Enhancing Cluster Cohesion: Integrating Self-Organizing Maps with K-Means Clustering for Improving Unsupervised Learning Distinctions

Ryan Park[1] and Karthik Thyagarajan[1]

Thomas Jefferson High School for Science and Technology, Alexandria, Virginia
01/08/24, Yilmaz, Quarter 2 Project

Abstract. Distinct clusters are important in data analysis because they clarify underlying patterns within datasets. Lowering cohesion metrics in clustering, such as measures that capture the average distance between data points within clusters, contributes to clearer cluster definitions. Reduced cohesion metrics imply closer proximity of data points within clusters to their centroids, meaning higher intra-cluster similarity. To improve the cohesion of clustering algorithms such as K-Means and K-means++, Self-Organizing Maps (SOMs) were looked to in order to topologically preserve the data whilst reducing them to fewer neurons. To compare the results of the SOM with other clustering techniques, both K-means and K-means++ algorithms were initially evaluated on the overall dataset. This preliminary evaluation provided a benchmark for clustering performance, allowing a comparative analysis with the SOM approach. After assessing K-means and K-means++ on the dataset, the Self-Organizing Maps algorithm was then applied. The intention was to explore how SOM, known for its capability to uncover topological structures, fared against traditional centroid-based approaches in capturing intricate data relationships.

Keywords: Clustering · Cohesion Metrics · K-Means · K-Means++ · Self-Organizing Maps.

1 Introduction

Unsupervised learning is a branch of machine learning where algorithms explore and identify inherent patterns, structures, or relationships within unlabeled data. Unlike supervised learning, this approach deals with datasets lacking explicit output labels.

In unsupervised learning, algorithms work to unveil hidden structures within data. They strive to uncover patterns or relationships based solely on input features (X) without the guidance of labeled output (Y).

Clustering: Clustering algorithms segment data into groups based on similarities among data points. Examples include K-means, DBSCAN, and hierarchical clustering.

Unsupervised learning models inherently seek patterns or structures within data without the need for labeled outputs. The training process involves discovering inherent relationships or clusters from the input features to reveal insights or provide meaningful representations of the data.

Distinct clusters are important in data analysis because they clarify underlying patterns within datasets. Lowering cohesion metrics in clustering, such as measures that capture the average distance between data points within clusters, contributes to clearer cluster definitions. Reduced cohesion metrics imply closer proximity of data points within clusters to their centroids, meaning higher intra-cluster similarity. Simultaneously, it indicates larger separations between clusters, leading to clusters with more well-defined boundaries. This reduction in cohesion metrics enables a clearer distinction between different groups within the data, enhancing the interpretability and usefulness of the clustering results in unsupervised learning scenarios.

2 Related work

In this research paper, our approach is entirely unique, as no previous works have integrated both K-Means and SOM together. While the commonality lies in the utilization of a clustering algorithm, we did not identify any other resemblances to existing literature. It is crucial to highlight that the references cited in this section diverge in terms of applications, encompassing datasets and models distinct from those explored in this paper.

2.1 SOM-KNN for Classification Tasks

Angusto and Emílio's SOM-KNN approach, integrating Self-Organizing Maps (SOM) with K-Nearest Neighbor (KNN), demonstrates strengths in classification speed and competitive recognition rates. Utilizing SOM as a pre-processing step efficiently narrows input patterns to a subset akin to the target pattern, enhancing subsequent KNN classification speed while maintaining competitive accuracy. The strengths stem from the synergistic use of SOM's clustering and KNN's decision-making based on neighboring patterns. Experimental results not only confirm SOM-KNN's effectiveness but also highlight its equivalence in recognition rates to benchmarking KNN classifiers. However, like any approach, there are weaknesses to consider, such as challenges in border regions of the SOM Map impacting classification sharpness, particularly when Best Match Unit (BMU) neurons are in these areas. Additionally, parameters like the pattern number (pn) in the SOM-KNN approach require careful consideration and may influence performance, necessitating further exploration and optimization. Despite these limitations, SOM-KNN's strengths in improved speed and competitive accuracy position it as a promising method for classification tasks, particularly in scenarios prioritizing efficiency. In terms of results, the SOM-KNN was found to perform competitively with both KNN and other models, while having decreased training times in digit recognition of car plates [2].

2.2 SOM and K-Means

Annisa Uswatun Khasanah's paper, "A Comparison Study: Clustering using Self-Organizing Map and K-means Algorithm," thoroughly assesses the performance of the Self-Organizing Map (SOM) and K-means clustering methods across three diverse datasets. The evaluation incorporates various metrics, such as percent misclassified, output visualization graphs, and Principal Component Analysis (PCA), providing a comprehensive view of the strengths and weaknesses of both algorithms. The paper's methodology is clear, and the detailed analysis of clustering outcomes for each dataset is commendable. However, a more in-depth exploration of dataset characteristics and a rationale for parameter choices would enhance the paper. Furthermore, a more extensive comparison with existing literature and the inclusion of sensitivity analysis would add robustness and contextualization to the findings. Despite these considerations, the study offers valuable insights into clustering methods. K-Means was shown to perform better than SOM as it forms centers around more dense areas [1].

Additionally, the application of artificial intelligence algorithms, specifically K-means and Self-Organizing Maps (SOM), in diagnosing spinal column issues addresses the widespread problem of back pain. The study underscores the scarcity of research in this field, attributing it to the complexities of conducting multiple medical exams for accurate vertebral issue diagnosis and the potential benign nature of some back pain causes. Employing AI algorithms, notably SOM, resulted in models with a generalization error below 10%, demonstrating the efficacy of these techniques in classification tasks. Comparative analysis using metrics such as sensitivity, specificity, precision, negative predictive value (NPV), and Cohen's Kappa index revealed the superior performance of the SOM model over the K-means model, particularly in detecting patients with vertebral problems. The study aligns the models' precision values with those reported in similar studies and by expert orthopedic physicians, bolstering the proposed approach's credibility. However, the study's weaknesses include limited details on specific dataset characteristics, potential biases, and challenges in interpreting the obtained models. Additionally, the findings' generalization may be limited by the focus on specific AI algorithms, and a more comprehensive exploration of various techniques could enhance the study's robustness. In terms of results, K-Means was shown to perform better than SOM as it forms centers around more dense areas. [4].

2.3 Hanwritten Numeral Recognition

The paper explores a novel hybrid system that integrates Support Vector Machines (SVM) with Hidden Markov Models (HMM) for handwritten numeral recognition. The aim is to address challenges related to pattern variability and distortions. The hybrid approach is commended for its improved discrimination capability, particularly emphasizing SVM's strengths in achieving higher recognition rates for isolated characters. The detailed experimental analysis conducted on the UCI Machine Learning dataset includes preprocessing, feature extraction

(employing Moment Invariants and Affine Moment Invariants), and the application of SVM with an RBF kernel. The paper acknowledges the higher memory space requirements of SVM due to support vectors and suggests potential space-saving measures. While the paper highlights SVM's superiority in recognition rates over HMM, it lacks a comprehensive comparison and analysis of the strengths and weaknesses of both methods, leaving room for a more nuanced understanding of trade-offs. The importance of parameter choices, such as C and gamma in SVM, is mentioned, but a detailed exploration of their impact or challenges in fine-tuning is needed. The paper outlines the integration of SVM with HMM for word recognition but could benefit from more insights into preprocessing, normalization steps, and potential challenges in the proposed hybrid system. Additionally, discussing the generalizability of findings to other datasets and scenarios would contribute to a more comprehensive understanding. The SVM model yielded an accuracy of 96% [5].

2.4 SOM-KNN in Anomaly Detection

In the domain of anomaly detection for health monitoring in mechanical and electronic systems, this research introduces an innovative method leveraging a self-organizing maps-based k-nearest neighbor (SOM-KNN) algorithm. The proposed approach adeptly tackles challenges associated with noisy datasets and non-convex data distribution during training. The algorithm strategically extracts best matching units (BMUs) from a self-organizing map trained on healthy data, eliminates noise-dominated BMUs, and employs k-nearest neighbor analysis for anomaly detection. The health indicator, computed through the Euclidean distance to nearest neighbors, serves as a robust quantitative measure. Experimental validation on cooling fan bearings effectively demonstrates the algorithm's prowess in monitoring system degradation, with the health indicator consistently increasing as the system deteriorates. Significantly, the method exhibits resilience to noise and demonstrates adaptability to non-convex data distributions, establishing its potential as a promising tool for anomaly detection in diverse applications within the realm of mechanical and electronic systems health monitoring [6].

3 Experiments

Effective evaluation metrics are essential in understanding the performance and suitability of clustering algorithms and evaluating cluster disparity, cohesiveness, and clarity. Each of the following metrics was used in quantifying the performance of our algorithms.

3.1 SSE (Sum of Squared Errors) Metric

The *Sum of Squared Errors* (SSE) is a common metric used in machine learning to evaluate the performance of clustering algorithms, especially when the ground

truth labels are unavailable. It quantifies the goodness of fit or the compactness of clusters.

The primary objective of SSE is to minimize the sum of squared distances between each data point in a cluster and its centroid. Given a dataset X comprising N data points $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ and K cluster centroids $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_K\}$, the SSE J_{SSE} for K clusters is computed as:

$$J_{\text{SSE}} = \sum_{i=1}^{N} \sum_{k=1}^{K} w_{ik} \|\mathbf{x}_i - \mathbf{c}_k\|^2,$$

where w_{ik} is an indicator variable denoting whether data point $\mathbf{x}_i$ belongs to cluster C_k ($w_{ik} = 1$) or not ($w_{ik} = 0$). This metric rewards tight, compact clusters by penalizing data points far from their respective centroids.

SSE serves as an internal evaluation measure for clustering algorithms. Lower SSE values generally indicate better clustering as they imply that the data points within each cluster are closer to their centroid, suggesting better cohesion of clusters.

Average SSE: The Average SSE measures the average distance between data points within a cluster and its centroid. It is calculated as:

$$\textbf{Average SSE} = \frac{1}{N} \sum_{i=1}^{N} \sum_{\mathbf{x}_i \in C} \|\mathbf{x}_i - \mathbf{c}\|^2$$

Here, N represents the total number of data points, C is a cluster, $\mathbf{x}_i$ denotes data points in cluster C, and $\mathbf{c}$ is the centroid of cluster C.

Average SSE assists in evaluating the clustering effectiveness, selecting an optimal number of clusters (K), and determining the quality of cluster assignments.

3.2 Silhouette Score

The *Silhouette Score* is a metric used to evaluate the quality of clusters produced by clustering algorithms. It quantifies the cohesion and separation of clusters.

The primary goal of the Silhouette Score is to measure how well each data point i fits within its assigned cluster compared to other clusters. It ranges from -1 to 1, where:

$$\textbf{Silhouette Score} = \frac{\mathbf{b}_i - \mathbf{a}_i}{\max(\mathbf{a}_i, \mathbf{b}_i)}$$

Here, $\mathbf{a}_i$ represents the average dissimilarity of i with other data points within the same cluster (**cohesion**), and $\mathbf{b}_i$ represents the lowest average dissimilarity of i to any other cluster (**separation**). A score near +1 indicates well-separated clusters, while scores near 0 suggest overlapping clusters and negative scores imply incorrect clustering assignments.

Computing Silhouette Score:

1. For each data point i:
   (a) Calculate $\mathbf{a}_i$, the average distance between i and other points within the same cluster.
   (b) Calculate $\mathbf{b}_i$, the average distance of i to points in the nearest neighboring cluster.
2. Compute the Silhouette Score for each data point using the formula.
3. Calculate the overall Silhouette Score:

$$\textbf{Average Silhouette Score} = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathbf{b}_i - \mathbf{a}_i}{\max(\mathbf{a}_i, \mathbf{b}_i)}$$

Higher Silhouette Scores indicate better-defined clusters with distinct boundaries and good internal cohesion. It aids in determining the optimal number of clusters (K) by comparing scores for different K values.

3.3 Dataset

Our algorithms were run and tested on the **optdigits.csv** dataset. The dataset is one of grayscale, handwritten digits on an 8x8 image, flattened to be a dataset with 64 features. The dataset contains 3823 samples with class labels from 0-9 corresponding to the digits 0-9 for each datapoint. The data was found in the UCI Machine Learning Repository and contained no missing values. Thus, preprocessing only included z-score normalization.

4 Methods

To compare the results of the Self-Organizing Maps (SOM) with other clustering techniques, both K-means and K-means++ algorithms were initially evaluated on the overall dataset. This preliminary evaluation provided a benchmark for clustering performance, allowing a comparative analysis with the SOM approach. After assessing K-means and K-means++ on the dataset, the Self-Organizing Maps algorithm was then applied. The intention was to explore how SOM, known for its capability to uncover topological structures, fared against traditional centroid-based approaches in capturing intricate data relationships.

4.1 K-Means

K-means, a well-known centroid-based clustering algorithm, is known for its simplicity, efficiency, and widespread applicability. This algorithm aims to partition a dataset into K clusters by optimizing a well-defined objective function, emphasizing the minimization of the intra-cluster variance.

The optimization objective involves minimizing the sum of squared Euclidean distances between data points and their assigned cluster centroids. For a dataset

$\mathbf{X}$ consisting of N data points $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ in a D-dimensional space, and K cluster centroids $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_K\}$, the objective function J is defined as:

$$J = \sum_{i=1}^{N} \sum_{k=1}^{K} w_{ik} \|\mathbf{x}_i - \mathbf{c}_k\|^2,$$

where w_{ik} is an indicator variable denoting whether data point $\mathbf{x}_i$ belongs to cluster C_k ($w_{ik} = 1$) or not ($w_{ik} = 0$). This binary indicator ensures that each data point is assigned to exactly one cluster.

K-Means Algorithm Steps: The K-means algorithm iteratively refines its cluster assignments and centroids through two fundamental steps:

1. **Assignment Step:** Assign each data point to the cluster whose centroid is closest. Mathematically, for each data point $\mathbf{x}_i$, the assignment is determined by:

$$w_{ik} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_i - \mathbf{c}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

   This step minimizes the within-cluster sum of squared distances.

2. **Update Step:** Recalculate the cluster centroids based on the newly assigned data points. The centroid of cluster C_k is updated as:

$$\mathbf{c}_k = \frac{1}{|C_k|} \sum_{\mathbf{x}_i \in C_k} \mathbf{x}_i.$$

   This step redefines the cluster centroids, ensuring convergence toward the optimal configuration.

K-means is sensitive to initial centroid placements, often requiring multiple random initializations to avoid convergence to local minima. Additionally, convergence is achieved when there is no change in cluster assignments or centroids, though the algorithm may not always converge to the global minimum due to its susceptibility to initialization conditions.

K-means finds utility in diverse domains, including image segmentation, customer segmentation, and anomaly detection. However, its effectiveness can be compromised when clusters exhibit irregular shapes or varying sizes, as K-means implicitly assumes clusters to be spherical and of comparable sizes.

4.2 K-Means++

K-means++, an improvement upon K-means, addresses the sensitivity of K-means to initial centroid placements. This improved approach refines the initialization step by introducing a probabilistic method for choosing the initial centroids, enhancing convergence, and mitigating the risk of local minima.

The objective function for K-means++ remains the same as the K-means algorithm.

K-means++ differs from K-means in the initialization phase. The steps are as follows:

1. **First Centroid:** Select the first centroid uniformly at random from the data points: $\mathbf{c}_1 \sim \text{Uniform}(\mathbf{X})$.
2. **Subsequent Centroids:** For each subsequent centroid j up to K, select the next centroid from the data points with probability proportional to the squared distance from the point to the nearest existing centroid:

$$\mathbf{c}_j \sim \text{Prob}(\mathbf{x}) = \frac{D(\mathbf{x})^2}{\sum_{\mathbf{x}' \in \mathbf{X}} D(\mathbf{x}')^2},$$

where $D(\mathbf{x})$ represents the distance from $\mathbf{x}$ to the nearest existing centroid.

Once initialized, K-means++ follows the same iterative assignment and update steps as the classic K-means algorithm.

K-means++ provides a more robust initialization strategy compared to random seeding, reducing the likelihood of converging to suboptimal solutions. Its effectiveness is particularly pronounced when K is relatively large. However, it does not entirely eliminate the sensitivity to the choice of K and may still converge to local minima in certain scenarios.

4.3 Self-Organizing Maps (SOMs)

Self-Organizing Maps (SOMs), are different traditional clustering approaches by preserving the topological structure of high-dimensional input data. Introduced by Kohonen, SOMs employ competitive learning to map input space onto a lower-dimensional grid, revealing intrinsic relationships between data points.

In competitive learning, neurons in the SOM grid compete to represent specific regions of the input space. Each neuron has a weight vector, and during training, the neuron with the weight vector most similar to the input data point is declared the winner. The weights of the winning neuron and its neighbors are then adjusted.

One distinguishing feature of SOMs is their ability to maintain the topological relationships between data points. Neurons positioned close in the SOM grid respond similarly to similar input patterns, ensuring that the spatial arrangement reflects the data's inherent structure.

SOMs minimize the quantization error, representing the discrepancy between the input data point and the weight vector of the winning neuron. For a dataset $\mathbf{X}$ with N data points $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, and a SOM grid with dimensions $M \times N$, the objective function J is defined as:

$$J = \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{w}_{\text{BMU}(\mathbf{x}_i)}\|^2,$$

where $\mathbf{w}_{\text{BMU}(\mathbf{x}_i)}$ is the weight vector of the Best Matching Unit (BMU), i.e., the winning neuron for the input data point $\mathbf{x}_i$.

SOM Algorithm Steps:

1. **Initialization:** Initialize the SOM grid with random weights.
2. **Training Iterations:** For each iteration, randomly select an input data point and find the BMU. Adjust the weights of the BMU and its neighbors based on a neighborhood function.
3. **Convergence:** Repeat the training iterations until the SOM reaches a stable configuration.

SOMs find applications in various fields, including data visualization, feature extraction, and clustering. They are particularly useful for high-dimensional datasets. However, SOMs require careful tuning of parameters such as learning rate and neighborhood size. The grid topology and convergence criteria also impact the final representation.

5 Results

The SOM algorithm creates a map matching the data's topological distribution.

5.1 Hyperparameters

A 30x30 grid was chosen for our SOM, along with a linearly decaying learning rate starting at 0.05 and a Manhattan Distance neighborhood function starting at a distance of 4 before reducing stepwise. The value of k for both clustering algorithms was chosen to be 10, as there are 9 possible digits.

Visual representations of the SOM are shown below. Calculating the label for each neuron was accomplished using the following steps:

1. Calculate the BMU for a datapoint and store the point's label in a list associated with the selected neuron. Repeat for every datapoint in the dataset
2. Assign the argmax of the list as the label for that neuron in the SOM. Those with an empty list are labeled as unclassified.

As seen in Figure 1, each of the nine digits in the dataset corresponds to a color, while -1 (dark blue) corresponds to an unclassifiable neuron. The boundaries between classes are relatively clear. The distribution is roughly circular, suggesting that the data may have a roughly spherical shape in high-dimensional space.

Since each neuron has 64 dimensions, tSNE was used to visualize the distribution of the grid in a 2D space, as seen in Figure 2. tSNE is similar to PCA, except it can visualize nonlinear distribution boundaries. It calculates similarities between points and places them on a normal distribution before using gradient

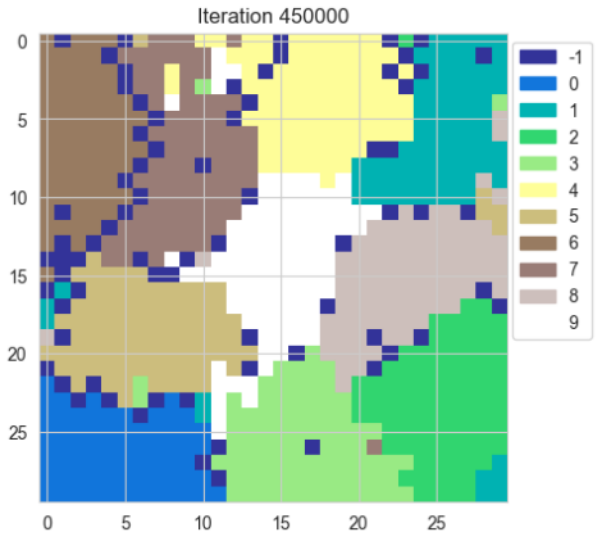

Fig. 1: A visual representation of the SOM grid after 144500 iterations.

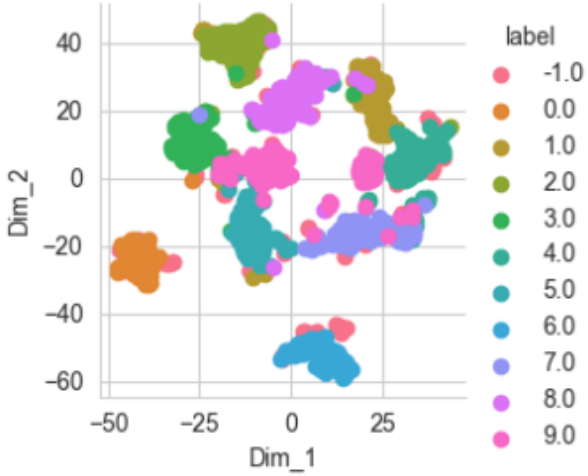

Fig. 2: A lower dimensional visualization of the SOM neurons using the tSNE algorithm.

descent to find the optimal distances between similar/dissimilar points. The details of this algorithm are beyond the scope of this report.

The labels assigned to the SOM have no value to the clustering algorithms and are merely included for visualization purposes.

Both K-Means and K-Means++ were run on the dataset before and after applying the SOM. After finding the centers, the SSE, Average SSE, and Silhouette Score were computed for each. The results are displayed in Table 1.

Table 1: Clustering Performance Before and After Applying SOM

	K-Means	K-Means++	K-Means With SOM	K-Means++ With SOM
SSE	2.491×10^6	2.492×10^6	5.691×10^3	5.721×10^3
Average SSE	651.716	651.715	19.695	19.795
Silhouette Score	0.198	0.198	0.331	0.279

6 Discussion

The impact of clustering algorithms incorporating Self-Organizing Maps (SOM) on the Sum of Squared Errors (SSE) becomes apparent in Table 1, where the average SSE is notably influenced. Moreover, the Silhouette Score, calculated across a range of values from 1 to 12, consistently peaked at 10, aligning with the number of labels in the dataset. The introduction of SOM to both K-Means and K-Means++ resulted in a significant upswing in the silhouette score. In comparison, K-Means with SOM demonstrated superior performance compared to K-Means++ with SOM. We attribute this distinction to the fact that the K-Means++ initialization algorithm is designed to position centers farthest away from each other. However, within SOM, the clusters exhibit closer proximity, diminishing the impact of the initialization algorithm, particularly when considering the large number of points in the original dataset used by K-Means++.

In terms of overall performance metrics, the K-Means algorithm with SOM exhibited superior clustering performance. This is likely due to the closer proximity of the centers within the machine learning model facilitated by SOM, resulting in denser and more compact clusters. The enhanced performance is also attributed to the increased distinctiveness of clusters, facilitating clearer grouping of other data points. This efficacy is achievable through SOM's ability to reduce the dataset while preserving its diversity and correlation with the dataset labels. There was no problem in overfitting the training set because the K-Means and K-Means++ algorithms rely on using the whole dataset as both the training and testing dataset.

In Figure 1, a two-dimensional representation of a high-dimensional dataset is presented on a map, where data points are color-coded based on their positions. The map exhibits a roughly circular shape, and the concentration of data points increases towards the center, hinting at a potential spherical distribution in the high-dimensional space. Distinct clusters of data points are discernible on the map, implying inherent groupings within the dataset. Notably, the size of data points correlates with their distance from the center, indicating a potential relationship where outliers or anomalies may be more prevalent among points situated farther from the central region.

7 Conclusion

We researched an analysis of the K-Means and K-Means++ clustering algorithms with and without SOMs. The goal was to battle the computationally inefficient processes present in clustering algorithms. K-Means with SOM was found to perform the best in terms of our clustering metrics, which included SSE, Average SSE, and Silhouette Score. The values were 5691.993598618154, 19.69547957999361, and 0.3311073255139434, respectively. We believe that K-Means consistently outperformed K-Means++ with and without SOM on the Optdigits Dataset because the K-Means initialization algorithm seeks to place the centers farthest apart when in fact they are closer in this dataset because of the large number of centers needed to be found.

For our future work, we aim to investigate the effectiveness of incorporating Self-Organizing Maps (SOMs) into more intricate classification tasks and diverse neural network architectures. Our motivation stems from the notable success we observed when combining SOMs with K-Nearest Neighbors (KNN), which substantially reduced training time. We aspire to extend this approach to various neural network models, examining its impact on training efficiency while maintaining high accuracy levels.

References

1. A. U. Khasanah, "A Comparison Study: Clustering using Self-Organizing Map and K-means Algorithm," *Performa*, vol. 15, no. 1, 2016, doi: 10.20961/performa.15.1.13754.
2. L. Augusto and E. Del-Moral-Hernandez, "A SOM Combined with KNN for Classification Task," in *2011 International Joint Conference on Neural Networks (IJCNN)*, 2011, pp. 2054–2059, doi: 10.1109/IJCNN.2011.6033525.
3. "Error Sum of Squares," Stanford University, 2024. [Online]. Available: `https://hlab.stanford.edu/brian/error_sum_of_squares.html`
4. N. A. Melo Riveros, B. A. Cardenas Espitia, and L. E. Aparicio Pico, "Comparison between k-means and self-organizing maps algorithms used for diagnosis spinal column patients," *Informatics in Medicine Unlocked*, vol. 16, pp. 100206, 2019, doi: 10.1016/j.imu.2019.100206.
5. A. Sharma, "Handwritten Digit Recognition using Support Vector Machine," 2012. [Online]. Available: `https://arxiv.org/pdf/1203.3847.pdf`
6. J. Tian, M. H. Azarian, and M. Pecht, "Anomaly Detection Using Self-Organizing Maps-Based K-Nearest Neighbor Algorithm," in *PHM Society European Conference*, vol. 2, no. 1, 2014, doi: 10.36001/phme.2014.v2i1.1554.
7. Van Hulle, M. M. (2012). Self-organizing Maps. Handbook of Natural Computing, 1, 585-622.