BM20A6100 Advanced Data Analysis and Machine Learning

Homework: Advanced Dimensionality Reduction (DL 3.11.)

Bohao Xing[1]

[1]**bohao.xing@student.lut.fi**

Solve the requirements of the exercise and submit a link to your GitHub repository.

1. Comparing linear and non-linear DR (4 points)
Compare PCA and t-SNE methods by visualizing Bike Sharing Rental dataset. Explore how the different features are shown in the DR components. Build a simple prediction model (for example, MLP or Random Forest) to predict the count of total rental bikes and compare the performance of the model with the different DR techniques.

2. Visualizing with SOM (3 points)
Visualize the MNIST-784 handwritten digits dataset with SOM and discuss what you can learn from the visualization.

1 COMPARING LINEAR AND NON-LINEAR DR

1.1 t-SNE

t-SNE (t-Distributed Stochastic Neighbor Embedding) is a non-linear dimensionality reduction technique mainly used for visualizing high-dimensional data in 2D or 3D. It models pairwise similarities between data points using probability distributions: Gaussian in the high-dimensional space and Student's t-distribution in the low-dimensional space.
The algorithm minimizes the Kullback–Leibler divergence between these two distributions using gradient descent, so that nearby points in the original space remain close after projection.
The key parameter, perplexity, controls the effective number of neighbors and affects the balance between local and global structure.

1.2 Dataset Description

The dataset used in this exercise is the **Bike Sharing Rental** dataset (OpenML ID 42712). It contains hourly records of bike rentals in a bike-sharing system, including both environmental and temporal features.
Each record represents the count of total rented bikes, divided into `casual` and `registered` users. The main features include:

- `season`, `year`, `month`, `hour`, `weekday`, `holiday`, and `workingday` — categorical or temporal indicators;
- `weather`, `temp`, `feel_temp`, `humidity`, and `windspeed` — continuous weather-related variables;
- `count` — the target variable representing the total number of rented bikes.

1.3 Hyper parameters

To further explore the behavior of t-SNE, different distance metrics were applied, including Mahalanobis, Cosine, Chebyshev, and Euclidean distances. The resulting visualizations in Figure 1 demonstrate how the choice of distance metric affects the structure of the low-dimensional embedding. Although all variants reveal similar local clusters, the Euclidean metric achieves slightly lower Kullback–Leibler (KL) divergence values, indicating better preservation of pairwise similarities in the reduced space. Therefore, the **Euclidean** metric is adopted for all subsequent experiments.
To examine the effect of the perplexity parameter in t-SNE, the algorithm was tested with values of 10, 30, and 50 using the Euclidean distance metric. As shown in Figure 2, a small perplexity (10) results in fragmented clusters and higher KL divergence, indicating that local neighborhoods are overemphasized. Perplexity = 30 was selected for subsequent experiments.

1.4 PCA and t-SNE Dimensionality Reduction

Then, the Bike Sharing Rental dataset was visualized using both PCA and t-SNE to compare linear and non-linear dimensionality reduction methods. PCA projects the data into a lower-dimensional space that preserves the directions of maximum variance, while t-SNE focuses on maintaining local neighborhood relationships to reveal non-linear patterns in the data. As shown in the Figure 3 and Figure 4, PCA produces a dense, continuous distribution with overlapping samples, capturing global variance but showing limited separation between different rental count levels. In contrast, t-SNE forms distinct local clusters that correspond to similar rental counts, better highlighting the underlying non-linear structure of the data.

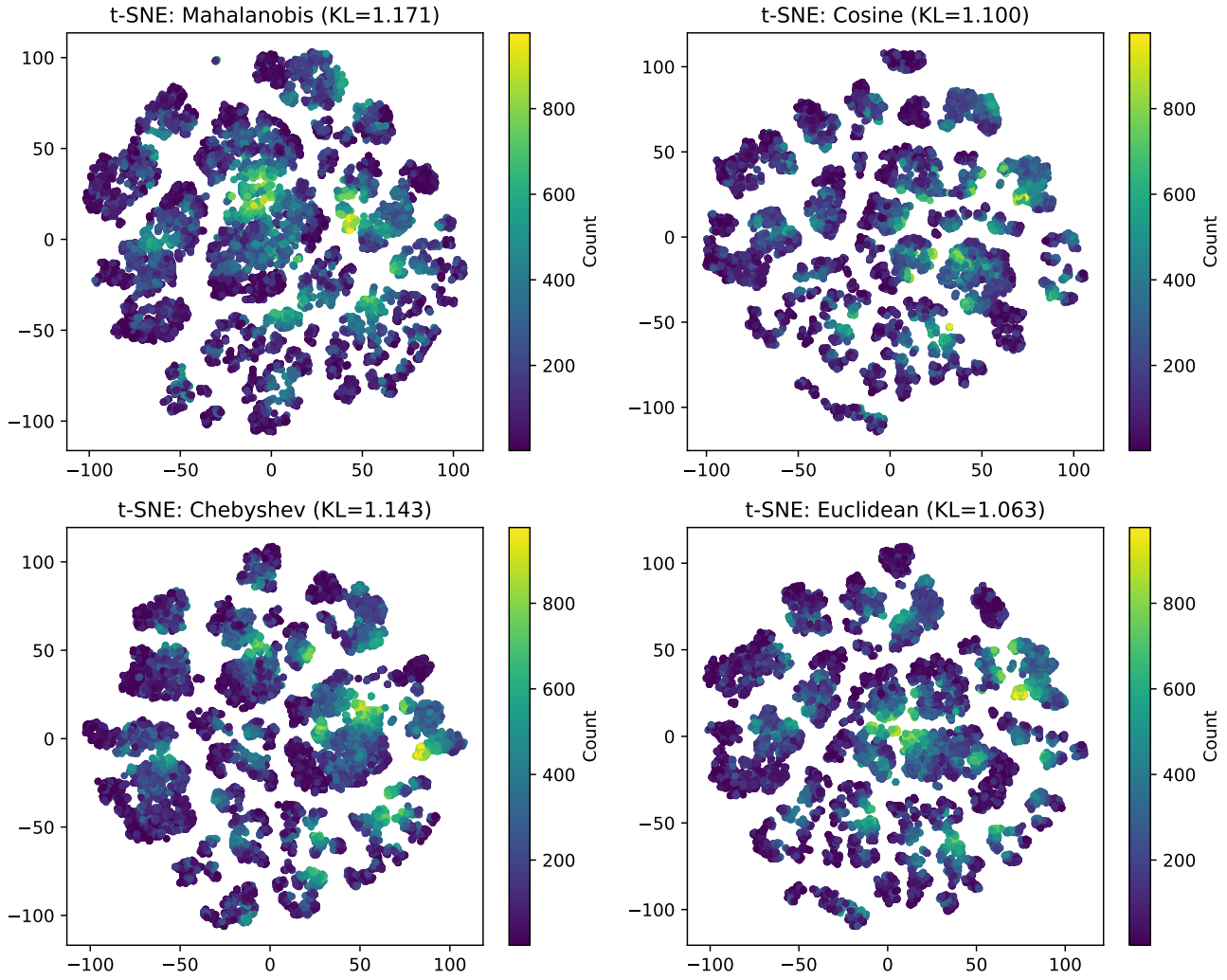


Figure 1. t-SNE visualizations of the Bike Sharing dataset using different distance metrics: Mahalanobis, Cosine, Chebyshev, and Euclidean.

However, the global distances in t-SNE are not directly interpretable, and results can vary depending on initialization and perplexity settings.

1.5 Explore how the different features are shown in the DR components

Figure 5 illustrates how individual features contribute to the first two principal components (PC1 and PC2). The *hour*, *temperature*, *feeling temperature*, and *registered user counts* show strong positive loadings on PC1, suggesting that this component largely captures variations related to **time-of-day and weather-driven demand intensity**. Meanwhile, *weather* and *humidity* contribute negatively to PC1, indicating their inverse relationship with the overall pattern captured by this axis. For PC2, features such as *month*, *feeling temperature*, and *humidity* exhibit higher positive weights, while *hour* and *windspeed* show negative contributions. This implies that PC2 may reflect **seasonal or environmental fluctuations** distinct from the day-to-day variations described by PC1.

Figure 6 visualizes how each original feature is distributed across the t-SNE embedding space. The t-SNE projection preserves local similarities among samples, allowing the visualization of how individual

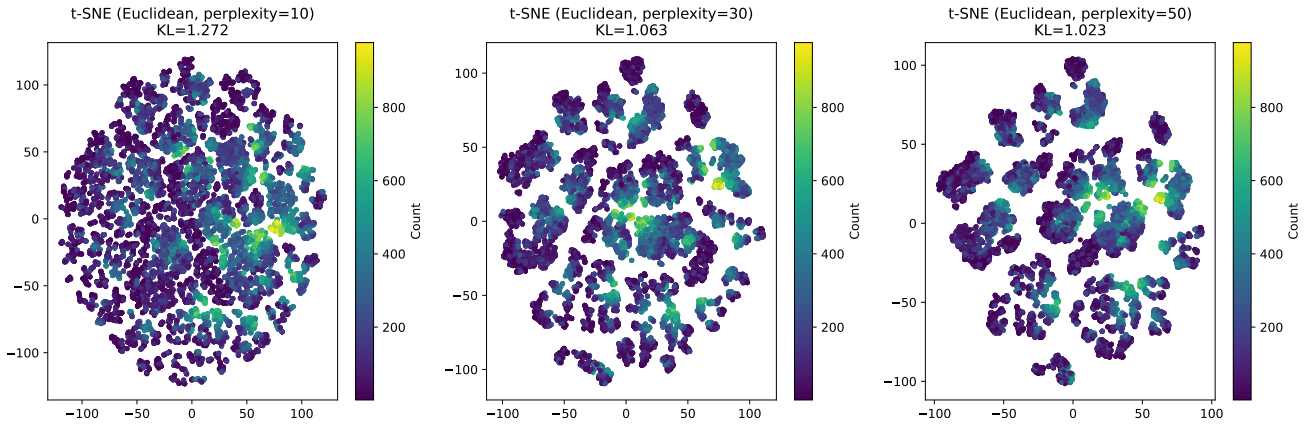


Figure 2. t-SNE visualizations of the Bike Sharing dataset using different perplexity values (10, 30, and 50).

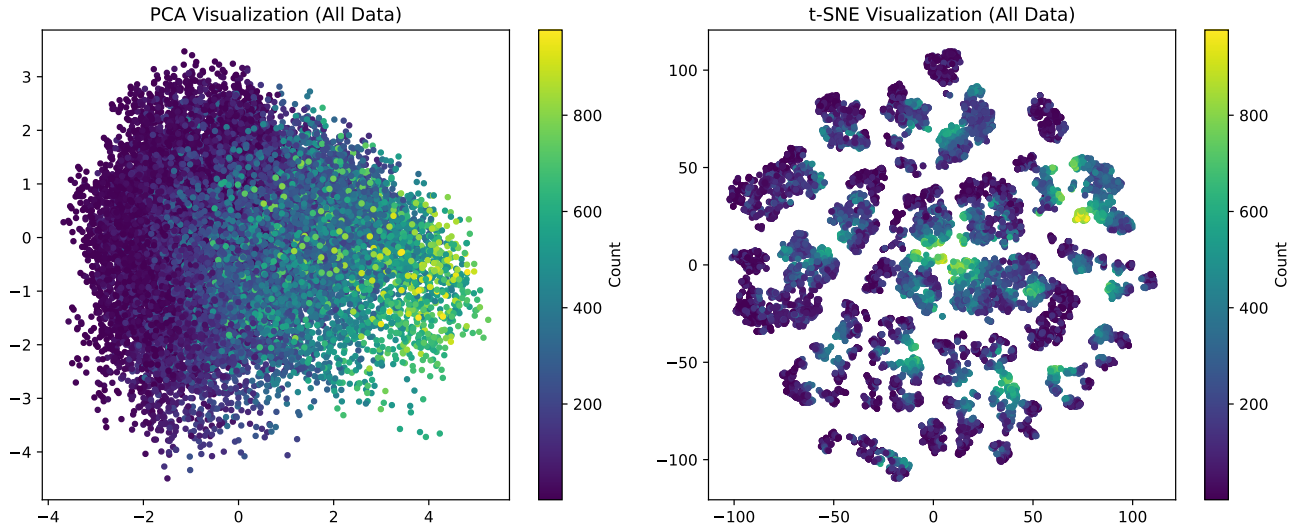


Figure 3. Two-dimensional visualization of the Bike Sharing dataset using PCA (left) and t-SNE (right).

features vary within the same low-dimensional manifold.

Distinct patterns emerge for several features. For instance, *season*, *year*, *month*, and *hour* exhibit clear block structures, reflecting temporal separations captured by the t-SNE embedding. Environmental factors such as *temperature*, *feeling temperature*, and *humidity* show smoother gradients across the space, indicating gradual transitions in weather conditions rather than abrupt clustering. In contrast, categorical or binary variables such as *holiday*, *workingday*, and *weather* appear as more discrete regions due to their limited value ranges. The demand-related variables *casual* and *registered* show partially overlapping yet distinct distributions, suggesting that user type is correlated with other contextual and environmental factors represented in the t-SNE structure.

1.6 Comparison of MLP Performance using PCA and t-SNE Representations

Figure 7 compares the performance of the MLP regression model when trained on low-dimensional representations obtained from PCA and t-SNE. Three evaluation metrics are presented: Root Mean

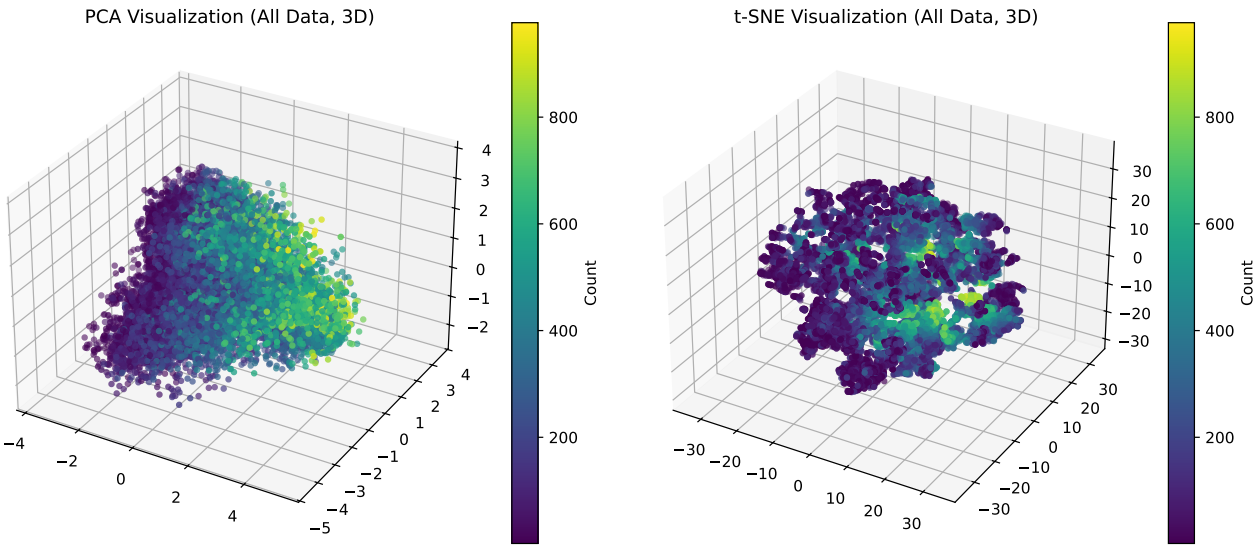


Figure 4. Three-dimensional visualization of the Bike Sharing dataset using PCA (left) and t-SNE (right).

Squared Error (RMSE), Mean Absolute Error (MAE), and the coefficient of determination (R^2). In general, both PCA and t-SNE show improvements as the number of components increases, with PCA consistently achieving slightly better performance across all metrics.

Although t-SNE is a nonlinear dimensionality reduction method capable of capturing complex manifold structures, the results suggest that nonlinear embeddings do not necessarily lead to superior predictive performance in regression tasks. This may be because t-SNE focuses on preserving local neighborhood relationships rather than global variance or linear correlations that are more relevant to numerical prediction. Consequently, the simpler linear PCA representation provides a more stable and generalizable basis for the MLP regression model.

It should also be noted that t-SNE does not provide a transformation function for unseen data. As a result, the embeddings used here were computed on the entire dataset before splitting into training and testing subsets, which introduces a potential data leakage issue. This means that some information from the test set may have indirectly influenced the learned representation, possibly leading to optimistic performance estimates.

Finally, the current analysis does not fully account for the effect of the number of retained principal components. While PCA can efficiently scale to higher dimensions, t-SNE becomes increasingly difficult to optimize when provided with more input components due to its high computational cost and sensitivity to hyperparameter settings. As a result, experiments involving larger feature spaces or higher-dimensional embeddings are often impractical for t-SNE, limiting its applicability compared to PCA in scenarios where scalability, interpretability, and reproducibility are critical considerations.

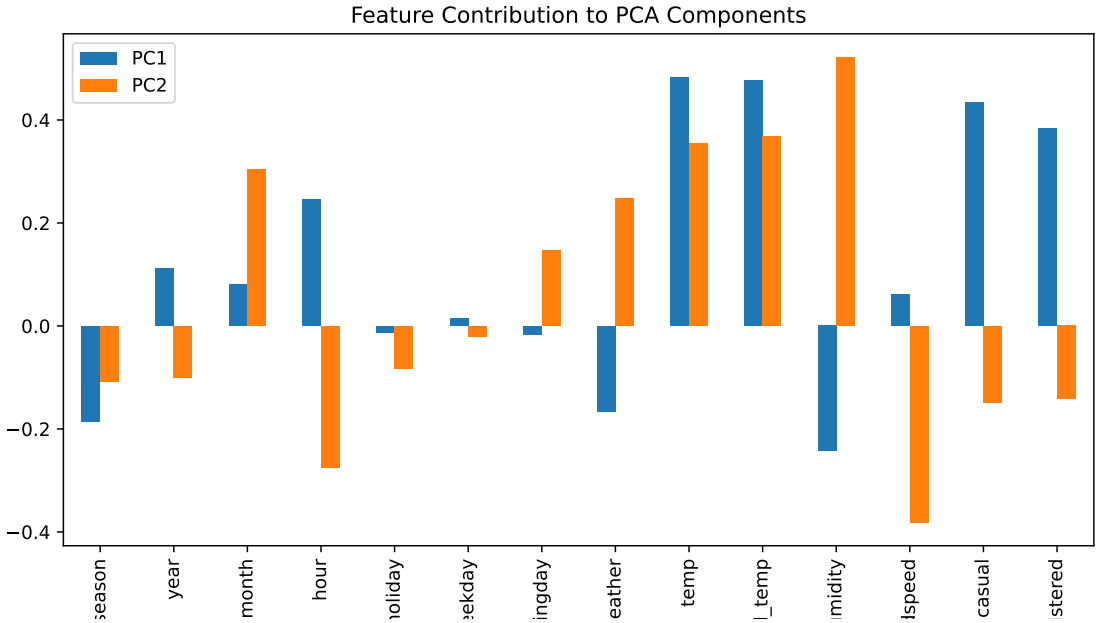


Figure 5. Feature contribution to the first two PCA components.

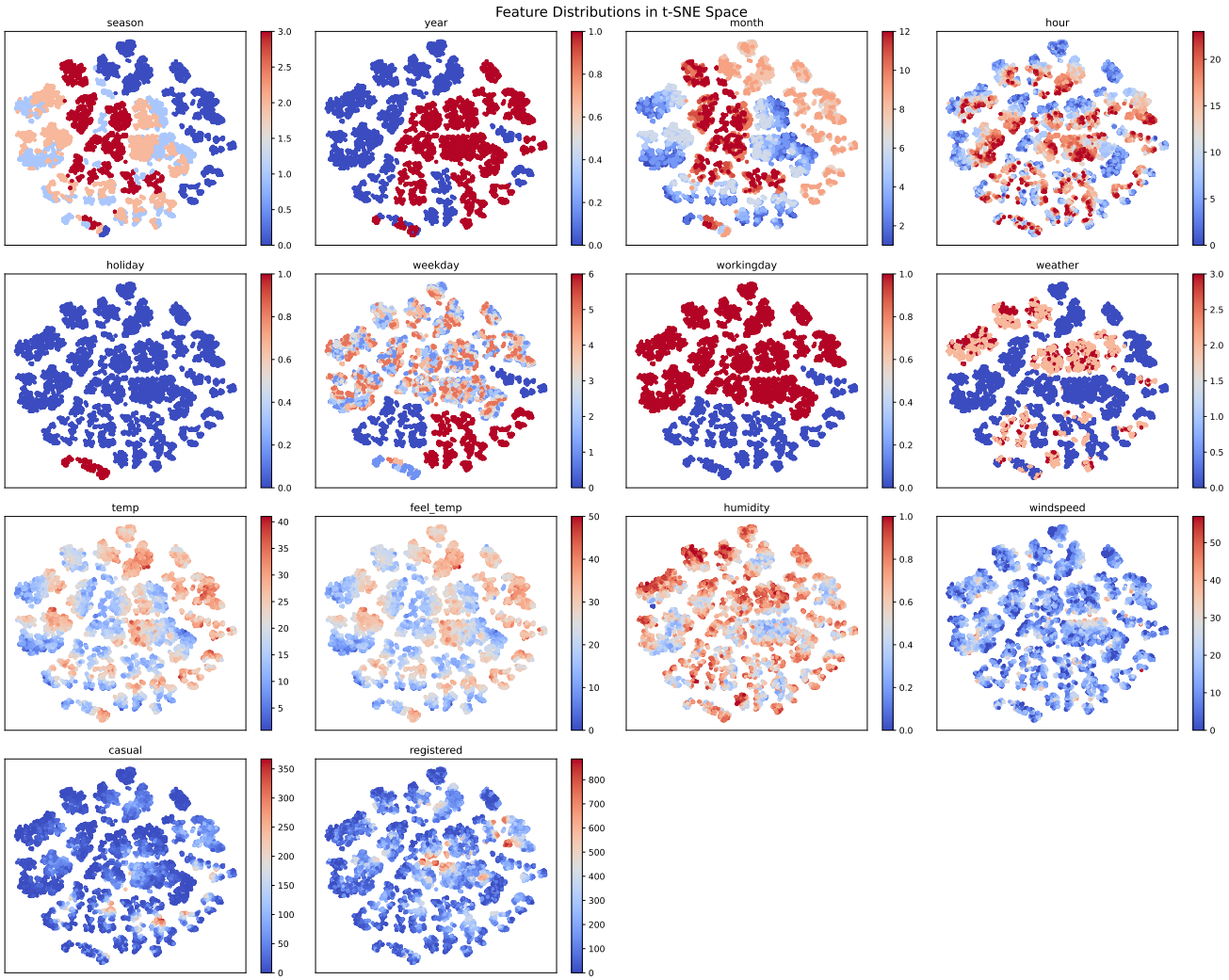


Figure 6. Feature distributions in tSNE space.

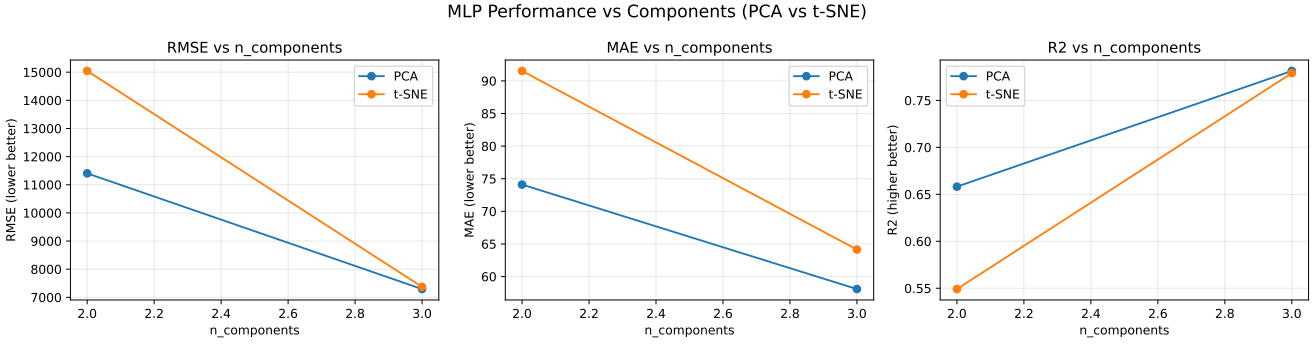


Figure 7. MLP performance on PCA and tSNE.

2 VISUALIZING WITH SOM

2.1 SOM

The Self-Organizing Map (SOM) is a type of unsupervised neural network used for non-linear dimensionality reduction and visualization. It maps high-dimensional data onto a low-dimensional (typically 2D) grid while preserving the topological relationships between input samples. During training, each data sample activates the most similar node, called the Best Matching Unit (BMU), and both the BMU and its neighboring nodes are updated toward the input vector. As learning progresses, similar inputs are organized close to each other on the map, forming meaningful clusters and smooth transitions between regions. SOM is particularly effective for visualizing complex, high-dimensional data such as images or time series.

Considering the large size of the MNIST dataset, training with `MiniSom` was computationally slow. Therefore, the implementation was switched to `TorchSOM`, which allows GPU acceleration and faster convergence. All experiments were conducted on Google Colab with GPU support. After a brief hyperparameter search, the mini-batch size was set to 512 and the learning rate to 0.5. Unlike the MATLAB version, the full dataset was not used as a single batch, as this approach led to slower convergence. Due to computational constraints, no ablation study was performed on these two parameters.

2.2 Dataset Description

The dataset used for this part is the **MNIST-784** handwritten digits dataset (penML ID 554). It contains 70000 grayscale images of handwritten digits ($0-9$), each represented as a 784-dimensional vector (28×28 pixels). The dataset is commonly used for benchmarking classification and visualization algorithms. In this experiment, SOM is applied to map the high-dimensional image space onto a 2D grid, allowing us to observe how different digits cluster.

2.3 Training Performance and Hyperparameter Justification

To evaluate the SOM training process, both **quantization error** and **topographic error** were monitored across 250 epochs, as shown in Figure 8. The quantization error measures the average distance between input samples and their best matching units (BMUs), indicating how well the SOM represents the input space. The topographic error measures how well the neighborhood relationships of the high-dimensional data are preserved in the 2D map. As shown in the figure, the quantization error decreases steadily and stabilizes after approximately 200 epochs, demonstrating consistent convergence of the network. Although the topographic error shows a gradual increase, it remains within a reasonable range, suggesting that the topology is generally preserved without significant distortion. These trends validate the choice of training parameters: a learning rate of 0.5 and 250 training epochs, which provide a good balance between convergence speed and stability. Further increasing the number of epochs or reducing the learning rate did not significantly improve the results in preliminary tests.

Moreover, an additional observation from the experiments is that increasing the number of SOM units and training epochs generally leads to higher classification accuracy on the MNIST dataset.

2.4 Hit Map Analysis

The hit map visualizes how input samples are distributed across the neurons in the SOM grid, where each cell represents a neuron and its color intensity indicates the number of data points mapped to that unit (i.e., the number of hits). A well-trained SOM should exhibit a relatively balanced hit distribution, where neighboring neurons receive similar numbers of samples, indicating good topological organization and clustering stability.

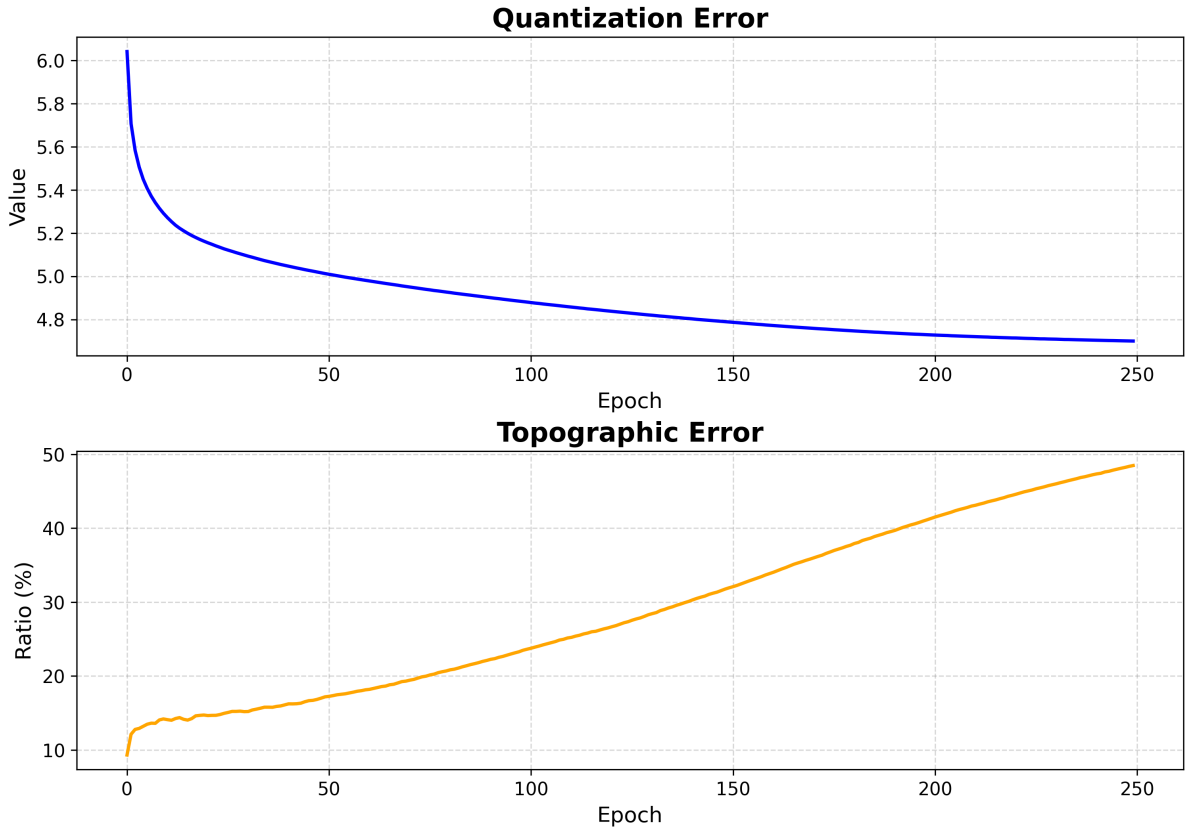


Figure 8. Training performance of the SOM model.

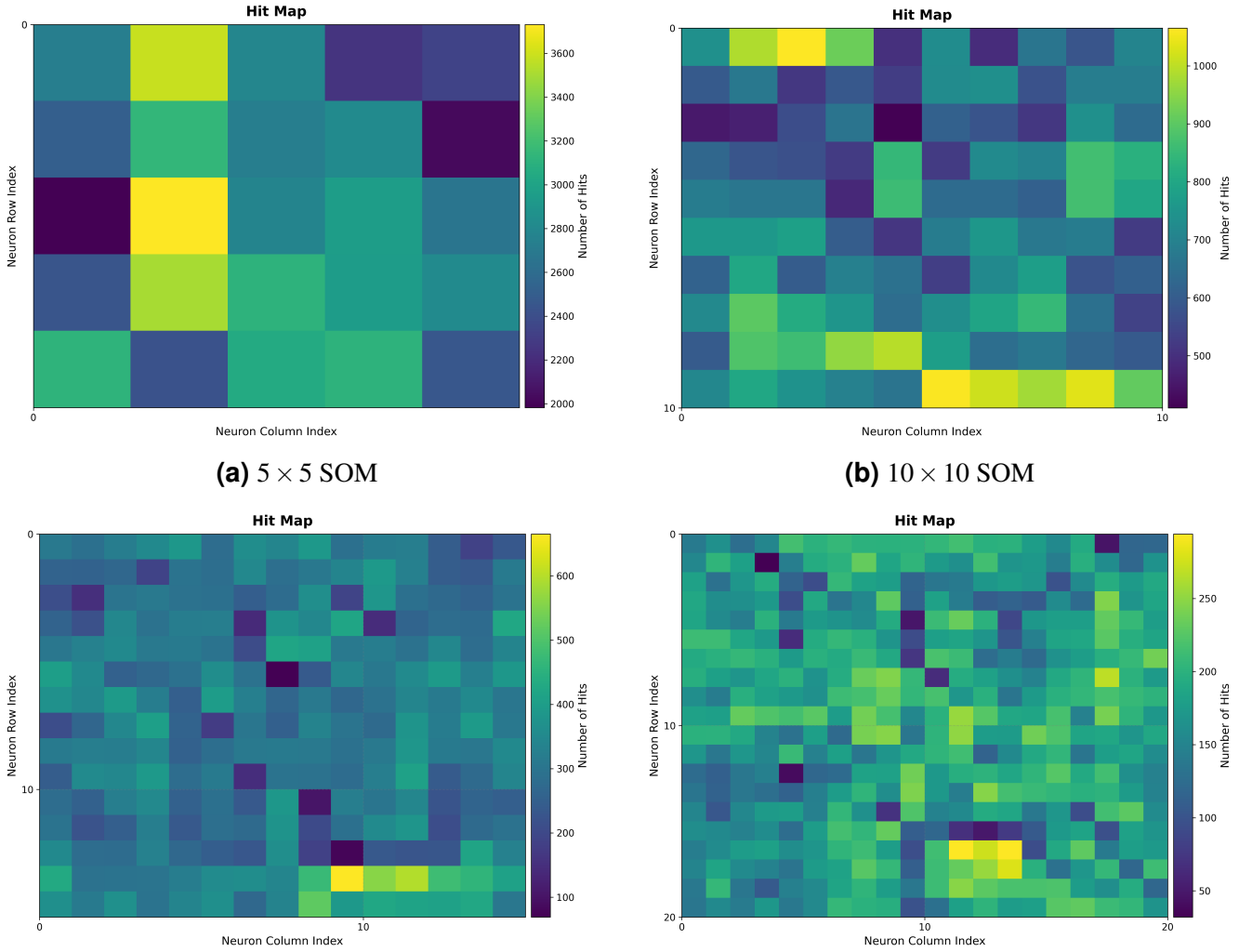
Figure 9 shows the hit maps obtained with different SOM grid sizes (5×5, 10×10, 15×15, and 20×20). When the map is small (e.g., 5×5), several neurons are overloaded while others remain inactive, suggesting limited representational capacity. As the number of units increases, the data distribution becomes more uniform, and local density variations become smoother, reflecting better preservation of the underlying manifold structure.

2.5 U-Matrix (Distance Map) Analysis

The Unified Distance Matrix (U-Matrix) visualizes the average distance between neighboring neurons in the SOM grid. It highlights cluster boundaries by showing areas of high inter-neuron distance in lighter colors and homogeneous regions in darker colors. A well-organized SOM typically presents clear boundaries between clusters and compact regions within clusters, indicating good topological preservation. Figure 10 shows the U-Matrix results for different SOM grid sizes (5×5, 10×10, 15×15, and 20×20). For smaller maps, the distance gradients are coarse and few distinct cluster boundaries can be observed, suggesting limited resolution of the feature space. As the map size increases, the U-Matrix becomes smoother and shows clearer transitions between dense and sparse regions, reflecting improved granularity and better separation of digit clusters. This observation is consistent with the hit map results, confirming that larger SOMs capture finer topological relationships within the data.

2.6 Classification Map Analysis

The classification map visualizes the most frequent class label assigned to each neuron in the SOM grid after training. Each color corresponds to one digit category, revealing how the network organizes similar



(a) 5×5 SOM

(b) 10×10 SOM

(c) 15×15 SOM

(d) 20×20 SOM

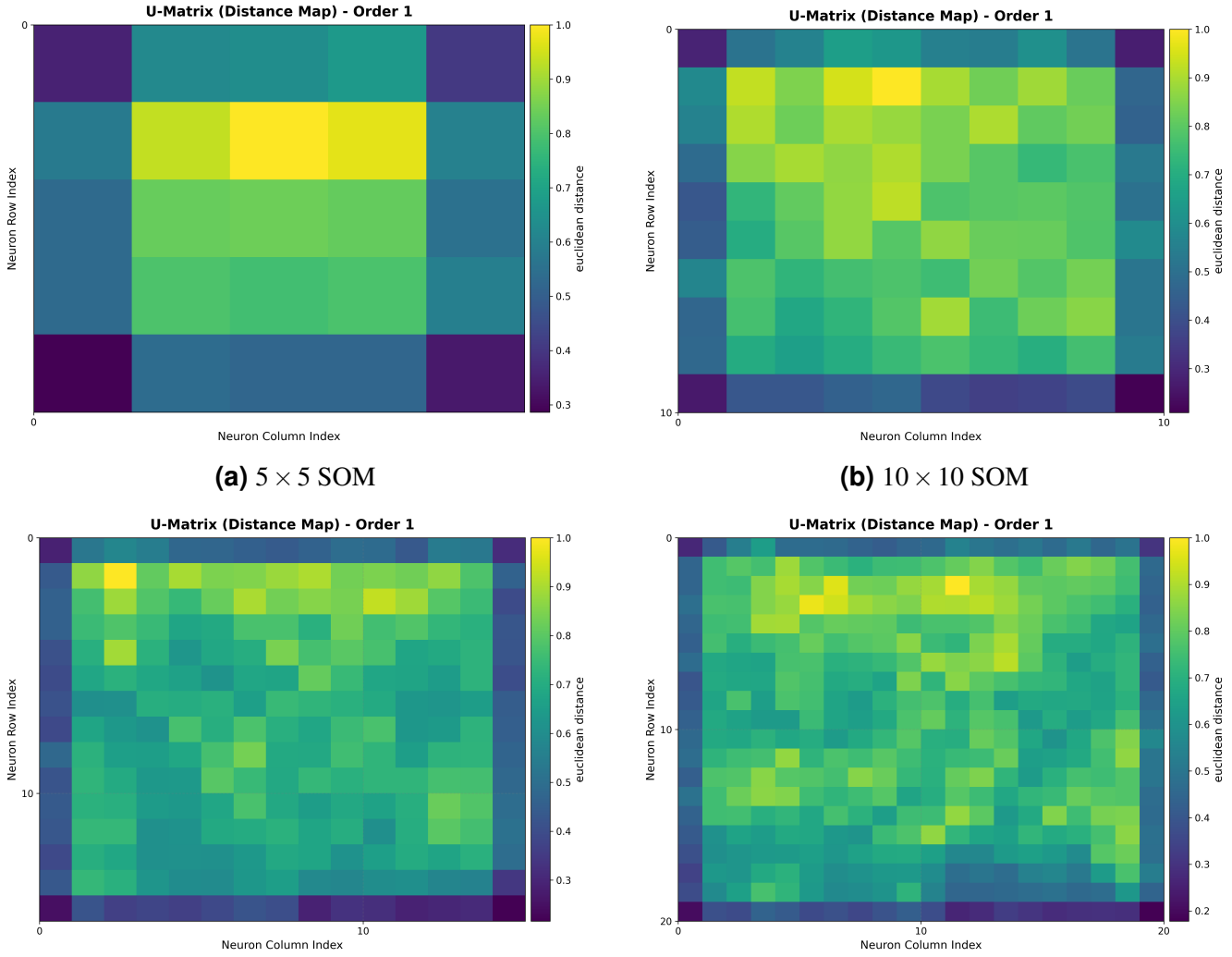
Figure 9. SOM hit maps with different grid sizes.

samples in the latent space. Ideally, neurons representing the same digit should form contiguous regions, while distinct digits should occupy separate areas, indicating good class separation.

Figure 11 shows the classification maps for different SOM grid sizes (5×5, 10×10, 15×15, and 20×20). In smaller maps (e.g., 5×5), the regions are coarse and several classes overlap, suggesting that the limited number of neurons cannot fully represent all digit categories. As the grid size increases, the map becomes more structured, with clearer boundaries between different digits and smoother intra-class transitions. Interestingly, visually or structurally similar digits (such as 4 and 9, or 3 and 8) tend to occupy neighboring regions in the map, reflecting the SOM's ability to preserve topological relationships in the input space. This improvement aligns with the reduction in quantization error and the more uniform hit distribution observed earlier, confirming that a larger SOM enhances the model's discriminative capacity and visual interpretability.

2.7 SOM Unit Distribution and Topology

To better understand the topological representation capability of SOM, Figure 12 shows the neuron (unit) positions after training with different map sizes. In these visualizations, blue points represent the data



(a) 5×5 SOM

(b) 10×10 SOM

(c) 15×15 SOM

(d) 20×20 SOM

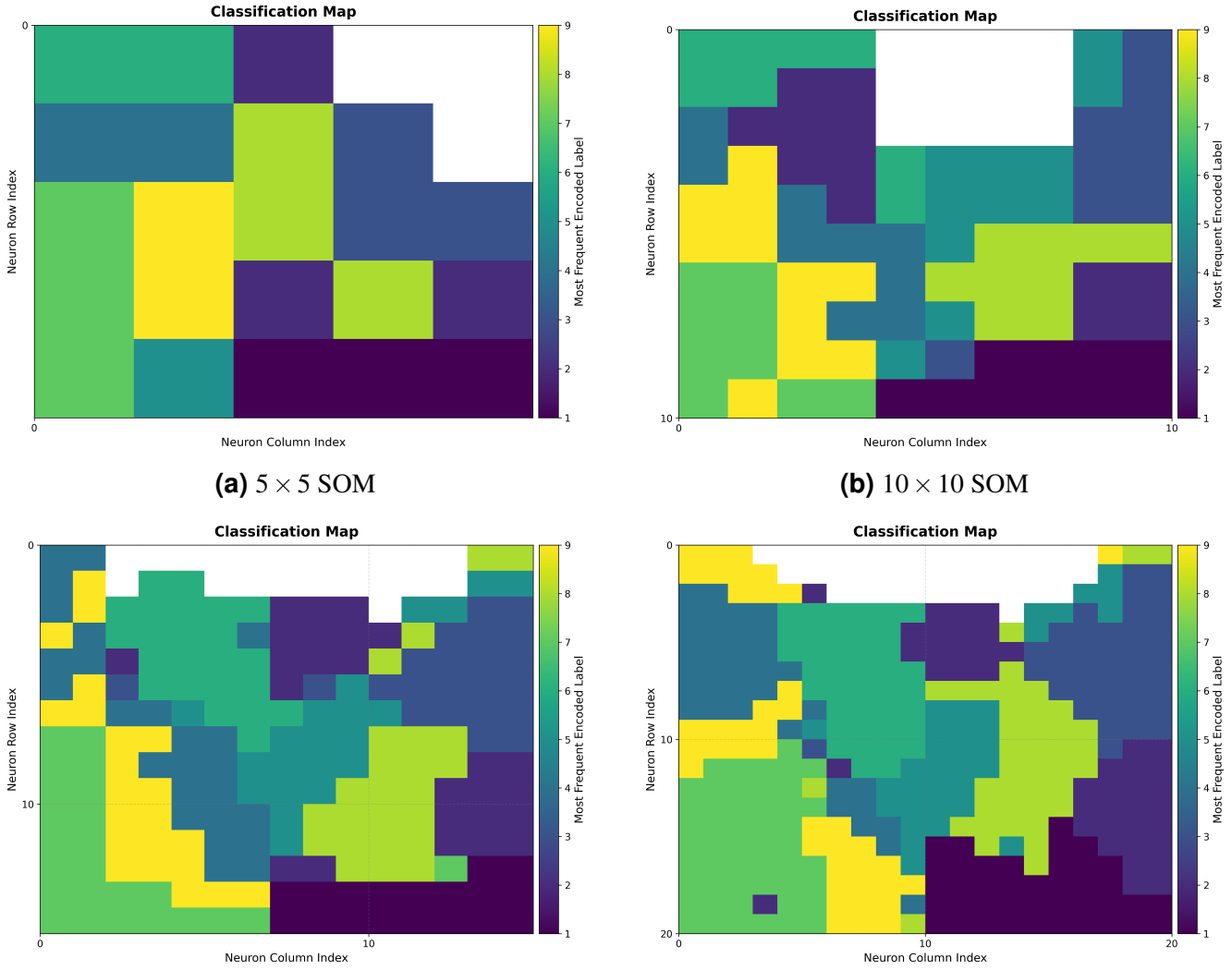
Figure 10. U-Matrix visualizations for SOMs with different grid sizes.

projected onto the first two principal components, while purple nodes denote SOM units connected by red edges to their neighboring neurons.

When the SOM contains few units (e.g., 25 or 100), the network captures only the coarse global structure of the data, leaving many regions underrepresented. As the number of units increases (e.g., 225 or 400), the neurons spread more uniformly over the data manifold, providing a finer approximation of the underlying distribution and preserving local topology more effectively. However, an excessively large number of units increases training cost and may lead to redundant nodes with limited contribution to clustering performance. This visualization further supports the earlier observation that a moderate grid size offers a good trade-off between resolution and efficiency.

2.8 Evolution of SOM Unit Positions over Different Epochs

Figure 13 illustrates the evolution of Self-Organizing Map (SOM) unit positions across three training stages (*epoch* = 10, 100, and 250). The purple points represent SOM neurons (unit weights), while blue points correspond to the input data projected into the same two-dimensional PCA space. Red lines indicate the topological connections between neighboring SOM units.



(a) 5×5 SOM

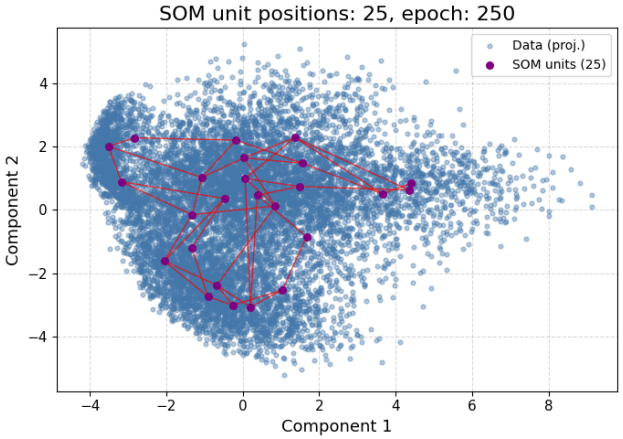
(b) 10×10 SOM

(c) 15×15 SOM

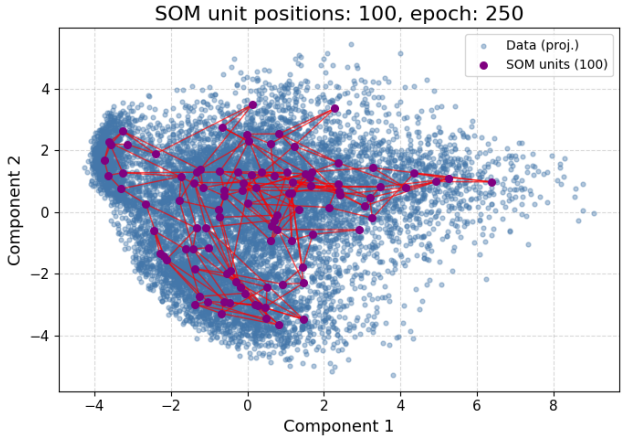
(d) 20×20 SOM

Figure 11. Classification maps of SOMs with different grid sizes.

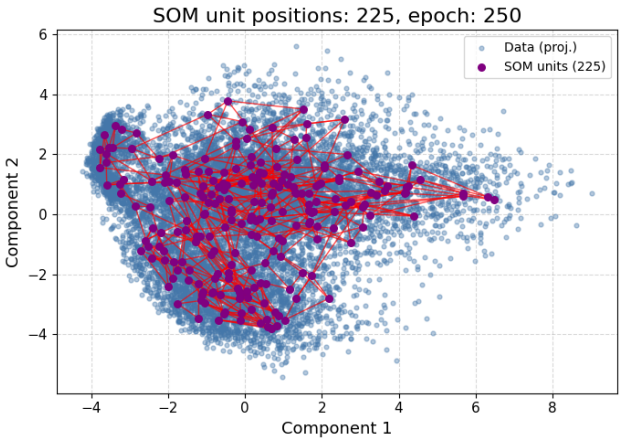
In this experiment, the SOM was **initialized using PCA**, the initial unit weights were arranged on a regular grid along the first two principal components of the data. Consequently, at the early training stage (epoch 10), the SOM forms a uniform lattice structure that progressively adapts and deforms toward the data distribution as training proceeds. By epoch 250, the map has become highly non-linear, closely following the manifold of the input data.



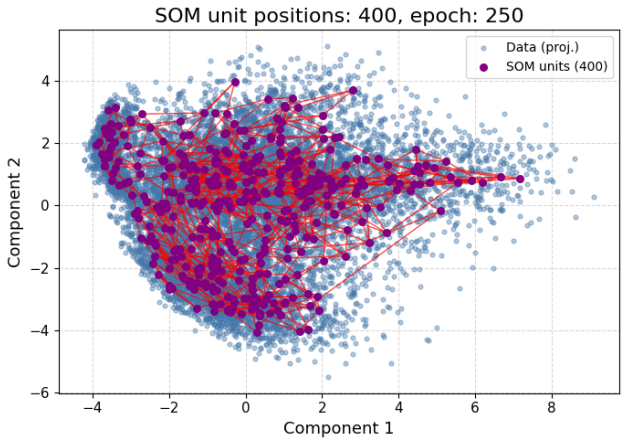
(a) 25 units (5×5)



(b) 100 units (10×10)



(c) 225 units (15×15)



(d) 400 units (20×20)

Figure 12. SOM neuron (unit) positions for different map sizes after 250 training epochs.

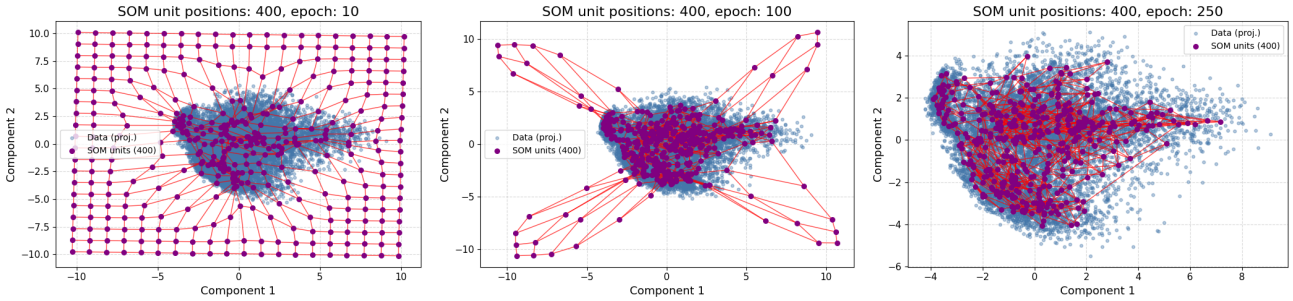


Figure 13. Evolution of SOM unit positions for 400 neurons over training epochs.