MA 326 Group Project

Michael Brown, Myles Gregor, Logan McLaurin, Bryant Willoughby

All authors contributed equally.

April 30, 2024

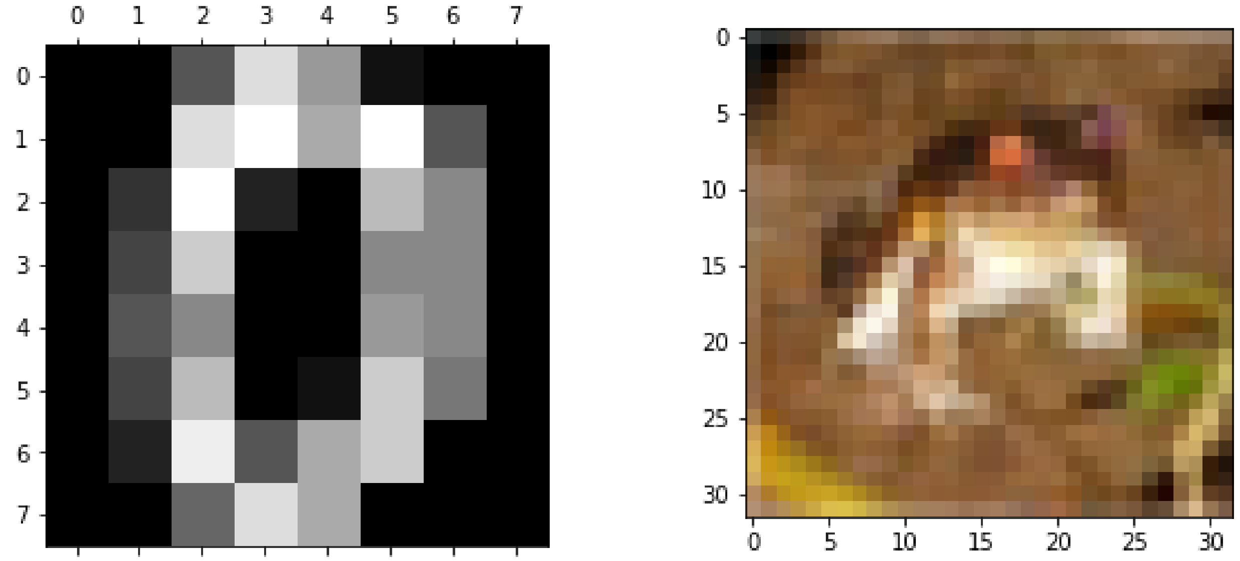
North Carolina State University

1 Introduction

The importance of computationally efficient image classification cannot be understated. From improved advanced security measures integral to national defense to more accurate identification of hazardous weather utilizing a host of imaging equipment, efficient image processing, and classification algorithms are critical to the betterment of our technological society. In this perspective, the ability to provide similar or increased accuracies from algorithms while simultaneously reducing computation resource expense is important in making this potentially life-saving tool more accessible. In our project, we seek to explore this challenge, investigating the impacts of low-rank-approximated imagery on the performance of an unsupervised algorithm K-means Clustering, and a supervised Support Vector Classifier. To increase the robustness of our investigation, we will also entertain a simple image dataset and a complex image dataset for the research to determine whether the results of our Low-Rank Approximation investigation are dependent on the complexity of the image provided to the classifier. Two datasets will be used in this research: a set of handwritten, singular alphanumeric numbers, and a set of complex color images. We will apply rank-reduction to both datasets that correspond to the percentiles $[5, 100]$ with a step of 5 corresponding to the number of singular values in the image matrix. Additionally, we look at ranks $[1 - 4]$ for each image. For each dataset, we will split the images into a 60% training and 40% testing partition. To measure the accuracy of the trained models, we will utilize our remaining test subset for each dataset, calculating the proportion of test images properly classified by both algorithms and comparing the results.

2 Data Description

We use two datasets in our project: the University of California-Irvine machine learning [2] hand-written digits dataset, and the Canadian Institute for Advanced Research [1] image dataset. The UCI ML hand-written image set returns 10 classes (integers of 0 through 9) of 8x8 gray-scale pixel images, with 1,797 images in the entire dataset. An example of the images in this dataset can be found in Figure 1a. We utilize the entire repository of images, using our partition function to set aside 60% of the dataset for training the models used in this project, and the remaining 40% of the dataset for testing purposes. Our complex case for our project involves utilizing the CIFAR color image dataset. This dataset consists of 60,000 32x32 pixel color images separated into 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. Figure 1b provides an example image of a color frog. Each image matrix contains a red, green, and blue channel, that, when stacked on top of each other, makes the color image. This is distinct from the simpler digits dataset, as each image matrix has three dimensions instead of two. We partition 2,000 randomly selected images from the CIFAR-10 image dataset into a 60% training subset and 40% testing subset for both model experiments. This subset of the CIFAR-10 was chosen to reduce compute time. We believe that the additional complexity of the RGB channels and increased pixel count in the CIFAR-10 image set as compared to the more simple grayscale UCI ML handwritten digits image set is a sufficient challenge for the models used in this project.



(a) Example image of the digit class ”0” from the UCI ML handwritten digits image repository.

(b) Example image of the image class ”Frog” from the CIFAR-10 color image repository.

Figure 1: Simple and complex images used for analysis

3 Methods

We will now discuss the mathematical details that motivate our analysis. The low-rank approximation used across both image types is discussed in Section 3.1. This includes details about the singular value decomposition of a matrix. Next, Section 3.2 describes the unsupervised K-means Clustering technique while Section 3.3 describes the Support Vector Classifier.

3.1 Low-Rank Approximation

We discuss techniques for determining redundancies in data. Consider a set of data vectors $\vec{x}_i \in \mathbb{R}^m$ for $i = 1, \ldots, n$. We assemble them into a data matrix $X = [\vec{x}_1; \vec{x}_2; \ldots; \vec{x}_n] \in \mathbb{R}^{m \times n}$. We are interested in the case where $r = \dim(\text{span}\{\vec{x}_1, \ldots, \vec{x}_n\}) < m$. In other words, $r = \text{rank}(X) < m$. That is, dimensionality reduction is attainable when the data has a dimension (r) that is smaller than the dimension of the data space (m) [8]. We will discuss how to quantify redundancies in data using the singular value decomposition (SVD) of a matrix.

Definition 3.1 (Singular Value Decomposition (SVD))**.** *Let $X \in \mathbb{R}^{m \times n}$ with rank$(X) = r$. The Singular Value Decomposition (SVD) of X is given by*

$$X = U \Sigma V^T, \tag{1}$$

where

- *$U \in \mathbb{R}^{m \times m}$ is an orthogonal matrix whose columns are the left singular vectors of X,*
- *$\Sigma \in \mathbb{R}^{m \times n}$ is a diagonal matrix with non-negative entries $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$ on its diagonal, known as the singular values of X,*
- *$V \in \mathbb{R}^{n \times n}$ is an orthogonal matrix whose columns are the right singular vectors of X.*

The SVD in Equation (1) *is often referred to as the full SVD [8].*

For a nonsingular matrix, all singular values are nonzero. For a general matrix, the number of nonzero singular values measures how much information is contained in a matrix. Accordingly, the number of zero singular values indicates the amount of redundancy [3]. Now, we can derive an economical representation for a matrix, where the size of the representation is proportional to the rank of the matrix.

Definition 3.2 (Reduced SVD)**.** *Let $X \in \mathbb{R}^{m \times n}$ have an SVD as defined previously. If rank$(X) = r$, then the matrix decomposition of X can be recast as*

$$X = \sum_{j=1}^{r} \sigma_j u_j v_j^T = U_r \Sigma_r V_r^T \textit{ where } U \in \mathbb{R}^{m \times r}, \Sigma \in \mathbb{R}^{r \times r}, V \in \mathbb{R}^{n \times r}. \tag{2}$$

A matrix of rank r can be expressed in terms of r outer products. The SVD in Equation (2) *is often referred to as the reduced (thin) SVD [6].*

Recall that the l_2 norm is defined as the largest singular value of the matrix. We will now demonstrate that the singular vectors corresponding to the k largest singular values of X define the rank k matrix closest to X in the l_2 norm [3]. Furthermore, the $(k+1)$-st singular value of X represents the absolute l_2 distance between X and the set of rank k matrices.

Theorem 1 (Low-Rank Approximation Theorem)**.** *Let $X \in \mathbb{R}^{m \times n}$ have an SVD as defined previously. Assume $k \leq$ rank(X). Suppose $X_k = \sum_{j=1}^{k} \sigma_j u_j v_j^T$. Then the absolute distance of X to the set of rank k matrices is*

$$\sigma_{k+1} = \min_{B \in \mathbb{R}^{m \times n}, rank(B) = k} \|X - B\|_2 = \|X - X_k\|_2. \tag{3}$$

X_k gives the best approximation of the matrix X by matrices of rank lower than or equal to k in the least squares sense [6].

3.2 K-means clustering

Clustering is an unsupervised learning technique which seeks to find homogeneous subgroups among the observations. K-means clustering is a particular class that partitions the data into a pre-specified number (K) of clusters [4]. In this section, we set up notation for this unsupervised learning technique, describe the approach for partitioning the data into K distinct, non-overlapping clusters, and overview the algorithm which accomplishes this goal.

Definition 3.3 (K-Means Clustering)**.** *Let K be the desired number of clusters, and let $D = \{x_1, x_2, \dots, x_n\}$ be the labeled data points, where $x_i \in \mathbb{R}^m$ for $i = 1, \dots, n$.*

1. ***Cluster Representation:*** *Given $D = \{x_i \in \mathbb{R}^m \mid i = 1, \dots, n\}$, we organize these vectors into K distinct clusters $D_l = \{x_j \mid j \in I_l\}$, where the index sets I_l satisfy:*

    - $I_l \cap I_j = \emptyset$ *if* $l \neq j$ *for* $l, j = 1, \dots, K$
    - $\bigcup_{l=1}^{K} I_l = \{1, 2, \dots, n\}$

    This ensures that each observation belongs to at least one cluster [5], and the clusters are non-overlapping (i.e., no observation belongs to more than one cluster).

2. ***Objective*** *The goal of K-means clustering is to achieve a partition of the data into subsets that are as coherent as possible. To formalize this, we introduce the following notation [5]*

    - *Define a representative vector $\vec{c}_l \in \mathbb{R}^m$ for each cluster, where $l = 1, \dots, k$.*
    - *The measure of cluster coherence is given by $q_l = \sum_{j \in I_l} \|\vec{x}_j - \vec{c}_l\|_2^2$.*
    - *Given a partition of the data $\pi = \{I_1, \dots, I_k\}$, the overall coherence of the clustering is defined as $Q(\pi, \vec{c}_1, \dots, \vec{c}_k) = \sum_{l=1}^{k} q_l$.*

Proposition 1 (K-means Clustering Objective)**.** *The formal goal of K-means clustering [5] is to determine the optimal partition π^* and the set of k representative vectors $\vec{c}_1^*, \dots, \vec{c}_k^*$ such that*

$$Q(\pi^*, \vec{c}_1^*, \dots, \vec{c}_k^*) = \min_{\pi, \{\vec{c}_l\}_{l=1}^{k}} Q(\pi, \vec{c}_1, \dots, \vec{c}_k). \tag{4}$$

The optimization problem described in Equation (4) is challenging due to the combinatorial nature of partitioning n observations into K clusters, resulting in K^n possible partitions. We present a simple algorithm that can find a local optimum instead of a global one. The algorithm involves two alternating minimization steps [5].

Theorem 2 (K-means Local Optimization)**.** *Let the alternating minimization steps have index $\tau = 1, 2, \ldots, \tau_{max}$.*

1. ***Initialization:*** *Randomly assign each observation to one of the K clusters as the initial partition $\pi^{(1)}$.*

2. ***Update Representative Vectors:*** *Given the current partition $\pi^{(\tau)}$, update the representative vectors $\{\vec{c}_l^{(\tau)}\}$ to $\{\vec{c}_l^{(\tau+1)}\}$ by solving the minimization problem*

$$Q(\pi^{(\tau)}, \vec{c}_1^{\tau+1}, \ldots, \vec{c}_k^{\tau+1}) = \min_{\{\vec{c}_l\}_{l=1}^{k}} Q(\pi^{(\tau)}, \vec{c}_1, \ldots, \vec{c}_k).$$

   The minimizer in this step is the centroid of each cluster, given by $\vec{c}_l = \frac{1}{|D_l|} \sum_{j \in I_l} \vec{x_j}$ for $l = 1, \ldots, k$.

3. ***Update Data Partition:*** *Fix the representative vectors $\{\vec{c}_l^{(\tau+1)}\}$ and update the partition $\pi^{(\tau)} \rightarrow \pi^{(\tau+1)}$ via minimization:*

$$Q(\pi^{\tau+1}, \vec{c}_1^{\tau+1}, \ldots, \vec{c}_k^{\tau+1}) = \min_{\pi} Q(\pi, \vec{c}_1^{(\tau+1)}, \ldots, \vec{c}_k^{(\tau+1)}).$$

   In this step, each observation is assigned to the cluster whose centroid is closest (using Euclidean distance).

As the algorithm iterates, the clustering improves until reaching a local optimum where further iterations no longer change the result [4]. It's important to note that this algorithm finds a local optimum, and the outcome may vary based on the random initial cluster assignments in step 1.

3.3 Support Vector Classifier (Machine)

Support Vector Machines (SVMs) are an alternative approach to classifying data. This section covers the geometric interpretation of SVMs, namely its connection to hyperplanes, and their utility in classification problems.

Definition 3.4 (Hyperplane in a p-dimensional Space)**.** *By definition, in a p-dimensional space, a hyperplane is a flat affine subspace of dimension $p - 1$ such that*

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0 \tag{5}$$

Remark that if $\vec{x} = (x_1, \ldots, x_p) \in \mathbb{R}^p$ satisfies (5)*, then $\vec{x}$ lies on the hyperplane. Equation* (5) *is the multi-dimensional analog of two-dimensional planes [7].*

Proposition 2 (Classification using Separating Hyperplane)**.** *If a separating hyperplane exists, then given a new data vector $\vec{x}^*$, we can classify it by the sign of $f(\vec{x}^*) = \beta_0 + \beta_1 X_{i1}^* + \beta_2 X_{i2}^* + \ldots + \beta_p X_{ip}^*$ such that*

1. *$f(\vec{x}^*) > 0 \Rightarrow$ assign $\vec{x}^*$ to class 1*
2. *$f(\vec{x}^*) < 0 \Rightarrow$ assign $\vec{x}^*$ to class -1*

Remark that this is an application of binary classification [7].

The SVM is a generalization of a simple classifier called the maximal margin classifier [7]. A maximal margin hyperplane is defined as a hyperplane that is the farthest from the observations. If we compute the distance from each x_i to the hyperplane, the minimal distance gives the margin. This optimization problem achieves a configuration such that all data are on the correct side of the hyperplane. Sometimes, it is inevitable that some samples will be on the wrong side of the hyperplane. Thus, there is an alternative linear support vector classifier [7]. This approach chooses the separating plane to correctly separate most of the training observations. However, there is a tolerance built in to incorrectly classify a few observations. This leads to the following theorem:

Theorem 3 (Optimization Problem for Linear Support Vector Classifier)**.** *Let* $x_1, \ldots, x_n \in \mathbb{R}^p$ *and* $y_1, \ldots, y_n \in \{-1, 1\}$. *Define* M *to be the width of the margin. Then, the constrained optimization problem for a linear support vector classifier can be denoted by*

$$\max_{\beta_0,\ldots,\beta_p,M} M \tag{6}$$

subject to the constraints

1. $\sum_{j=1}^{p} \beta_j^2 = 1$

2. $y_i(\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \ldots + \beta_p X_{ip}) \geq M(1 - \epsilon_i)$ *for* $i = 1, \ldots, n$

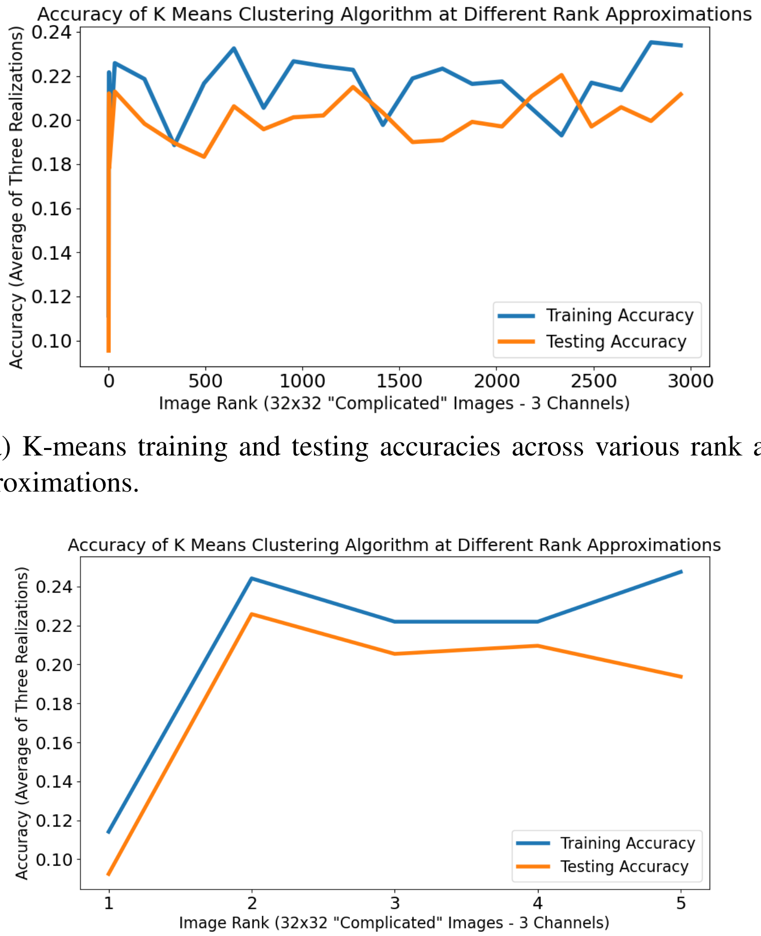
where $\epsilon_i \geq 0$ *is a slack variable that allows the samples to be on the wrong side of the hyperplane, and* $\sum_{i=1}^{n} \epsilon_i < c$ *for a tuning parameter* $c \geq 0$ *[7].*

The SVM theorem (see 3) assumes binary classification, but this is not always the case. Many applications require classifying data into more than two groups. For binary classification, a single hyperplane suffices. With 2 separating hyperplanes, there are 4 classification categories; for 3, there are 8, and in general, with N hyperplanes, there can be up to 2^N categories. Additionally, note that SVMs handle nonlinear decision boundaries, but our implementation focuses on linear boundaries.

4 Results

We now discuss the results of implementing this methodology for image classification according to the UCI ML image dataset and the CIFAR-10 image dataset.

Beginning with the K-means Clustering method, the algorithm struggled heavily with the complex CIFAR-10 images. Training accuracies and testing accuracies on this image dataset could not eclipse a 25% accuracy, with a maximized training accuracy of just over 24% and a maximized testing accuracy of just over 22%. The relative plateau in the Image Rank versus Accuracy graph



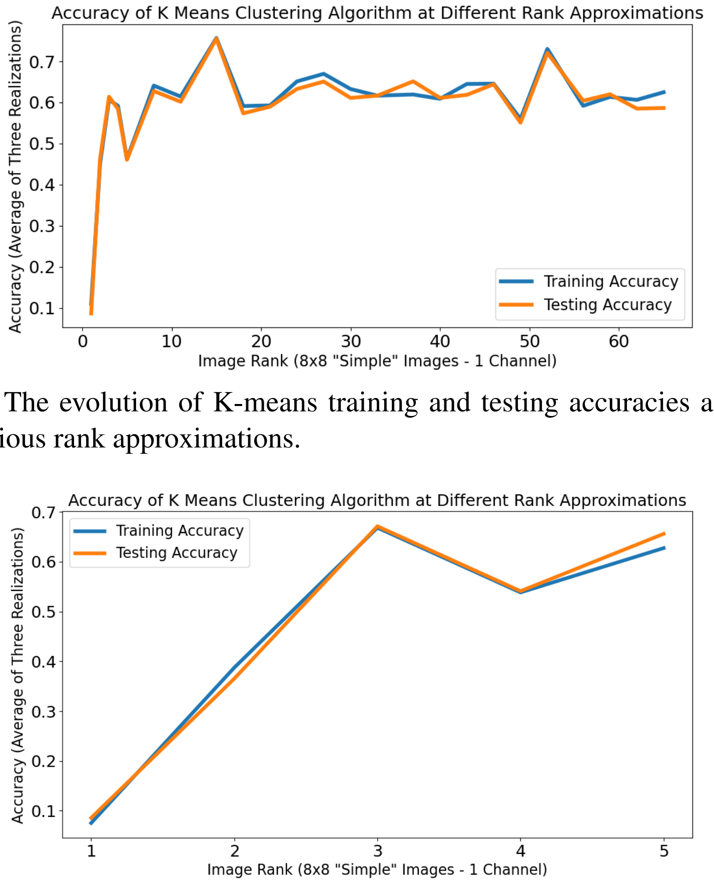
(a) K-means training and testing accuracies across various rank approximations.

(b) K-means training and testing accuracies for lowest sequence of rank approximations.

Figure 2: K-means training and testing accuracies for the complex CIFAR-10 image dataset

shown in Figure 2a indicates that, beyond the first few ranks, the addition of information back into the images did not help enhance the algorithms' ability to correctly classify the image type. A zoomed-in version seen in Figure 2b shows that upon reaching Rank 2, both the training and testing accuracies have already reached very close to their maximum values, indicating that the subsequent higher rank iterations did not impact or improve the accuracy of the classifier.

When applying the classifier to the simple grayscale images found in the UCI ML image dataset, a similar result follows, though classification accuracies are much improved. As seen in Figure 3a, there is once again a plateau in accuracy after the first several image ranks. Training and testing accuracies are both near the 75% range, and both do so at rank 15. A zoomed-in ver-



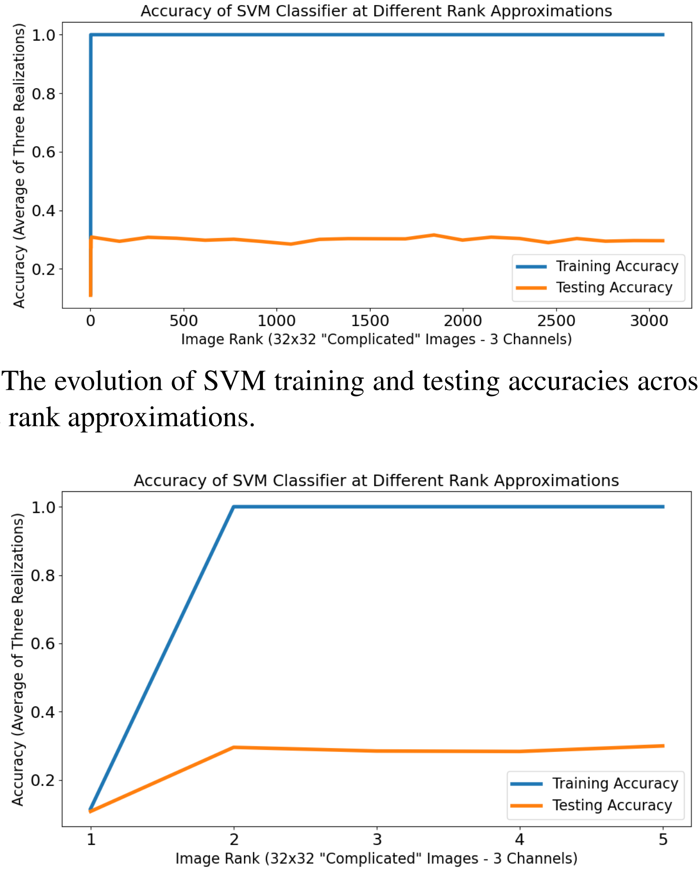
(a) The evolution of K-means training and testing accuracies across various rank approximations.

(b) K-means training and testing accuracies for lowest sequence of rank approximations.

Figure 3: K-means training and testing accuracies for the simple UCI image dataset

sion can be found in Figure 3b: within the first 3 ranks, the K-means classifier reaches accuracies exceeding 65%, and, as noted above, the classifier accuracies peak at rank 15 which is less than one-quarter of the full rank image. Beyond this rank 15 maximum, accuracies for both testing and training datasets level off between 60% and 70% for subsequent higher-rank images. Additionally, the difference between the testing and training accuracies for all usages of the K-means Clustering algorithm remains within 3-4% accuracy.

We will next discuss the results of the SVM Classifier with the complex CIFAR-10 image dataset. The SVM Classifier produces substantially better results with training accuracies, noted by the blue line in Figure 4a, but poor performance with testing accuracies for classifying these color images (indicated by the orange line). While training accuracies neared 100%, testing accuracies



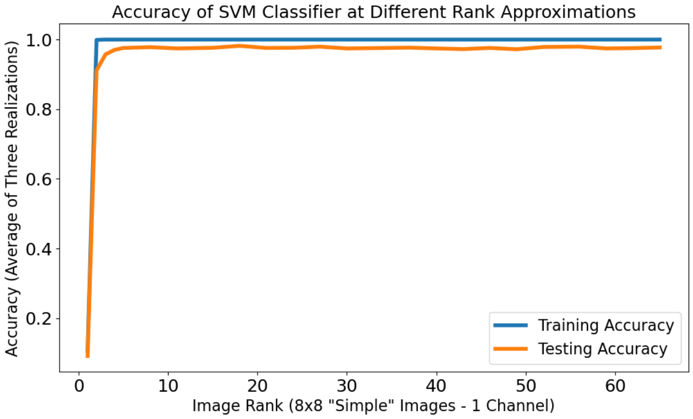
(a) The evolution of SVM training and testing accuracies across various rank approximations.

(b) SVM training and testing accuracies for lowest sequence of rank approximations.

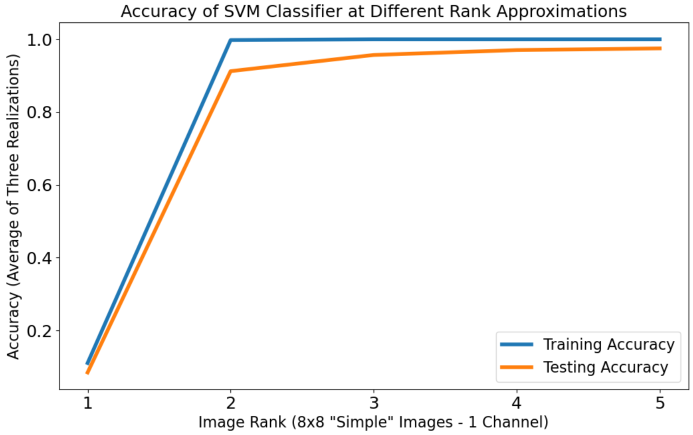
Figure 4: SVM training and testing accuracies for the complex CIFAR-10 image dataset

for the complex dataset with the SVM did not exceed far past 30%. A zoomed-in picture of the Image Rank versus Accuracy (Figure 4b) reveals the large discrepancies in accuracy, starting at image rank 2. At this point, the classifier displays nearly perfect classification on the training dataset but could only return around 30% accuracy for the test image dataset: both curves plateau after image rank 2 with minimal improvements shown for higher image ranks.

The wide range of accuracies across the training and test data is not a concern for the implementation of the SVM Classifier with the simple image dataset from the UCI ML repository. As seen in Figure 5a, accuracies for both the testing and training datasets reach nearly perfect scores before quickly plateauing. Testing accuracies are a few percentage points lower than what is recorded during training, but the range in accuracy differences is small, only exceeding a few percentage



(a) The evolution of SVM training and testing accuracies across various rank approximations.



(b) SVM training and testing accuracies for lowest sequence of rank approximations.

Figure 5: SVM training and testing accuracies for the simple UCI image dataset

points a handful of times. Figure 5b further describes the nature of the classifier's accuracy; both training and testing data accuaracies plateau after image rank 2, again indicating that higher image rank does not substantially impact the classifiers' performance. It should be noted that all accuracies discussed are an averaged value from 3 realizations to provide a more robust estimate of the algorithms' performance.

5 Discussion

Overall, it is apparent that the SVM algorithm performs more proficiently in terms of accurately classifying both simple and complicated datasets compared to the K-means algorithm. This was an expected conclusion of the experiment in terms of directly comparing the accuracies of both a supervised learning algorithm (SVM) and an unsupervised learning algorithm (K-means). It is critical to note that the only instance where the two algorithms shared similar accuracies was for the testing datasets for all rank approximations of the complex CIFAR-10 image set. This can be attributed to the fact that an accuracy below 30% is likely indicative of a sub-optimal classification scheme for any particular grouping. Therefore, it is reasonable to conclude that both the K-means and the SVM were simply performing similarly as poor in this case as opposed to the K-means showing a relative improvement over any other cases. Additionally, there is a higher degree of variability of the K-means accuracy for any given rank approximation relative to the SVM for both simple and complex datasets respectively. The collection of somewhat heterogeneous results almost certainly stems from the random initialization procedure applied during each realization of the K-means algorithm that determines the centroids utilized for classification. Despite taking the average accuracy over the course of three realizations, this fact persists.

Arguably the most prominent finding pertains to the uniform plateauing of the training and testing accuracies for both algorithms that persists beyond only two or three of the most aggressive rank approximations (beyond rank 2 or rank 3). This pattern could possibly be explained by the principle of the intrinsic dimension of the images which essentially represents the theory that images can be represented by significantly fewer variables than the image size might suggest. For our project, this directly applies to the low-rank approximation process. Previous research has empirically shown that datasets with lower intrinsic dimensions allow for the optimized training and efficiency of neural networks [10]. Furthermore, the number of samples required for adequate training is exponentially related to the intrinsic dimensions of the images where these respective classes lie [9]. With regards to this project, both the UCI and CIFAR-10 datasets demonstrate a similarly low intrinsic dimension considering how these can be generalized with extremely low rank approximations. Both the K-means and SVM algorithms help to allude to this conclusion. While the expectation would be that the UCI dataset would have a lower intrinsic dimension than the CIFAR-10, this is not a clear observation that can be shown from the methods above. The significance of this finding with regard to the plateaued accuracies cannot be overstated: the efficiency of image classification algorithms is conserved at each rank approximation only until the approximation falls below the intrinsic dimension of the images classified. Therefore, the overall success of deep learning algorithms including K-means and SVM demonstrated on high dimensional data is likely due to the low dimensional structure that can be derived from these images.

Appendix: Code Used

The code used in our project has the following files and methods:

1. **LRA.py** LRA(img, rank). Returns a low-rank approximation (LRA) of a 3-channel color image by first calculating LRA on each channel individually and then stacking them.
   LRA_grayscale(img, rank). Returns LRA at rank r of a 1-channel grayscale image.

2. **SVMCommon.py** partition(data, target, p). Returns shuffled train and test partition where p is the percentage of data to be used in train_data. Returns four values: train_data, train_target, test_data, test_target
   svc_analysis(train_data, train_target, test_data, test_target). Returns the training and testing accuracy of a linear SVC classifier scheme. Requires previous split of training data, test data, training target labels, and testing target labels.

3. **kmeansCommon.py**
   partition(data, target, p). Returns shuffled train and test partition where p is the percentage of data to be used in train_data. Returns four values: train_data, train_target, test_data, test_target
   randomInit(vectorLength). Returns k randomly initialized representative vectors of length

vectorLength contained data from the range $[0, 1]$. This code was modified from class lectures.

findClosestCluster(data, centroids). Returns the closest centroid for each data point. This code was modified from class lectures.

KMeansClusters(initalizationTechnique, train_data). Returns the closest cluster of points within train_data based on the KMeansClustering algorithm from class lectures. Returns the closest cluster and final centroids.

4. **kmeansComplicated.py**: Using the first 2,000 images of the shuffled CIFAR-10 dataset, run the K-Means algorithm.

   To check the accuracy of an unsupervised clustering algorithm, we assume the greatest proportion of the target data in each cluster is the correct classification for that cluster.

   Example: In Cluster 0 we have 100 data points, 60 of these points resolve to a ”3” in terms of their target. We assume that ”3” is the correct classification for Cluster 0. The accuracy of this cluster would then be 60%.

   kmeansSimple.py This file describes similar code to kmeansComplicated.py, except the dataset used is the UCI dataset described earlier.

   SVMComplex.py Using the first 2,000 images of the shuffled CIFAR-10 dataset, run the linear SVM algorithm.

   SVMSimple.py This file describes a similar code to SVMComplex.py, except the dataset used is the UCI dataset described earlier.

5. **Analysis Process:** We computed the train and test accuracies of the data at each rank percentile of image size $[5, 100]$ with a step of 5. Additional ranks included are ranks 1-4. Each rank has three realizations run and the average accuracy is calculated for train and test data respectively.

6. *Complete repository available at:* `https://github.com/bryant-willoughby/WilloughbyMA326.git`

References

[1] Vinod Nair Alex Krizhevsky and Geoffrey Hinton. The cifar-10 dataset. Website, 2009.

[2] E. Alpaydin and C. Kaynak. Optical Recognition of Handwritten Digits. UCI Machine Learning Repository, 1998. DOI: https://doi.org/10.24432/C50P49.

[3] Ilse C. F. Ipsen. *Numerical Matrix Analysis*. Society for Industrial & Applied Mathematics, Philadelphia, US, 2009.

[4] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, and Jonathan Taylor. *An introduction to statistical learning: With applications in python*. Springer Nature, 2023.

[5] Hangie Ji. K-means clustering. Lecture, 2024. Lecture 6.

[6] Hangie Ji. Pca & low-rank approximation. Lecture, 2024. Lecture 8.

[7] Hangie Ji. Support vector machines. Lecture, 2024. Lecture 10.

[8] Hangie Ji. Svd & dimensionality reduction. Lecture, 2024. Lecture 7.

[9] Hariharan Narayanan and Partha Niyogi. On the sample complexity of learning smooth cuts on a manifold. In *COLT*, 2009.

[10] Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021.