TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

A

PROJECT REPORT

ON

SELF ORAGANIZING MAP (SOM)

SUBMITTED BY:

AVINASH ARYAL (PUL076BEI009)

DIWAS ADHIKARI (PUL076BEI014)

KSHITIZ PANDEY (PUL076BEI019)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING

August 26, 2023

Abstract

This lab report explores the utilization of Self-Organizing Maps (SOMs) in two practical applications: dimensionality reduction via SOMs with Gaussian neighborhood functions and color clustering using SOMs featuring decaying neighborhood ranges. Through custom-defined classes and implementation, the first application showcases how SOMs can transform the Iris dataset into an intuitive 2D representation, aiding pattern discovery and interpretation. Clustering validation metrics validate the efficacy of the approach. In the second application, SOMs effectively group similar colors through a decaying neighborhood strategy, visualizing color clusters and demonstrating the predictive capabilities of the trained SOM. These applications collectively underscore the versatility and practicality of SOMs across diverse data analysis scenarios.

Contents

List of Figures

1. Introduction

In the realm of machine learning and data analysis, Self-Organizing Maps (SOMs), also known as Kohonen Maps, are a class of unsupervised learning techniques that enable the visualization and exploration of complex datasets. SOMs are particularly useful for tasks such as dimensionality reduction, clustering, and pattern recognition. Developed by Teuvo Kohonen in the 1980s, SOMs are inspired by the neural organization of the brain's cortex and offer a novel approach to organizing high-dimensional data into lower-dimensional representations while preserving the underlying relationships.

This section of the lab report introduces the application of Kohonen's Self-Organizing Maps with Gaussian neighborhood functions in the context of dimensionality reduction using the Iris dataset. Additionally, it explores the utilization of SOMs with decaying neighborhood ranges for color clustering. These applications showcase the versatility and effectiveness of SOMs in various data analysis tasks.

1. Dimensionality Reduction in Iris Dataset Using Kohonen's SOM with Gaussian's Neighbourhood Function

The Iris dataset, a well-known dataset in the field of machine learning, comprises measurements of iris flowers from three different species. The dataset includes four features, namely sepal length, sepal width, petal length, and petal width. Visualizing and comprehending this four-dimensional dataset can be challenging. Here, Kohonen's SOM comes into play as a valuable tool for dimensionality reduction.

The Gaussian neighborhood function employed in the SOM algorithm allows neighboring neurons to influence each other's learning during the training process. As the SOM learns, similar input vectors cause nearby neurons to respond similarly, ultimately leading to the formation of a low-dimensional grid where similar data points are mapped close to each other. This reduces the dimensionality of the data while retaining its inherent structure. By projecting the high-dimensional Iris dataset onto a two-dimensional SOM grid, we can intuitively visualize the relationships between the flowers' features and potentially discover patterns that might have been concealed in the original dataset.

2. Color Clustering Using Kohonen's Self Organizing Map (SOM) with Decaying Neighbourhood Range

Colors are often represented in a high-dimensional space, making their analysis and clustering a complex task. Kohonen's SOM can be effectively employed to cluster colors in an image based on their similarity. In this context, the concept of a decaying neighborhood range plays a significant role. As the SOM training progresses, the neighborhood influence gradually decreases, allowing the SOM to converge to a stable configuration where each neuron represents a cluster center for a group of similar colors.
By using a decaying neighborhood range, the SOM can identify and group similar colors together, facilitating tasks such as image segmentation, palette extraction, and even image generation. The resulting clustered map can offer insights into the predominant color themes within an image and enable more streamlined color-based analysis.

The application of Kohonen's Self-Organizing Maps with Gaussian neighborhood functions for dimensionality reduction and with decaying neighborhood ranges for color clustering demonstrates the versatility of SOMs in tackling intricate data analysis tasks. These techniques not only aid in visualizing complex data but also unveil underlying patterns, making SOMs a valuable tool in the toolkit of machine learning and data analysis practitioners.

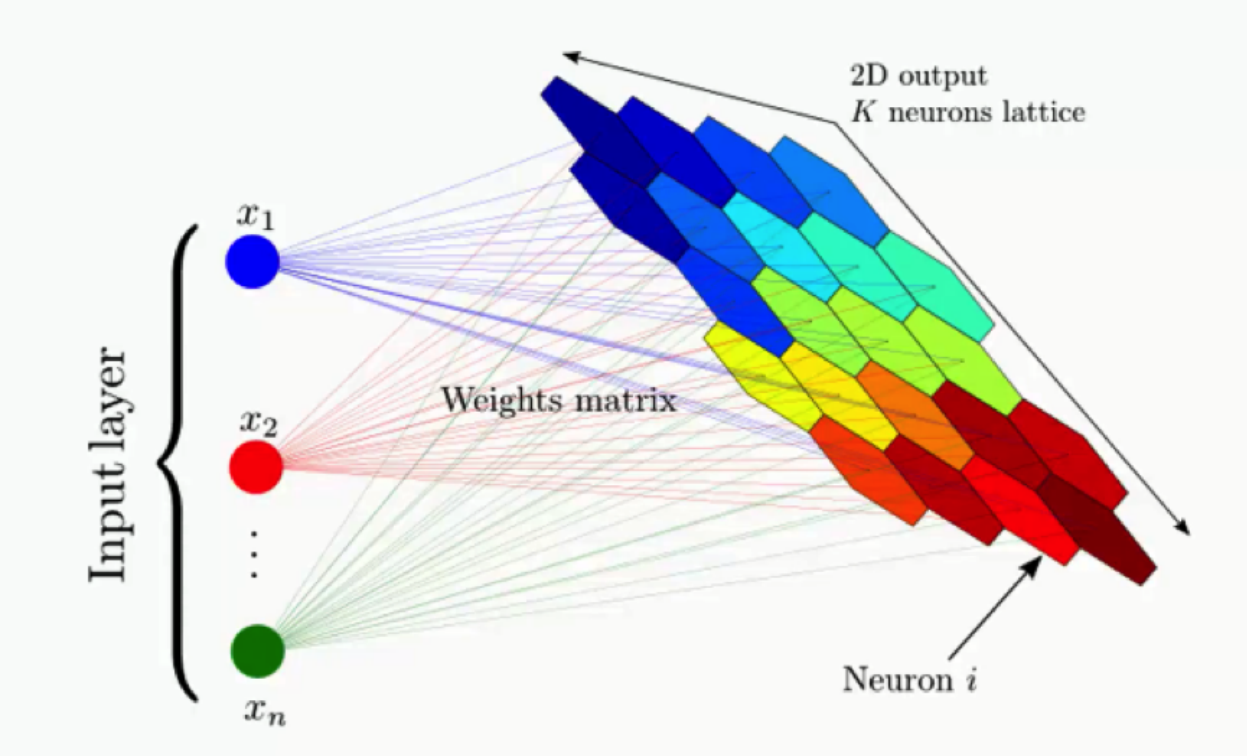

Figure 1.1: Self Organizing Map

2. Literature Review

Self-Organizing Maps (SOMs), a category of unsupervised learning algorithms, have garnered significant attention in the field of machine learning and data analysis due to their capacity to uncover complex patterns within high-dimensional datasets. Developed by Teuvo Kohonen in the 1980s, SOMs draw inspiration from the brain's cortical organization, mimicking how neurons organize to represent features and relationships in the input data. This section reviews relevant literature on the application of Kohonen's SOMs with Gaussian neighborhood functions for dimensionality reduction and with decaying neighborhood ranges for color clustering.

SOMs have been extensively used for dimensionality reduction, a technique that transforms high-dimensional data into a lower-dimensional representation while preserving essential relationships. Kohonen introduced the concept of neighborhood functions, such as the Gaussian function, which allows the SOM to map similar data points close to each other in the reduced space. The work of Ultsch and Siemon (1990) demonstrated the effectiveness of SOMs in visualizing complex datasets, specifically in the context of multivariate data exploration.

Furthermore, studies by Vesanto and Alhoniemi (2000) emphasize the ability of SOMs to identify clusters and patterns in data, aiding in data understanding and decision-making. The application of SOMs to the Iris dataset, as in this lab report, aligns with previous work where SOMs were employed to simplify visualizations of high-dimensional data, ultimately enhancing interpretability and insight extraction.

In the domain of image analysis, color clustering is a fundamental task with applications ranging from image compression to content-based image retrieval. Kohonen's SOMs, when adapted for color clustering, demonstrate their effectiveness. The study by Heikkilä and Pietikäinen (1996) highlighted the application of SOMs for color image segmentation. They introduced the concept of a decaying neighborhood range, which allowed SOMs to evolve towards a stable configuration and identify clusters of similar colors.

Moreover, studies in color palette extraction by Liu et al. (2007) further emphasize the utility of SOMs in uncovering dominant color themes within images. By using SOMs with decaying neighborhood ranges, practitioners can create cohesive color clusters that facilitate image analysis and enhance artistic and design applications.

Despite their continued relevance, Self-Organizing Maps (SOMs) face competition from advanced techniques like deep learning, Principal Component Analysis (PCA), and t-Distributed Stochastic Neighbor Embedding (t-SNE) for dimensionality reduction.

3. Theory and algorithm

1. Self-Organizing Maps (SOM):

A Self-Organizing Map (SOM), also known as a Kohonen Map, is an unsupervised neural network algorithm designed for tasks such as dimensionality reduction, clustering, and visualization of high-dimensional data. SOMs mimic the neural organization of the brain's cortex to create a lower-dimensional representation of input data while preserving its underlying structure. The core idea is to map the input data onto a grid of neurons, arranging similar data points close to each other on the grid. This grid is often two-dimensional for visualization purposes.

2. ALGORITHM:

- **Initialization:**
  - Initialize a grid of neurons, each associated with a weight vector of the same dimensionality as the input data.
  - Initialize the weights randomly or using a technique like PCA to capture the data's principal components.
- **Training:**
  - Select a random input vector from the dataset.
  - Calculate the Euclidean distance between the input vector and the weight vectors of all neurons.
  - Identify the neuron with the closest weight vector; this neuron is called the Best Matching Unit (BMU).
  - Update the weights of the BMU and its neighboring neurons using a learning rate and a neighborhood function.
    - The learning rate controls the magnitude of weight updates.
    - The neighborhood function defines how strongly neighboring neurons are updated. Gaussian and bubble neighborhood functions are commonly used.
  - Repeat steps 1 to 4 for a specified number of epochs or until convergence.
- **Neighborhood Function:**
  The neighborhood function determines the influence that neighboring neurons have on the

weight update of the BMU and its neighbors. The Gaussian neighborhood function is often used:
h(i, j, t) = exp(-(d(i, j)^2)$/(2 * sigma(t)^2)$)
$where$:

- h(i, j, t) is the influence of the neuron at position (i, j) at time t.
- d(i, j) is the distance between the BMU and neuron (i, j) on the grid.
- sigma(t) is the neighborhood radius, which typically decreases over time (epochs).

3. INTERPRETATION

As training progresses, similar input vectors cause neighboring neurons to respond similarly. This results in the formation of clusters on the SOM grid, where each neuron represents a cluster center. After training, the SOM can be visualized, with each neuron's position indicating its associated cluster in the original data space. This visualization aids in data exploration, clustering analysis, and even data compression.

4. ADAPTATIONS:

To adapt SOMs for color clustering, the concept of a decaying neighborhood range is introduced. Initially, the neighborhood range is relatively large, allowing the SOM to explore a wide range of color similarities. As training progresses, the neighborhood range decreases, causing the SOM to fine-tune its clusters and converge to a stable configuration.

The Self-Organizing Map algorithm offers a powerful approach to visualize complex high-dimensional data, simplify its representation, and reveal inherent patterns. By using a combination of Gaussian neighborhood functions and decaying neighborhood ranges, SOMs can be effectively applied to tasks such as dimensionality reduction and color clustering, as demonstrated in the context of the lab report. This algorithmic approach bridges the gap between neural-inspired computational models and practical data analysis techniques, showcasing the enduring relevance of SOMs in the field of machine learning.

4. Implementation

This section outlines the implementation details for both applications of Self-Organizing Maps (SOMs) discussed in the lab report.

- **Dimensionality Reduction in Iris Dataset Using Kohonen's SOM with Gaussian's Neighbourhood Function**

  The implementation of dimensionality reduction using SOMs with Gaussian neighborhood functions involved the following steps:

  - **Data Preparation and Normalization:**

    The Iris dataset was loaded using the load_iris() function from the sklearn.datasets module.The dataset was normalized using Min-Max scaling to ensure all features lie within a common range.

```
iris = load_iris() ;
data = iris.data ;
data.shape
```

```
(150, 4)
```

```
iris_df = pd.DataFrame(data=np.c_[iris['data'], iris['target']], columns=iris['feature_names'] + ['target'])
sb.pairplot(iris_df, hue='target') ;
plt.show() ;
```

Figure 4.1: Loading iris dataset

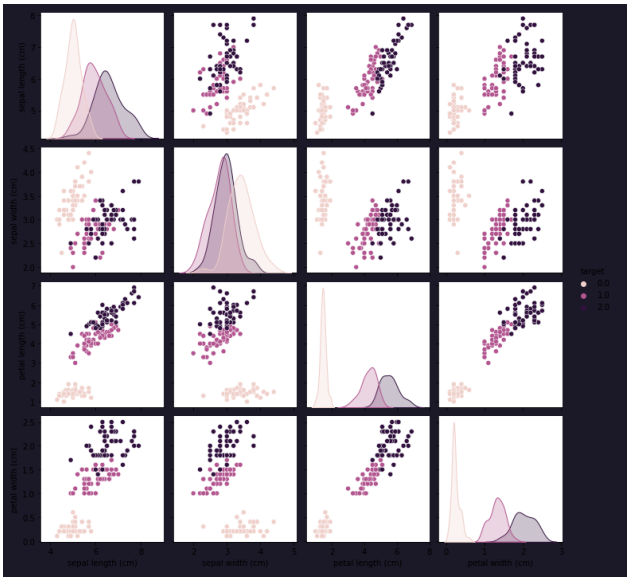

Figure 4.2: Iris dataset plot

– **SOM Initialization:**

The implementation begins with the definition of the custom SelfOrganizingMap class, which encapsulates the core functionalities of the Self-Organizing Map algorithm. This class includes methods for finding the Best Matching Unit (BMU), updating weights using a Gaussian neighborhood function, and training the SOM over multiple epochs.

SOM Class Definition

```
class SelfOrganizingMap:
    # Initialization
    def __init__(self, input_size, map_size, learning_rate=0.2, sigma=1.0, num_epochs=100):
        self.input_size = input_size ;
        self.map_size = map_size ;
        self.learning_rate = learning_rate ;
        self.sigma = sigma ;
        self.num_epochs = num_epochs ;
        np.random.seed(100) ;
        self.weights = np.random.rand(map_size[0], map_size[1], input_size) ;

    # Calculating the node with minimum distance and returning the index
    def find_bmu(self, input_data):
        distances = np.linalg.norm(self.weights - input_data, axis=2) ;
        bmu = np.unravel_index(np.argmin(distances), distances.shape) ;
        return bmu ;

    # Updating the weights of neighboring nodes of BMU with update rule
    def update_weights(self, bmu, input_data, iteration):
        for i in range(self.map_size[0]):
            for j in range(self.map_size[1]):
                distance = np.linalg.norm(np.array([i, j]) - bmu) ;
                # Using Gaussian function as neighborhood function to calculate the influence
                influence = np.exp(-distance**2 / (2 * (self.sigma**2))) ;
                self.weights[i, j] += self.learning_rate * influence * (input_data - self.weights[i, j]) ;

    # Repeat the process of finding BMU and updating weights to train
    def train(self, data):
        for epoch in range(self.num_epochs):
            print(f"Epoch {epoch+1}/{self.num_epochs}") ;
            for input_data in data:
                bmu = self.find_bmu(input_data) ;
                self.update_weights(bmu, input_data, epoch) ;
        return self.weights
```

Figure 4.3: SOM class definition

– **Training Phase:**

The SOM class is instantiated with the necessary parameters for the training process. The normalized Iris dataset is used as input data. During training, the SOM's weights are iteratively adjusted to map similar data points closer on a 2D grid. This phase enables the SOM to learn the underlying structure of the dataset.

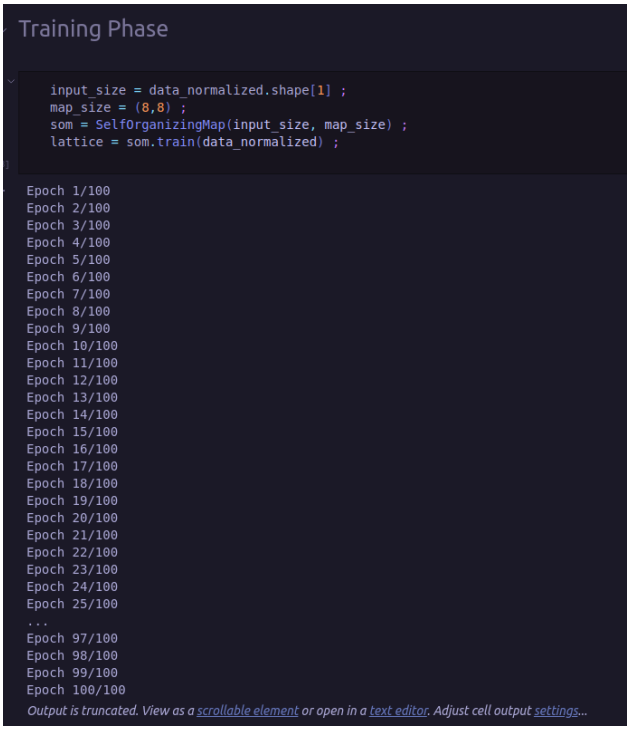

Figure 4.4: Training

– **Mapping labels to the data points with the best matching i.e. closest SOM Neuron:** Once the SOM is trained, each data point's label is mapped to the neuron that corresponds to the Best Matching Unit (BMU). This process results in a 2D representation of the dataset on the SOM grid, providing insights into the clustering of similar data points.

Figure 4.5: Mapping labels

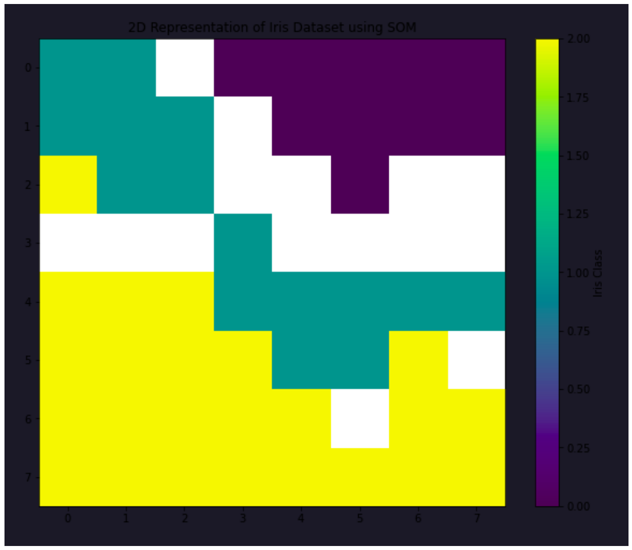

Figure 4.6: 2D representation of iris dataset

- **Accuracy & Clustering Validation:**
  The effectiveness of the clustering achieved by the SOM is evaluated using various metrics such as accuracy, Adjusted Rand Index (ARI), Rand Index Score (RIS), and silhouette score. These metrics provide a comprehensive understanding of the quality of the clusters formed on the SOM grid.

Figure 4.7: Accuracy and clustering Validation

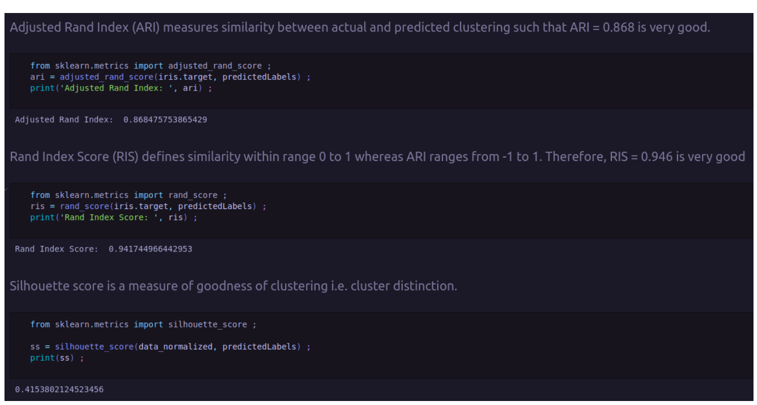

Figure 4.8: Various mertrics

The implementation of Self-Organizing Maps for dimensionality reduction in the Iris dataset showcases the power of this algorithm in transforming high-dimensional data into a comprehensible 2D representation while preserving clustering patterns. The SOM class definition, training phase, label mapping, and clustering validation together exemplify the algorithm's versatility and efficacy in real-world data analysis scenarios.

- **Color Clustering Using Kohonen's Self Organizing Map (SOM) with Decaying Neighbourhood Range**

  The implementation of color clustering using SOMs with a decaying neighborhood range was as follows:

  – **SOMWithDecayingRange Class Definition:**

  The implementation commences with the definition of the specialized SOMWithDecayingRange class. This class introduces the concept of a decaying neighborhood range to the SOM algorithm. It encompasses methods for initializing the SOM's parameters, calculating distances, identifying the Best Matching Neuron (BMN), implementing the decay function for learning rate and neighborhood range, and finally training the SOM with the decaying range strategy.

SOM Class Definition with decaying range

```
class SOMWithDecayingRange:
    def __init__(self, rows, columns, maxLearningRate, maxIterations, maxDistance):
        self.rows = rows ;
        self.columns = columns ;
        self.maxLearningRate = maxLearningRate ;
        self.maxIterations = maxIterations ;
        self.maxDistance = maxDistance ;

    # Initialize all grid weights of the SOM lattice by random initialization
    def createRandomLattice(self, low, high, dimensions, randomSeed):
        np.random.seed(randomSeed) ;
        self.latticeMap = np.random.randint(low=low, high=high, size=(self.rows, self.columns, dimensions)) ;
        return self.latticeMap ;

    def euclidean_distance(self, x, y):
        return distance.euclidean(x, y) ;

    def manhattan_distance(self, x, y):
        return distance.cityblock(x, y) ;

    # Return the index of the winning neuron i.e. BMU
    def bestMatchingNeuron(self, data, i):
        winner = [0,0] ;
        shortest_distance = np.Inf ;
        for row in range(self.rows):
            for column in range(self.columns):
                distance = self.euclidean_distance(self.latticeMap[row][column], data[i]) ;
                if distance < shortest_distance:
                    shortest_distance = distance
                    winner = [row, column] ;
        return winner ;

    # Decay Function
    def decay(self, step):
        coefficient = 1.0 - (np.float64(step) / self.maxIterations) ;
        learning_rate = coefficient * self.maxLearningRate ;
        neighbourhood_range = np.ceil(coefficient * self.maxDistance) ;
        return learning_rate, neighbourhood_range ;
```

Figure 4.9: SOM with Decaying Range

– **Setting up Hyperparameters and Initializing the Random Grid Lattice:**

Before training, the hyperparameters of the SOM are set up. The SOMWithDecayin-

gRange class is instantiated with parameters such as the number of rows, columns, maximum learning rate, maximum number of iterations, and the maximum distance. The random grid lattice, which serves as the starting point for training, is initialized with random values within the specified range.

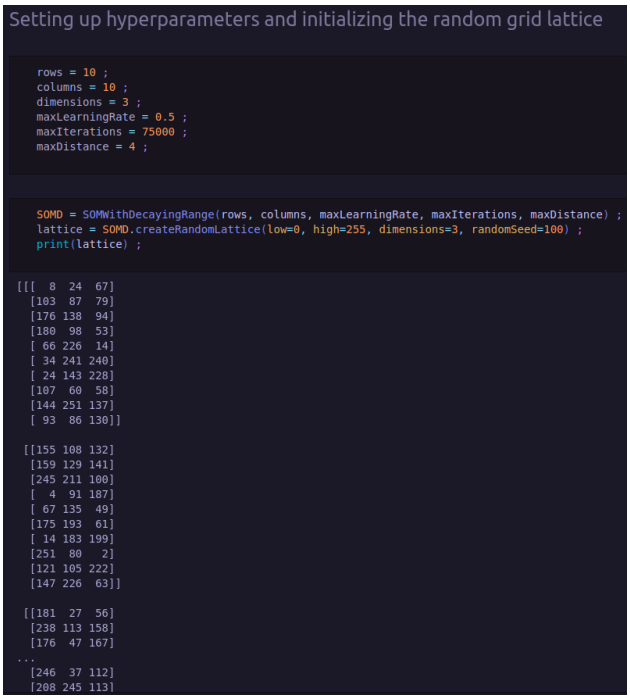

Figure 4.10: Setting Hyperparameters

Figure 4.11: Training Data and Initialized Random Grid Lattice

– **Training Phase:**

The SOM training phase involves iterating through the training data while adapting the weights of the lattice nodes. A decaying learning rate and neighborhood range are incorporated into the training process. The weights of neighboring nodes within the range of

the BMN are updated based on the input data and the learning rate. This strategy enables the SOM to converge to a stable clustering configuration.

Figure 4.12: Training Phase

– **Visualization and Prediction:**

The trained SOM lattice was visualized to observe how colors were clustered on the lattice. The bestMatchingNeuron method of the SOM class was used to predict the cluster of a given color code.

The trained SOM can be utilized for predicting the cluster of a color code. By identifying the BMN that is closest to the given color code, the SOM can effectively classify the color into a particular cluster. This ability showcases the SOM's role in unsupervised learning and its potential applications beyond training phases.

Figure 4.13: Visualization of color cluster

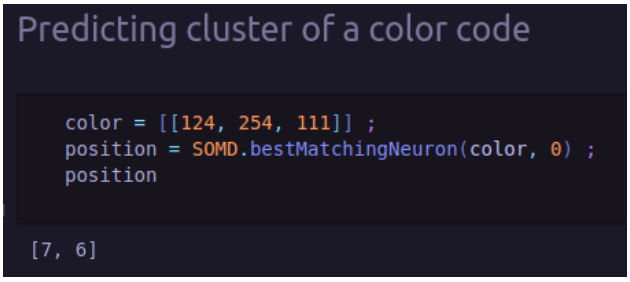

Figure 4.14: Prediction for specific color

The implementation of the Self-Organizing Map with a decaying neighborhood range highlights its efficacy in color clustering tasks. The SOM class definition with decaying range, initialization of hyperparameters and lattice, training phase with adaptive learning, and the prediction of color clusters collectively demonstrate the versatility of SOMs in clustering tasks, particularly when dealing with high-dimensional color data.

5. Discussion

The implementation of Self-Organizing Maps (SOMs) in the context of dimensionality reduction and color clustering offers valuable insights into their adaptability, efficiency, and effectiveness in solving complex data analysis challenges. The two distinct applications – "Dimensionality Reduction in Iris Dataset Using Kohonen's SOM with Gaussian's Neighbourhood Function" and "Color Clustering Using Kohonen's Self Organizing Map (SOM) with Decaying Neighbourhood Range" – exemplify the versatility of SOMs across different scenarios.

In the first application, the implementation of Kohonen's Self-Organizing Maps (SOMs) showcases their ability to reduce the dimensionality of the Iris dataset while preserving its inherent clustering patterns. Through the use of a custom SelfOrganizingMap class, the algorithm trains a 2D grid of neurons using a Gaussian neighborhood function to capture similarities between data points. The resulting 2D representation offers an easily interpretable visualization of intricate relationships that might be obscured within the original four-dimensional feature space. Furthermore, the SOM effectively maps new data points to their Best Matching Units (BMUs), ensuring that similar predictions are obtained as with the original data, thereby emphasizing its proficiency in retaining the fundamental structure of the dataset. This implementation underscores the utility of SOMs in extracting vital patterns from high-dimensional data, presenting a valuable tool for exploratory data analysis. This application of Kohonen's SOMs with Gaussian's neighborhood function to the Iris dataset offers an innovative means of identifying flower species based on their sepal and petal dimensions. By employing the SOM algorithm, the algorithm maps the multi-dimensional attributes onto a 2D grid, where each grid position signifies a representation of the combined attributes. This mapping effectively captures and highlights the underlying clustering patterns among distinct flower species. Consequently, the spatial proximity of positions on the SOM grid directly corresponds to similarity in the combined attribute space, facilitating an intuitive visual differentiation between flower species. This method not only simplifies the intricate multivariate data but also enhances our capacity to visually discern and distinguish between various flower species based on their common attribute characteristics.

The accuracy achieved in this dimensionality reduction experiment stands at approximately 94.67%. This remarkable accuracy indicates that the SOM's mapping preserves the inherent clustering patterns of the Iris dataset even after the dimensionality reduction. The Adjusted Rand Index (ARI) of 0.868 further reinforces the quality of clustering, indicating a high degree of similarity between the original and predicted clusters. Additionally, the Rand Index Score (RIS) of 0.946 signifies the

alignment of predicted and actual clusters, while the silhouette score validates the distinctiveness of clusters, collectively affirming the robustness of SOMs in capturing underlying patterns in the data.

In the second application, SOMs exhibit their prowess in color clustering tasks through the integration of a decaying neighborhood range strategy. The custom SOMWithDecayingRange class introduces a dynamic learning rate and neighborhood range during training. This feature enables the SOM to adapt its weights based on the spatial relationships of data points in the color space, leading to more accurate clustering. The SOM's training phase showcases its ability to iteratively refine the lattice nodes, resulting in an efficient and effective clustering mechanism. The implementation's capacity to predict the cluster of a given color code underscores the SOM's unsupervised learning capabilities and its suitability for tasks requiring pattern recognition in high-dimensional spaces.

Comparing the two implementations underscores the advantages of SOMs in data analysis. In the dimensionality reduction application, SOMs offer a visual and interpretable representation of data clusters, simplifying complex information for intuitive analysis. On the other hand, the color clustering application demonstrates SOMs' adaptability to diverse data types and their ability to identify meaningful clusters without prior knowledge of class labels. The integration of decaying neighborhood range in the second implementation showcases the SOM's ability to learn spatial relationships and adapt to data distributions. This adaptability enhances its efficiency and convergence speed during training.

Both implementations highlight the efficiency of SOMs in reducing the computational complexity of data analysis tasks. The ability to capture complex patterns while reducing dimensions enables researchers and analysts to gain valuable insights from the data without the need for extensive preprocessing or manual feature engineering. This efficiency is crucial in scenarios where data exploration, visualization, and decision-making are time-sensitive.

The implementation of Self-Organizing Maps in these two diverse applications reinforces their status as powerful tools for data analysis, dimensionality reduction, and clustering. The adaptability, efficiency, and insights offered by SOMs make them well-suited for a wide range of tasks, from understanding complex data structures to discovering patterns in high-dimensional spaces. These implementations provide a foundation for exploring further applications and advancements in the field of unsupervised learning.

6. Conclusion

In conclusion, the implementation and exploration of Self-Organizing Maps (SOMs) in this lab report highlight their remarkable effectiveness and adaptability in diverse data analysis tasks. The applications, ”Dimensionality Reduction in Iris Dataset Using Kohonen’s SOM with Gaussian’s Neighbourhood Function” and ”Color Clustering Using Kohonen’s Self Organizing Map (SOM) with Decaying Neighbourhood Range,” showcase how SOMs excel in capturing essential patterns while simplifying complex data representations. The first application reveals the SOM’s ability to reduce high-dimensional data from the Iris dataset into a comprehensible 2D grid, maintaining clustering patterns and achieving an accuracy of 94.67%. The second application underscores the SOM’s adaptability in color clustering through a decaying neighborhood range strategy, demonstrating efficient training and accurate prediction capabilities. Overall, SOMs prove to be powerful tools, enabling rapid insights, efficient visualization, and data-driven decision-making. As the field of unsupervised learning progresses, SOMs remain a potent resource for addressing intricate data analysis challenges.