Centroids Visualizations as Representations of Image Data

Kevin Buhler
University of Colorado Boulder

Nate Weaver
University of Colorado Boulder

1 Introduction

Think of a green apple. Now think of a fluffy kitten. As you were imagining these, you might have noticed that you have internal 'representations' of various objects and concepts. Can we create an algorithm that has internal representations of handwritten digits and fashion items?

Traditionally, we can visualize representations of activations in sophisticated models such as convolutional neural networks (CNNs) or vision transformer architectures, such as in this paper. These models are known for their high accuracy in tasks like number recognition but come with significant drawbacks like computational intensity and the inability to understand their decision-making processes.

Given these challenges, exploring alternative, less conventional methods for image representations becomes appealing. Instead of these large scale models, could we try an unconventional technique from CSCI 4022 for representations, namely *k*-means clustering, in an efficient and interpretable manner. *k*-means clustering can segment images based on pixel similarity which allows for a form of image representation that is not only efficient but also more interpretable through a simple visualization of each centroid acting. This interpretability is crucial in applications where understanding the model's decision making processes is as important as the decision itself, such as in medical diagnostics or security-related image analysis.

Moreover, using *k*-means as a clustering approach before applying more complex classification methods can enhance performance by simplifying the input features and reducing noise. This preprocessing step can make subsequent analyses both faster and more accurate, providing a scalable solution that is both simple and efficient.

We posit that with a large amount of image data then *k*-means clustering's centroids, when plotted, will resemble the most common item assigned to the cluster and will be an identifiable representation of the datasets classes.

2 Data

In the computer vision field, one of the most famous image dataset is MNIST. The MNIST dataset (Modified National Institute of Standards and Technology) data is public and is easily accessible through Tensorflow or Keras in their dataset module, but is also available on general sites like Kaggle as well.

MNIST consists of 70,000 28x28 pixel images of hand written numbers ranging from the digits 0-9. In the dataset, each image is labeled with the digit it represents, providing a straightforward, single-feature dataset used for training various image processing systems. This dataset is typically used in machine learning validation testing as it is split into two sets, 60,000 digits in the training dataset and 10,000 in the testing. It has long been used to benchmark the performance of novel machine learning architectures and we can use it to benchmark results of our *k*-means image classification algorithm.

After obtaining good results with MNIST alone, we decided to do some preliminary experiments with the Fashion-MNIST dataset. Like MNIST, Fashion-MNIST consists of 70,000 28x28 images but of common clothing items instead of digits. Like MNIST, Fashion-MNIST also has 10 classes so we were able to use it without any modifications to our algorithm.

3 Real-world

In the real world, the problem of centroid initialization in *k*-means clustering is important in various industries with applications like customer segmentation, inventory categorization, and image recognition. Effective clustering helps companies optimize marketing strategies, improve customer service, and enhance decision-making processes. Companies like Amazon and Netflix, for example, use clustering to personalize recommendations, showing the commercial significance of clustering algorithms.

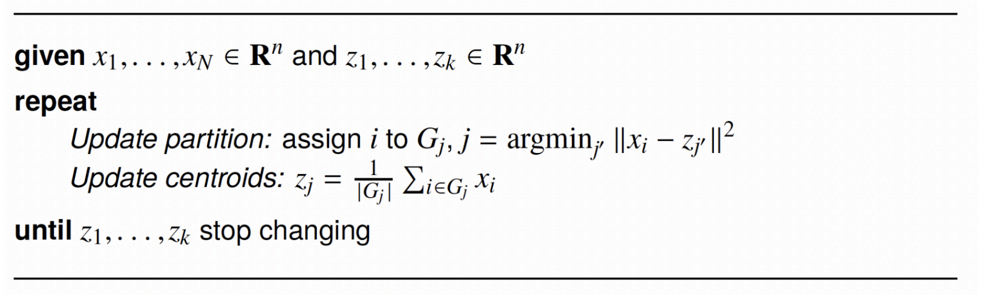
With a brief literature review on Google and Google Scholar we found a couple of papers who have employed similar methods to our own. We highlight K -Means Clustering and Related Algorithms as a similar approach that uses clustering algorithms for MNIST data. Hundreds of other similar but slightly different approaches can be found on Google Scholar, but it's important to note that these methods primarily use MNIST. We find brand new results by including Fashion-MNIST.

4 Exploratory

The preliminary work we did to establish our hypothesis was based on past understanding of *k*-means which was covered in the classroom lectures and slideshows. In class we were introduced to three versions of initialization techniques: Random, MaxMin, and Hierarchical. From this we were under the impression that MaxMin or Hierarchical would outperform Random and be able to establish a lower error rate overall. This is because each of these has a 'method' to their selection trying to optimize what goes where rather than just a random guess. From similar logic we also assumed that ten would be the right amount of clusters, each of which would be assigned to the values 0-9.

Exploring the data, we found that MNIST has an uneven distribution for each class. For example, 1 has 6742 data points while 5 has 5421. This is a 24% difference in counts and is non-negligible, which we aim to investigate later to determine if results are affected.

5 Methods



```
given $x_1, \ldots, x_N \in \mathbf{R}^n$ and $z_1, \ldots, z_k \in \mathbf{R}^n$
repeat
    Update partition: assign $i$ to $G_j$, $j = \operatorname{argmin}_{j'} \|x_i - z_{j'}\|^2$
    Update centroids: $z_j = \frac{1}{|G_j|} \sum_{i \in G_j} x_i$
until $z_1, \ldots, z_k$ stop changing
```

Figure 1: *k*-means pseudocode from February 1 2024 class presentation slide 22

In this project we used *k*-means clustering like Figure 1 with three different techniques of centroid initialization. For our centroid initializations we experimented with hierarchical, MaxMin, random, digits from 1-10, and complete noise.

For updating our partitions, we experimented with different distance measurements such as Manhattan distance (L_1), Euclidean distance (L_2), and Chebyshev distance (L_∞).

In order to convert MNIST image data into an appropriate form, we first preprocess and normalize the data. We take each 28x28 image and then flatten the pixels into a one dimensional vector. Once done, we collect each flattened vector and then feed them into the *k*-means algorithm. We use *k*=10 for both MNIST and Fashion-MNIST since we have 10 different unique classes for each and it represents a good value in our elbow plot. After *k*-means converges with a tolerance of 0.05, we start visualizing each centroid to determine if they are good representations of their respective cluster. In order to do so, we revert the shape of these 10 centroid vectors from shape 1x728 to 28x28 shape images. Since MNIST has an uneven distribution of training data, we create a balanced dataset where each class has the same amount of data points.

6 Results

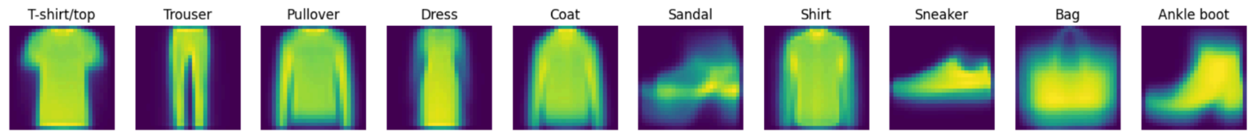


Figure 2: Average of Fashion-MNIST classes using random initialization

Throughout both datasets we find that using L_2 distance makes our *k*-means converge faster. We also find that normalizing the image values during preprocessing to range [0,1] has minimal impact on the time to converge and the quality of each centroid visualization. In figure 2 we show our plotted centroids on Fashion-MNIST. We see good results, as each centroid looks like its assigned label. For example our t-shirts in the first centroid of figure 2 do indeed look like t-shirts. When using hierarchical initialization we also see similar results of recognizable representations of digits and clothing items, although compared to L_2 distance with *k*-means it took approximately 2x the iterations to converge.

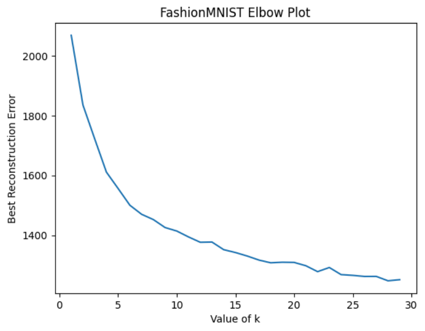


Figure 3: Elbow plot for Fashion-MNIST

We constructed elbow plots for MNIST and Fashion-MNIST and found that the optimal number of classes to be 10 in both. This aligns with our initial predictions and is supported by being a reasonable value in the elbow plot, as can be seen in figure 3. This is interesting because Fashion-MNIST has 10 classes of images, so it would make sense that *k*-means naturally represents each class. Despite the intuitive sense, we found that not all images get their respective representations as a centroid. For example, some digits like 4 would never get a clearly recognizable centroid visualization. We tried to remedy this by hand selecting digits 0-9 as the initial centroids, shown in Figure 4 below.

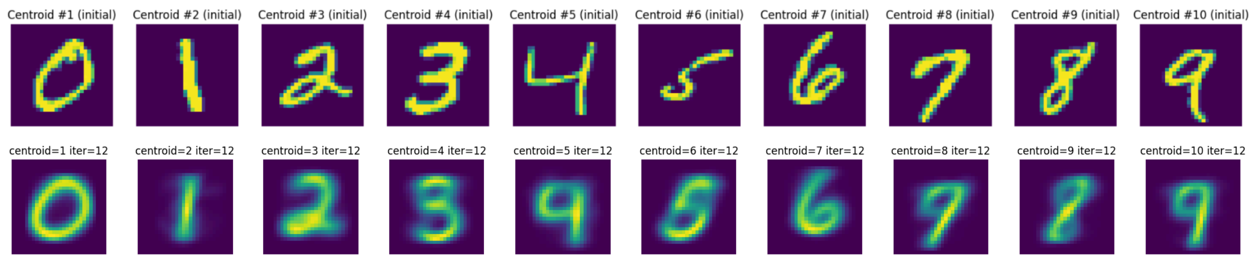


Figure 4: *k*-means when manually selecting the initial centroids to be numbers from 0-9 with a balanced dataset. Top: initial centroid and Bottom: after convergence.

Even with hand selected centroids as seen in Figure 4, we see that some numbers tend to be overrepresented. For example, four and nine look very similar, but the centroid almost always turns out to look like a nine because it is easier for the centroid to adapt to the top of the nine than maintain the original four shape. In addition, fives are shown to react similarly to one as one dominates the centroid representation due to the diagonal central axis shared between them. It is still important to point out that these adaptations are subject to change, based on the random values 0-9 used to initialize the centroids.

These results make sense as there is an uneven distribution in the dataset for the classes. However, we find that making a uniform distribution by balancing the dataset so that each class has the same number of datapoints helps ensure that each class has a centroid. We can see the outcome of using a balanced dataset in figure 4 when manually selecting the initial centroids to be numbers. Notice how in the final convergence each digit is present.

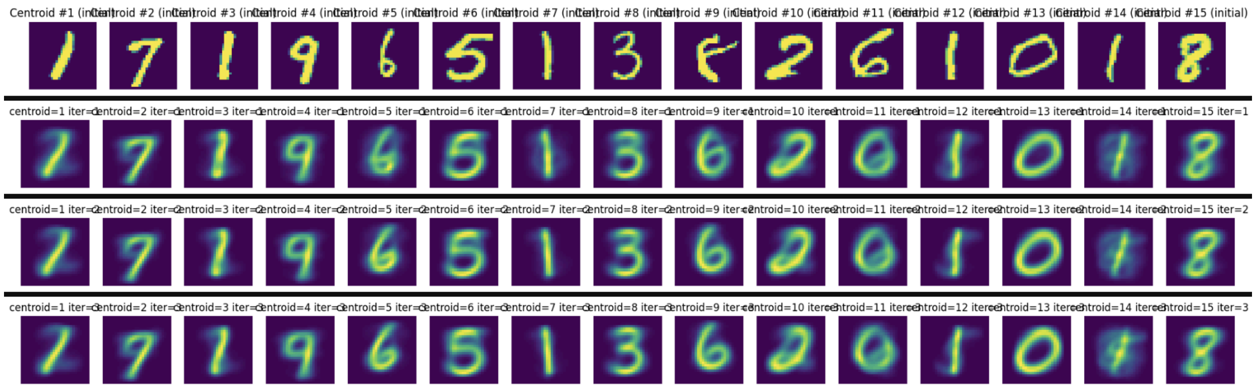


Figure 5: *k*-means with random selection of initial centroids and k = 15 (tolerance = 0.05). Top: initial centroid and Bottom: after convergence

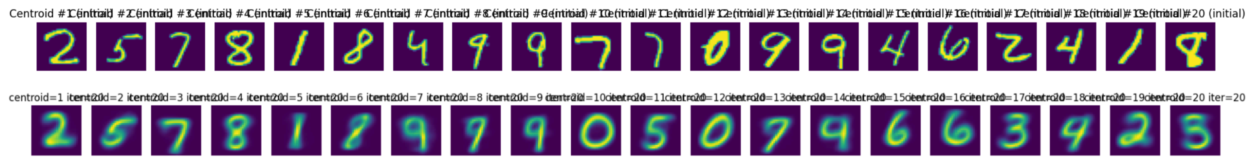


Figure 6: *k*-means with random selection of initial centroids and k = 20 (tolerance = 0.05). Top: initial centroid and Bottom: after convergence

We take this further by experimenting with visualizing different values of k, and find that with sufficiently high values of k (greater than 10) each digit is represented by a centroid. As shown in figure 5, only 4 is missing a centroid. In figure 6, we find that each digit consistently gets a centroid representative when using a balanced dataset. When using these sufficiently high values of k, we did note lower error values, but this is expected as with more clusters to be a part of, there should be a smaller distance between them. Though these values are handwritten and often inconsistent, the elbow plot associated with these higher k values showed insignificant improvement compared to a $k = 10$.

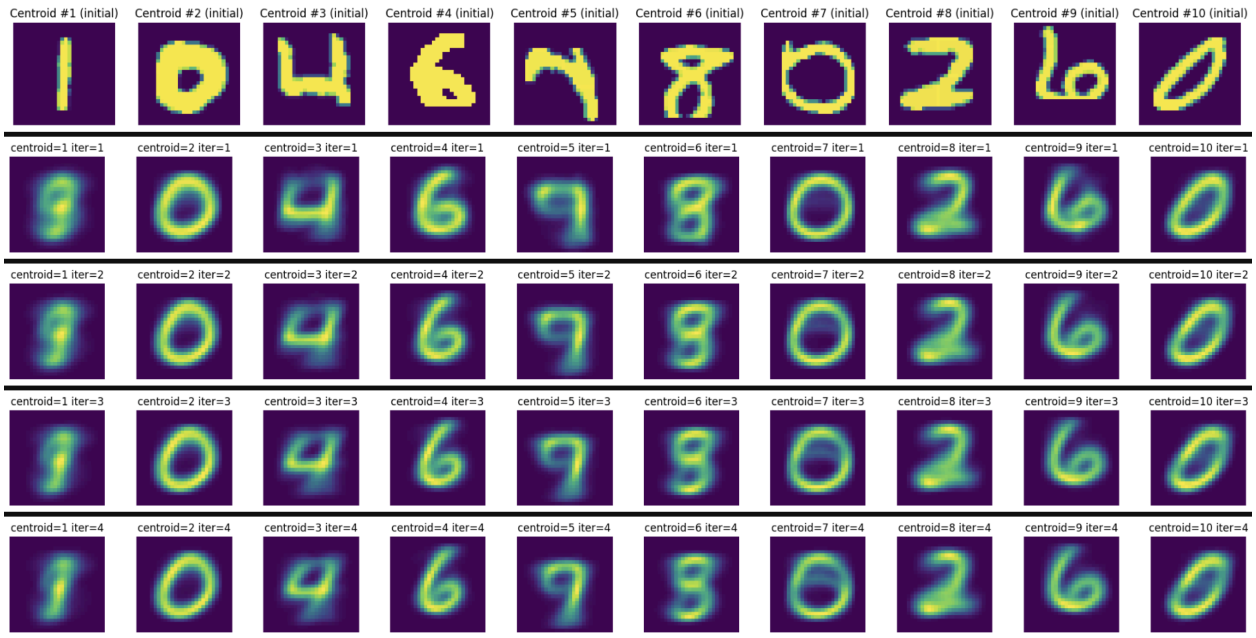


Figure 7: MaxMin initialization of centroids on $k = 10$. Topmost: initial centroid and Bottommost: after convergence

Using MaxMin as seen in Figure 7 demonstrates the adaptivity of these centroids, creating some of the most blurry or adaptive centroids we have seen thus far. It is also interesting to note that even when searching for the furthest distance point from the already existing ones, three zeroes were collected. This alone demonstrates the unpredictable nature of some of these handwritten values, as one digit can look so different from its counterparts.

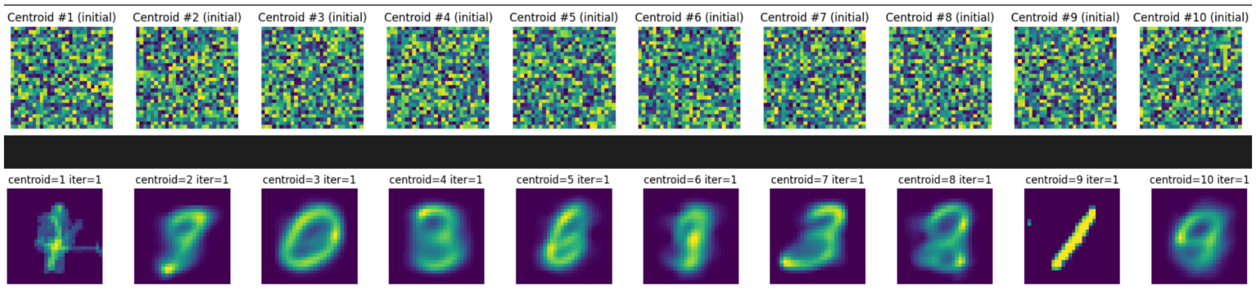


Figure 8: k-mean convergence from centroids initialization as complete noise, Top: before convergence and Bottom: after convergence

Using complete random noise as the initial centroids has a negative impact on distinguishing each value. Notice how in Figure 8 above the centroid representations look like a mix of different digits. This represents a high ability to adapt to other values, but for our purposes of number classification this is a negative property.

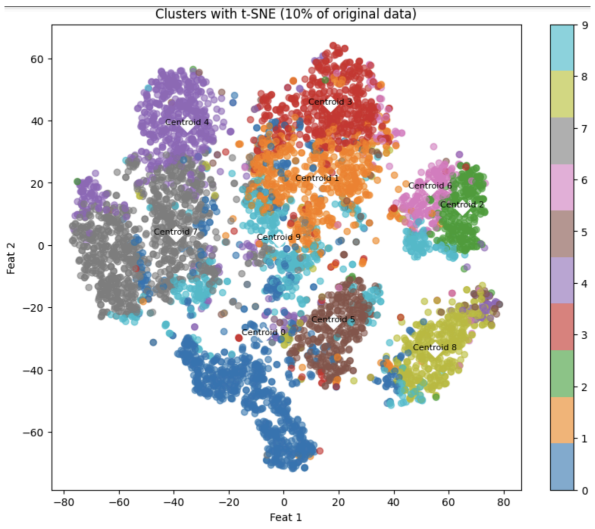


Figure 9: t-SNE projection of MNIST data with points colored by cluster and centroids marking them respectively.

Figure 9 shows a t-SNE (t-distributed stochastic neighbor embedding) that captures the high dimensional distances between data points as probabilities and translates them to a lower dimension space. This means that each cluster's density and separation displays the inherent similarities and differences among the handwritten digits. These are plotted with the respective centroids, showing how the centroids have settled after 10 iterations with random initialization. It is important to note that this is only showing 10% of the 60,000 digits we did K-means on, and thus is not a complete representation of all of the data we processed.

7 Conclusion

Overall we found that *k*-means centroids very closely resemble the cluster assignment for both MNIST and Fashion-MNIST. By analyzing centroids and elbow plots, we supported that the optimal number of clusters to be 10. However in *k*-means some numbers like 5 and 4 tend to be too similar to 1 and 9 respectively so they don't get a clear centroid representation. Consistently, the best distance metric was L_2 euclidean distance as it converged the fastest and matched all other distances in centroid visualization sharpness. We find that Hierarchical initialized centroids also result in good representations, but don't surpass *k*-means as it takes much longer to converge. From our data it is important to point out that blurry centroids may suggest that these conform better to all of its surrounding values, however it is important to note that this does not mean they are necessarily better for the value clustering we were hoping to achieve. While we worked on very simple 28x28 images, we think that this is a good method to see what the algorithm represents and how it classifies certain images. This method is much easier to analyze rather than reverse engineering what CNN representations at each layer or dissecting each activation of a neural network. The optimal combination of hyperparameters were using a balanced dataset, $k = 10$, L_2 as the distance metric, manual initialization for digits 0-9 for digit data, else random initialization, and tolerance=0.05. This results in the fastest to converge, most recognizable cluster representations for most digits and fashion items. Next steps can include experimenting with more complex algorithms like K-means++, Bradley and Fayyad's refinement, or PCA.