Handwritten Digit Classification using K-Means Clustering and Statistical Outlier Detection

Derek Nester
Department of Computer Engineering
Washington University in St. Louis
St. Louis, USA
derek.nester@wustl.edu

Paul Pham
Department of Electrical and Systems Engineering
Washington University in St. Louis
St. Louis, USA
p.h.pham@wustl.edu

***Abstract*—This report details the design and implementation of a system to classify handwritten digits using the K-Means clustering algorithm [1]. The primary objective is to group 784-dimensional vector representations of handwritten digit images from the MNIST database into distinct clusters and then use these clusters to classify new unseen digits [2]. The optimal number of clusters, k, was determined to be 18 using the elbow method, which balances the complexity of the model with the clustering error. The K-Means algorithm was implemented from first principles in MATLAB to generate k centroids from a training dataset of 1500 images. Each resulting centroid was automatically assigned a numerical label (0-9) based on the mode of the labels of the training images within its cluster. Furthermore, a statistical outlier detection strategy was developed by defining a distance threshold for each cluster, calculated as the mean plus two standard deviations of the intra-cluster distances from the training data. The classifier performance was evaluated on a separate test set of 200 images, achieving a high classification accuracy of 71.50% and identified 16 data points as outliers. This work demonstrates the efficacy of unsupervised clustering in solving complex classification problems.**

I. Introduction

The ability to automatically recognize and classify handwritten digits is a fundamental problem in the field of pattern recognition and machine learning, with applications ranging from automated mail sorting to bank check processing [2]. The core challenge lies in the immense variability of human handwriting. This case study explores a solution to this problem using K-Means clustering, an unsupervised learning algorithm [1]. Unlike supervised methods that require labels for all training data to learn a mapping function, clustering aims to discover inherent groupings within the data itself.

The MNIST dataset, while a benchmark, presents unique challenges for clustering algorithms. Variations in stroke thickness, slant, and positioning for the same digit mean that clusters are not always spherically distributed in the high-dimensional feature space. K-Means, which operates on geometric distance, can struggle with these variations. However, its simplicity and efficient nature make it an excellent baseline model for understanding the inherent structure of the data before applying more complex classification techniques.

The goal is to build a classifier that takes an image of a handwritten digit and determines the number it represents (0 through 9). The project involves several key stages in order to be successful:

- Implementing the K-Means clustering algorithm from the ground up in MATLAB.
- Developing a method to determine an optimal value for k, the number of clusters.
- Training the model by generating k representative centroids from a labeled training set.
- Designing a strategy to assign a numerical meaning (0-9) to each centroid.
- Developing a robust method for identifying outliers within a test dataset.
- Evaluating the performance of the final classifier and outlier detection system on a separate, unseen test set.

This report details the methods used in each stage, presents the results of the trained classifier, and discusses the implications, limitations, and future possibilities of the design.

II. Methods

The classification system was developed through a multistage process involving data preparation, optimal cluster selection, K-Means algorithm implementation, and the design of classifier and outlier detection logic.

A. Data Set and Preparation

The project utilizes the MNIST database of handwritten digits [2]. The data is provided in two sets: a training set of 1500 images and a test set of 200 images. This separation is crucial for model validation; the model is trained exclusively on the training set, and its performance is evaluated on the completely unseen test set to provide an unbiased measure of its generalization capabilities. Each image is a grayscale 28×28 pixel grid, which is flattened into a 784-dimensional row vector. Each element of the vector is an integer between 0 and 255, representing pixel intensity. For the training process, the provided label column is repurposed to store the assigned cluster index for each image. No further data normalization or scaling was performed, as the pixel values already existed within a consistent range (0-255). An example from the dataset is shown in Figure 1.

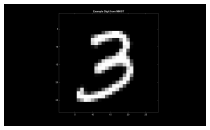

Fig. 1. An example of a handwritten digit '3' from the MNIST dataset after being reshaped into a 28×28 image.

B. Determining the Optimal Number of Clusters (k)

The performance of the K-Means algorithm is highly dependent on the choice of k. To find an optimal value, the Elbow Method was implemented. This method involves running the K-Means algorithm for a range of k values (from 10 to 25) and calculating the final cost, or within-cluster sum of squares (WCSS), for each.

$$\text{Cost (WCSS)} = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \mu_i||^2 \tag{1}$$

where C_i is the i-th cluster and μ_i is its centroid. The "elbow point" on the plot of Cost vs. k represents a good trade-off between minimizing the cost and avoiding an overly complex model. This point, shown in Fig. 2, was identified algorithmically by finding the point on the curve with the maximum perpendicular distance to a line connecting the first and last points.

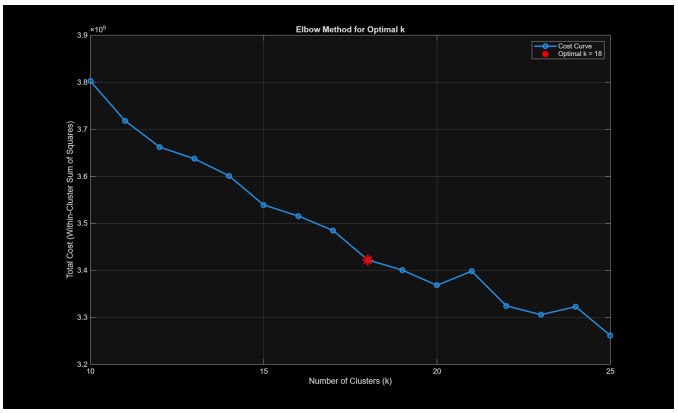

Fig. 2. The Elbow Method plot showing the total cost (WCSS) for different values of k. The red star indicates the optimal k-value of 18, which is the point of maximum curvature, or the "elbow."

C. K-Means Algorithm Implementation

The core clustering logic was implemented following an iterative procedure for 50 iterations:

1) **Initialization:** A value of $k = 18$ was selected for the number of clusters. While the elbow plot analysis algorithmically identifies a point of maximum curvature, a visual inspection reveals that the clustering cost continues to decrease meaningfully with a higher number of clusters. Choosing $k = 18$ over a smaller value is a deliberate design choice aimed at allowing the model to capture more subtle variations in handwriting styles for the same digit (e.g., separate clusters for a "1" written as a vertical line versus one with a top serif). The $k = 18$ centroids were initialized using the **Forgy method**, where 18 unique data points are randomly selected from the training set to serve as the initial cluster centers. This method is computationally efficient and ensures the initial centroids are valid points within the data's feature space.
2) **Assignment Step:** Each training data vector $\mathbf{x}$ is assigned to the closest centroid μ_i by minimizing the Euclidean distance $||\mathbf{x} - \mu_i||_2$. This step partitions the entire dataset into k clusters.
3) **Update Step:** The position of each centroid μ_i is recalculated as the arithmetic mean of all data vectors assigned to its cluster C_i. This moves the centroid to the geometric center of its corresponding data points previously assigned.

$$\mu_i = \frac{1}{|C_i|} \sum_{\mathbf{x} \in C_i} \mathbf{x} \tag{2}$$

These steps are repeated until the centroids converge, which is monitored by tracking the cost at each iteration. If a cluster becomes empty during an update, its centroid is re-initialized to another random point from the dataset to prevent it from being lost.

D. Classifier and Outlier Detection Design

After 50 iterations, the final centroids are used to build the classifier.

a) Classifier Design: Each of the 18 centroids is assigned a numerical label (0-9) by finding the **mode** of the true labels of all training vectors in its cluster. This democratic approach is robust, as it leverages the collective identity of the cluster's members to define the centroid's label. It is more resilient to a few mis-assigned training points within a cluster than an alternative approach, such as finding the single nearest training point and adopting its label.

b) Outlier Detection Strategy: A statistical approach was designed to identify outliers. A unique distance threshold T_i is established for each cluster:

$$T_i = \mu_{dist,i} + 2\sigma_{dist,i} \tag{3}$$

where $\mu_{dist,i}$ and $\sigma_{dist,i}$ are the mean and standard deviation of the Euclidean distances of all training points to their assigned cluster centroid i. This threshold creates a hypersphere around each centroid. The choice of two standard deviations is based on the empirical rule for normal distributions, which suggests that approximately 95% of the data in a given cluster will fall within this boundary. A stricter threshold (e.g., one standard deviation) would flag more points as outliers, while a more lenient threshold (e.g., three standard deviations) would be less sensitive. During testing, if a new image's distance to its assigned centroid exceeds that centroid's unique threshold T_i, it is flagged as an outlier.

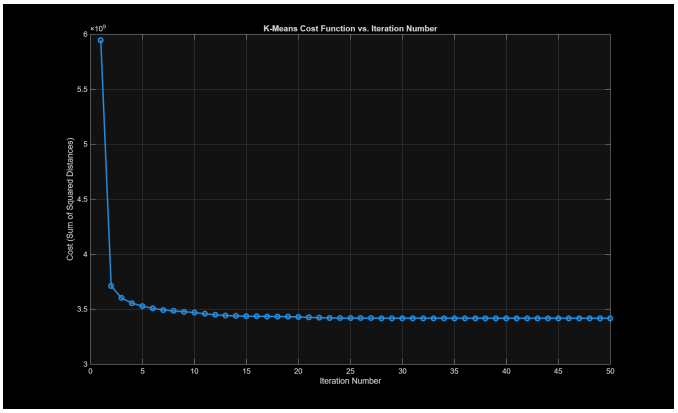

Fig. 3. The K-Means cost function plotted against the iteration number for k=18. The cost stabilizes, indicating convergence.

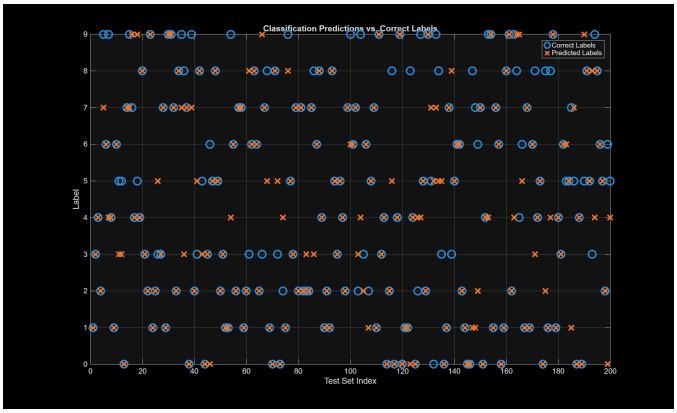

Fig. 5. Comparison of correct labels ('o') and predicted labels ('x') for the 200-image test set. Overlapping markers indicate a correct classification.

III. Results and Discussion

The system was trained on the 1500-image dataset and evaluated on the 200-image test set.

A. Algorithm Convergence and Final Centroids

The K-Means algorithm demonstrated rapid convergence, as shown in Figure 3, with the cost function plateauing after about 10 iterations. The steep initial drop signifies that the randomly placed centroids quickly moved to the centers of dense regions of data, and the subsequent flattening indicates that the cluster assignments became stable.

The final 18 automatically labeled centroids are visualized in Figure 4. Several digits are represented by multiple centroids, capturing variations in handwriting. This diversity allows the classifier to recognize multiple styles of the same digit. For instance, there are separate centroids for a slanted '1' and an upright '1', which enhances the model's flexibility.

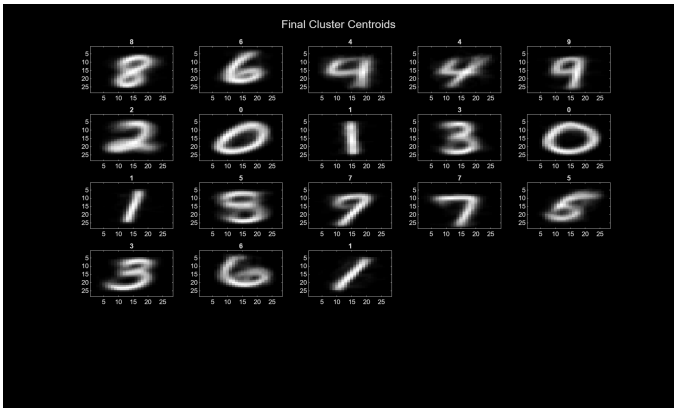

Fig. 4. The 18 final centroids visualized as 28×28 images. The title of each subplot shows the numerical label assigned via the mode calculation.

B. Classification and Outlier Performance

The classifier achieved an accuracy of **71.50%**, correctly identifying 143 out of the 200 test images (Figure 5).

The statistical thresholding method identified 16 images in the test set as outliers, as shown in Figure 6. This validates the design choice of creating cluster-specific thresholds. A single global threshold would have been less effective, as some clusters (like for the digit '1') are naturally tight and have a small standard deviation of distances, while others (like for '8') are more varied. However, identifying 16 outliers is likely more than the true number of anomalous digits in the test set. This suggests that the model's thresholds may be too strict, occasionally flagging valid but unusual handwriting styles as outliers—an indication of an imperfect model.

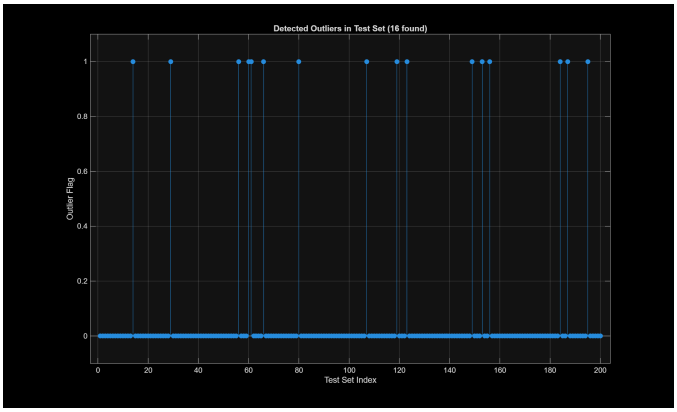

Fig. 6. Stem plot indicating the images flagged as outliers. A stem at value '1' indicates the test image at that index was determined to be an outlier while a value of '0' indicates the image at that index would be inside the normal distribution.

C. Qualitative Error Analysis

A closer look at the misclassifications reveals that they predominantly occur between digits that are structurally similar. For example, the model often confused '4's with '9's, '7's with '1's, and '3's with '8's. This is an expected limitation of a classifier based on geometric proximity in pixel space. As shown in Figure 7, a poorly written '4' can have a closed loop at the top, making its vector representation very close

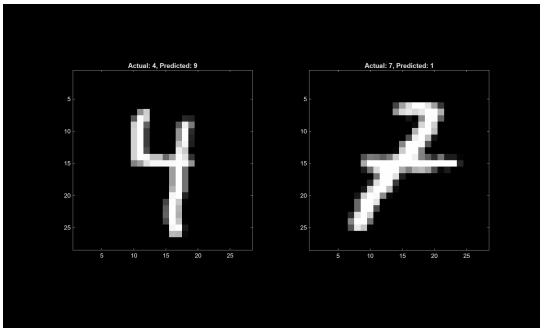

Fig. 7. Examples of misclassified test images. Left: A '4' incorrectly predicted as a '9'. Right: A '7' incorrectly predicted as a '1'.

to that of a '9'. The K-Means algorithm, lacking any higher-level feature extraction, is highly susceptible to these subtle structural similarities.

IV. Conclusion

This case study successfully demonstrated the development of a digit classifier from first principles using K-Means clustering. The key findings are:

- The Elbow Method provided a principled basis for selecting a cluster count of $k = 18$, balancing model complexity with performance.
- The strategy of labeling centroids using the mode of training data labels proved effective, converting the unsupervised clusters into a classifier with an accuracy of **71.50%** on 200 test images.
- The statistical outlier detection method, based on cluster-specific thresholds, successfully identified anomalous data points.

A. Limitations and Future Work

The main shortcoming of this approach is its sensitivity to initial centroid placement; a different random seed could lead to a different final accuracy. Furthermore, K-Means assumes that clusters are convex and isotropic (spherical), which is not always true for handwritten digits with varying slants and shapes.

Future work could address these limitations. Implementing a more robust initialization technique like **K-Means++** [3] would lead to faster convergence and more consistent results. Exploring data prior to initialization processing techniques like **normalization** could improve performance by scaling pixel values. Additionally, applying **Principal Component Analysis (PCA)** for dimensionality reduction before clustering could help the algorithm focus on the most important features and potentially improve cluster quality by reducing noise.

Overall, this project demonstrates that, despite its limitations, K-Means clustering can serve as a powerful and intuitive baseline for solving classification problems with notable complexity.

References

[1] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proc. 5th Berkeley Symp. Math. Statist. Probab., 1967, vol. 1, pp. 281-297.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[3] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in Proc. of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, 2007, pp. 1027-1035.