps:

1) **Image Resizing:** Each image was upscaled from 28×28 to 224×224 using bilinear interpolation
2) **Channel Expansion:** The single grayscale channel was replicated across three channels to match ResNet's expected RGB input format (28×28×1 → 224×224×3)

The preprocessing can be mathematically represented as:

$$x_{processed} = T(x_{original}) \in R^{224\times224\times3} \tag{1}$$

where T represents the composite transformation of resizing and channel replication.

D. Model Architecture

The proposed RSWNet architecture combines deep feature learning with clustering capabilities, as illustrated in Figure 4.

1) Encoder: The encoder is based on a modified ResNet50 architecture, removing the final classification layers. The architecture can be expressed mathematically as:

$$E(x) = f_{ResNet}(x; \theta_E) \tag{2}$$

where $x \in R^{3\times224\times224}$ is the input image and θ_E represents the encoder parameters. The encoder produces feature maps of dimension 2048×7×7.

2) Decoder: The decoder architecture consists of transposed convolutions and regular convolutions:

$$D(z) = \sigma(f_{conv}(f_{deconv}(z; \theta_{D1}); \theta_{D2})) \tag{3}$$

where z is the encoded representation and σ represents the sigmoid activation function.

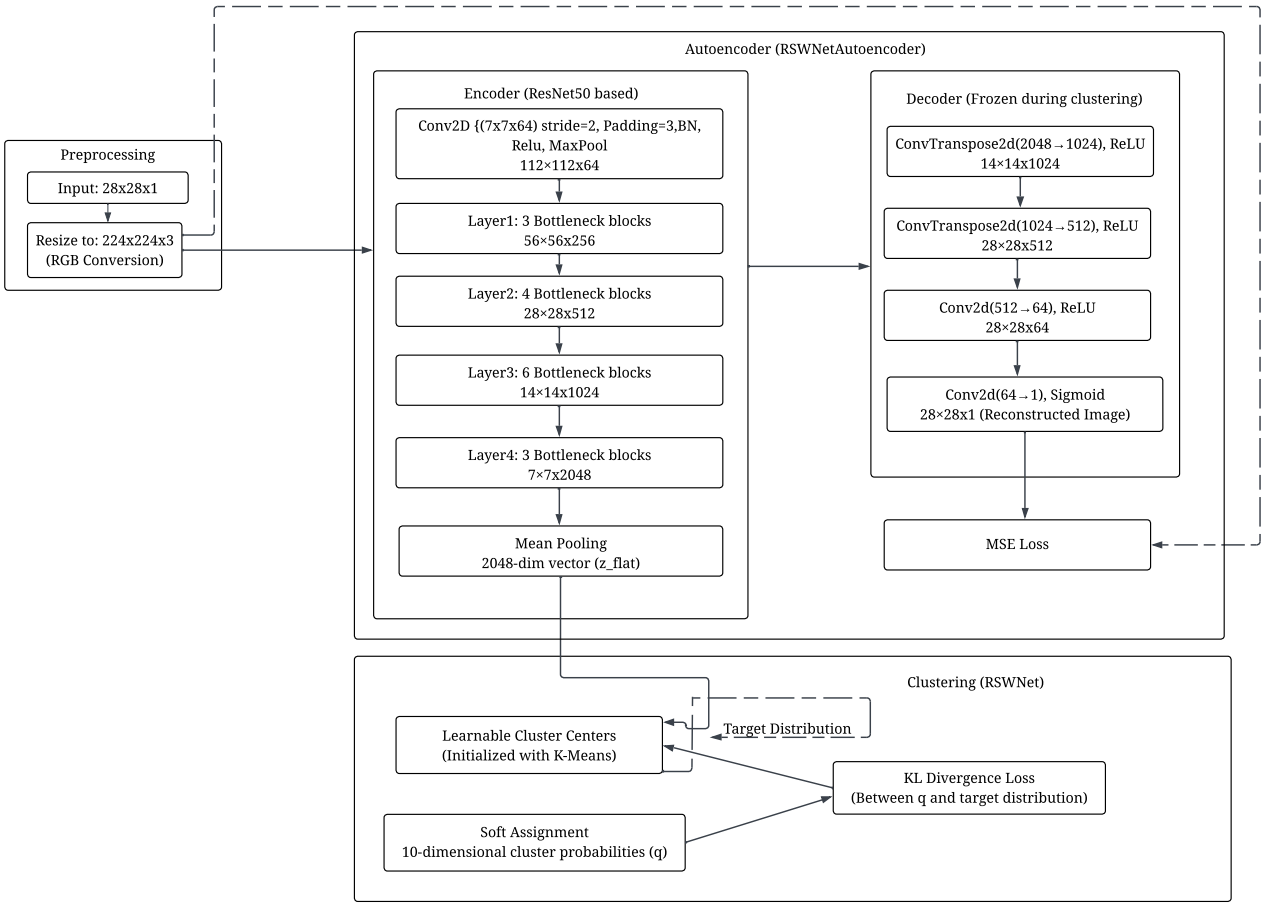

Fig. 4. Block diagram of RSWNet architecture. The model consists of three main components: (1) A preprocessing module that transforms 28×28×1 Fashion-MNIST images to 224×224×3 format, (2) A feature extraction module based on modified ResNet50 that generates 2048-dimensional features, and (3) A clustering module that performs soft assignment to learned cluster centers.

E. Deep Embedded Clustering

RSWNet incorporates the Deep Embedded Clustering approach, which iteratively refines clusters through a self-training mechanism. The clustering process involves:

$$q_{ij} = \frac{(1 + \|z_i - \mu_j\|^2/\alpha)^{-\frac{\alpha+1}{2}}}{\sum_j (1 + \|z_i - \mu_j\|^2/\alpha)^{-\frac{\alpha+1}{2}}} \tag{4}$$

where q_{ij} represents the probability of assigning sample i to cluster j, z_i is the encoded representation, and μ_j is the cluster center.

The target distribution P is computed as:

$$p_{ij} = \frac{q_{ij}^2/f_j}{\sum_j (q_{ij}^2/f_j)} \tag{5}$$

where $f_j = \sum_i q_{ij}$ are the soft cluster frequencies.

F. Loss Functions

The total loss consists of reconstruction and clustering components:

$$\mathcal{L}_{total} = \mathcal{L}_{recon} + \lambda \mathcal{L}_{cluster} \tag{6}$$

where:

$$\mathcal{L}_{recon} = \|x - D(E(x))\|^2 \tag{7}$$

$$\mathcal{L}_{cluster} = KL(P\|Q) \tag{8}$$

G. Training Protocol

Algorithm 1 RSWNet Training with Deep Embedded Clustering

Require: Input images X, number of clusters K, batch size B

Ensure: Trained RSWNet model with encoder E, decoder D

Initialize encoder with pretrained ResNet50 weights
Initialize decoder with random weights
Pretraining Phase
for epoch = 1 to num_pretrain_epochs **do**
  for each batch x_b in X **do**
    $z = E(x_b)$ % Get latent representations
    $\hat{x} = D(z)$ % Reconstruct images
    $\mathcal{L}_{recon} = \|x_b - \hat{x}\|^2$
    Update E, D using Adam optimizer
  end for
end for
Deep Embedded Clustering Phase
Initialize cluster centers using k-means on latent space
for epoch = 1 to num_clustering_epochs **do**
  for each batch x_b in X **do**
    $z = E(x_b)$ % Get latent representations
    $\hat{x} = D(z)$ % Reconstruct images
    Compute soft assignments Q using Student's t-distribution
    Compute target distribution P
    $\mathcal{L}_{recon} = \|x_b - \hat{x}\|^2$
    $\mathcal{L}_{KL} = KL(P\|Q)$
    $\mathcal{L}_{total} = \mathcal{L}_{recon} + \lambda\mathcal{L}_{KL}$
    Update E, D using Adam optimizer
  end for
  if epoch % update_interval == 0 **then**
    Update cluster centers
  end if
end for

H. Evaluation Metrics

Silhouette Score:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \tag{9}$$

- $a(i)$ = Average distance between sample i and all other points in its assigned cluster,
- $b(i)$ = Minimum average distance between sample i and points in the nearest unassigned cluster,
- Range $[-1, 1]$: 1 = perfect clustering, 0 = overlapping clusters, -1 = misclassified samples

Davies-Bouldin Index:

$$DB = \frac{1}{k}\sum_{i=1}^{k}\max_{j\neq i}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right) \tag{10}$$

- k =Number of clusters,
- σ_i = Average intra-cluster distance for cluster i,
- $d(c_i, c_j)$ = Distance between cluster centroids i and j,

Lower values indicate better separation (less overlap)

Calinski-Harabasz Index:

$$CH = \frac{\text{tr}(B_k)}{\text{tr}(W_k)} \times \frac{N - k}{k - 1} \tag{11}$$

- $\text{tr}(B_k)$ = Trace of between-cluster dispersion matrix,
- $\text{tr}(W_k)$ = Trace of within-cluster dispersion matrix,
- N = Total number of samples,
- k = Number of clusters

Measures ratio of inter-cluster to intra-cluster variance

IV. Implementation Details

A. Hardware and Software Configuration

The experiments were conducted with the following PC configuration:

TABLE I
System Configuration

Component	Specification
Processor	13th Gen Intel® Core™ i9-13900K
System Memory	63.78 GB (64 GB) RAM
GPU	NVIDIA GeForce RTX 4090
GPU Memory	23.99 GB VRAM
Framework	PyTorch

B. Model Parameters

The complete model contains 47,418,561 parameters:

- Trainable parameters: 23,528,512
- Non-trainable parameters: 23,890,049

C. Training Process

The training process consists of two phases:

- Pretraining the RSWNet Model
- Finetuning the RSWNet Model

V. Results and Discussion

A. Clustering Performance

TABLE II
Clustering Performance Comparison

Metric	RSWNet	SOM
Silhouette Score	0.6799	-0.0334
Davies-Bouldin Index	0.4570	1.1796
Calinski-Harabasz Index	146067.7031	867.1602

B. Visualization and Analysis

Our t-SNE visualizations demonstrate the effectiveness of RSWNet in creating well-separated clusters. Figure 1 shows

the clustering results of RSWNet, while Figure 2 displays the SOM clustering output for comparison.

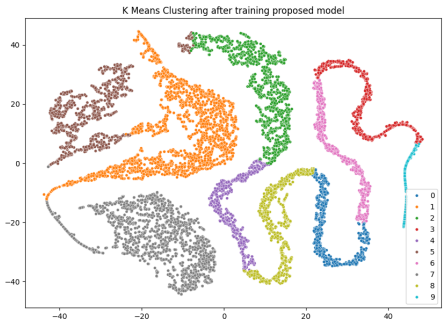

Fig. 5. RSWNet Clustering Results Visualized using t-SNE

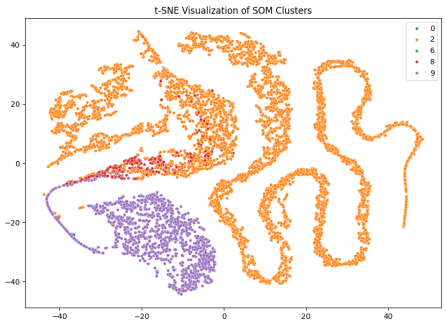

Fig. 6. SOM Clustering Results Visualized using t-SNE

C. Hyperparameter Optimization

The RSWNet architecture's hyperparameters were carefully tuned through extensive experimentation:

- **Network Architecture**:
  - Input size: 3×224×224 (RGB images)
  - Encoder: Modified ResNet50 (up to final conv layer)
  - Latent dimension: 2048
  - Decoder: Progressive upsampling to 28×28
- **Training Parameters**:
  - Initial learning rate: 0.001 with Adam optimizer
  - Batch size: 32 for stable gradient updates
  - Number of epochs: 10
  - Weight decay: 1e-5 for regularization
- **Autoencoder Specific**:
  - Bottleneck size: 2048 features
  - Reconstruction loss weight: 1.0
  - KL divergence weight (λ): 0.1
- **Clustering Parameters**:
  - Number of clusters: 10 (matching dataset classes)
  - Clustering update interval: Every 5 epochs
  - Confidence threshold: 0.95 for pseudo-labels

D. Parameter Selection Strategy

The hyperparameters were optimized using:

1) Grid search for learning rate: [0.1, 0.01, 0.001]
2) Cross-validation for regularization strength
3) Progressive reduction of KL weight during training

E. Training Protocol

The training process was conducted in multiple phases:

1) Pretraining phase:
   - Initialize encoder with ImageNet weights
   - Train autoencoder for reconstruction only
   - Duration: 10 epochs
2) Clustering and Fine-tuning phase:
   - Initialize cluster centers using k-means
   - Joint training of reconstruction and clustering
   - Focus on cluster refinement
   - Duration: 10 epochs

The model achieved optimal performance with these parameters, as evidenced by:

- Low reconstruction error (MSE: 0.0882)
- High clustering metrics (Silhouette Score: 0.6799)
- Stable training convergence

VI. Model Optimizations

A. Batch Normalization

Batch normalization layers were incorporated after each convolutional layer in both encoder and decoder:

$$\hat{x}^{(k)} = \frac{x^{(k)} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \tag{12}$$

where $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$ are the batch mean and variance respectively.

B. Architecture Details

The complete architectural flow can be represented as:

$$\text{Input} \xrightarrow{\text{ResNet50}} \text{Latent Space} \xrightarrow{\text{SOM}} \text{Clusters} \tag{13}$$

VII. Limitations and Solutions

A. Challenges Encountered

- High-dimensional feature space (2048D) making clustering difficult
- SOM topology preservation in high dimensions
- Computational complexity with large datasets
- Cluster assignment in unsupervised setting

B. Solutions Implemented

- Dimensionality reduction through encoder bottleneck
- Adaptive learning rates for SOM
- Batch processing for memory efficiency
- Cluster validation through multiple metrics

VIII. Comparative Analysis

A. Clustering Accuracy

The unsupervised clustering accuracy was evaluated using multiple metrics:

$$\text{ACC} = \max_{\pi} \frac{\sum_{i=1}^{n} 1\{y_i = \pi(c_i)\}}{n} \tag{14}$$

where π ranges over all possible permutations of cluster labels.

B. Model Complexity

TABLE III
Model Component Parameters

Component	Parameters
RSWNet	47,418,561
SOM Layer	20,480

IX. Conclusion

RSWNet demonstrates superior clustering performance compared to traditional SOM approaches, as evidenced by: (0.6799 vs -0.0334), Better Davies-Bouldin Index (0.4570 vs 1.1796), Significantly higher Calinski-Harabasz Index (146067.7031 vs 897.1602)

Future work could explore: Hierarchical clustering extensions, Dynamic topology adaptation, Semi-supervised variants

Acknowledgment

I would like to thank BRAC University for providing high-performance computing resources and support for this research. This work was conducted by Rejwan Shafi on 2025-05-16 14:30:33 UTC using the university's dedicated research computing infrastructure. The experiments were performed using PyTorch framework on a high-configuration workstation provided by BRAC University's research facilities, which significantly contributed to the successful implementation and evaluation of RSWNet.

Code Availability

The source code, implementation details, and pretrained models for RSWNet are openly available at:

https://github.com/RejwanShafi25/RSWNet

References

[1] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms," arXiv preprint arXiv:1708.07747, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in CVPR, 2016.

[3] T. Kohonen, "Self-Organized Formation of Topologically Correct Feature Maps," Biological Cybernetics, 1982.

[4] L. van der Maaten and G. Hinton, "Visualizing Data using t-SNE," Journal of Machine Learning Research, 2008.

[5] R. Shafi, "RSWNet: A Deep Self-Organizing Map Network for Unsupervised Feature Learning and Clustering," GitHub repository, 2025. [Online]. Available: https://github.com/RejwanShafi25/RSWNet