Comparison of Dimensionality Reduction Techniques with respect to Facial Recognition

Minu Jung 304449985

June 6, 2020

1 Introduction

As sample size and feature dimensionality increase, the amount of information that can be gained from a dataset increases as well. However, this also leads to the drawback of reaching compute limits, especially at runtime, and redundant features that are colinear, which can lead to less accurate interpretation of model parameters. Dimensionality reduction techniques can help preserve the information of the dataset, while reducing complexity and increasing model accuracy. This project analyzes 4 different dimensionality reduction techniques: Truncated Singular Value Decomposition, Truncated QR Factorization, K-means clustering, Linear Discriminant Analysis (LDA), with respect to complexity and accuracy in facial recognition using a Naïve Bayes Classifier.

For the purposes of this project $A \in \mathbb{R}^{m \times n}$ refers to data matrix with m samples and n features, $\tilde{A} \in \mathbb{R}^{m \times k}$ refers to the reduced data matrix with rank k. $a_i^T, \tilde{a}_i^T$ refers to the ith feature vector of the original and reduced data matrices, respectively.

The Dataset used is Labeled Faces From the Wild from the University of Massachusetts, Amherst

2 Dimensionality Reduction

Given matrix $A \in \mathbb{R}^{m \times n}$, the objective of dimensionality reduction (low rank approximation) is to compute the reduced data matrix $\tilde{A} \in \mathbb{R}^{m \times k}$ of rank k and basis $Q \in \mathbb{R}^{n \times k}$ that has orthonormal columns such that the low rank approximation of data matrix is formed by $\tilde{A}Q^T$. The quality of the dimensionality reduction can be measured by Frobenius norm of the original matrix and the low rank approximation.

$$||A - \tilde{A}Q^T||_F^2$$

2.1 Truncated SVD

Without proof, we state that any $A \in \mathbb{R}^{m \times n}$ can be factorized as

$$A = U\Sigma V^T$$

Where $U \in \mathbb{R}^{m \times m}$, $\Sigma \in \mathbb{R}^{m \times n}$, $V \in \mathbb{R}^{n \times n}$. U, V are orthogonal matrices and Σ has only diagonal entries that are the singular values. Without loss of generality, we assume that the singular values are sorted such that

$$\sigma_1 \geq \sigma_2 ... \geq \sigma_{min\{m,n\}}$$

Note that this is the full SVD, and that U, Σ, V refer to the full SVD factorization for the remainder of the project.

Theorem 1 (Eckart-Young)
Let $B \in \mathbb{R}^{m \times n}$. For $A = U\Sigma V^T = \sum_{i=1}^{min\{m,n\}} \sigma_i u_i v_i^T$, $B = \sum_{i=1}^{k} \sigma_i u_i v_i^T$ is the solution to the following problem:

$$\min ||A - B||_F^2 \quad \text{such that} \quad rank(B) = k < rank(A)$$

Proof (Non-formal)
Rank k matrix B can be factorized as $B = XY$, where $X \in \mathbb{R}^{m \times k}$, $Y \in \mathbb{R}^{k \times n}$. X has linearly independent independent columns, Y has linearly independent rows, and both are non-unique.

$$\begin{aligned} \min ||A - B||_F^2 &= \min ||A - XY||_F^2 \\ &= \min ||U\Sigma V^T - XY||_F^2 \\ &= \min ||\Sigma - U^T XYV||_F^2 \end{aligned}$$

The last line follows from the orthogonal invariance of Frobenius norm.

Since U, V are orthogonal matrices $rank(XY) = rank(U^T XYV)$. Therefore, the objective of the constrained optimization problem is to compute the best rank k approximation of $\Sigma = diag(\sigma_1, \sigma_2 \cdots \sigma_{min\{m,n\}})$. Given that the singular values are sorted, the best rank k approximation is

$$\Sigma \approx diag(\sigma_1, \sigma_2 \cdots \sigma_k, 0_{k+1} \cdots 0_{min\{m,n\}}) = \Sigma_k$$

Substitute back into the optimization problem:

$$\begin{aligned} \Sigma_k &= U^T XYV \\ U\Sigma_k V^T &= XY \\ \sum_{i=1}^{k} \sigma_i u_i v_i^T &= XY \end{aligned}$$

Since $XY = B$, $B = \sum_{i=1}^{k} \sigma_i u_i v_i^T$ is the solution to constrained optimization problem. This is the truncated SVD.

Reduced Data Matrix from Low-rank Approximation
Given the best rank k approximation of $A \approx \sum_{i=1}^{k} \sigma_i u_i v_i^T$, the reduced data matrix can be computed as such:

$$B = \sum_{i=1}^{k} \sigma_i u_i v_i^T = \tilde{A} Q^T$$
$$\sum_{i=1}^{k} \sigma_i u_i v_i^T = U_k \Sigma_k V_k^T$$

For $U_k = [u_1 \ u_2 \ \cdots u_k]$, $V_k = [v_1 \ v_2 \ \cdots v_k]$. Note that since $V_k \in \mathbb{R}^{n \times k}$ has orthonormal columns,

$$Q = V_k$$
$$\tilde{A} = U_k \Sigma_k = AQ$$

Centered Data Matrix (Principle Components)
In the case where the data matrix is centered ($\mathbf{1}^T A = 0$), the columns of Q are the principle components and has interesting properties relating to the covariance matrix of A. In this section, we assume that A is already centered. The covariance matrix S of a data matrix A with i^{th} row a_i^T is

$$\begin{aligned} S &= \frac{1}{m} \sum_{i=1}^{m} (a_i - \bar{a_i})(a_i - \bar{a_i})^T \\ &= \frac{1}{m} \sum_{i=1}^{m} a_i a_i^T = \frac{1}{m} A^T A \end{aligned}$$

The last line follows from the assumption that A is already centered, and that $A = [a_1 \ a_2 \cdots a_m]^T$. Using SVD factorization,

$$\begin{aligned} \frac{1}{m} A^T A &= \frac{1}{m} V \Sigma^T U^T U \Sigma V^T \\ &= \frac{1}{m} V \Sigma^T \Sigma V^T \\ &= V(\frac{1}{m} \Sigma^T \Sigma) V^T \end{aligned}$$

Note that $S = \frac{1}{m} A^T A$ is symmetric, so the eigenvalue decomposition is
$S = V \Lambda V^T$. Substituting back into expression:

$$V(\frac{1}{m} \Sigma^T \Sigma) V^T = V \Lambda V^T$$
$$\frac{1}{m} \Sigma^T \Sigma = \Lambda$$

Therefore, the principle components would be the columns of V, which are the eigenvectors of the Covariance Matrix.

Geometric Interpretation
From the max-min properties of SVD, the following constrained maximization problem has solutions:

$$\max_{||x||=1} ||Ax|| = \sigma_1$$
$$\arg\max_{x,||x||=1} ||Ax|| = v_1$$

Therefore, the first component maximizes the distance onto the projection of the rows of A onto v_1. Without derivation, subsequent values of v_i maximize the projection of the rows of A projected recursively on all previous component directions. In the case of the centered matrix, this would maximize the variance.

Complexity
The complexity of truncated svd is lower bound on the complexity of SVD factorization, which is $\mathcal{O}(2mn^2 + 11n^3)$. Notice that the complexity is not dependent on the dimensionality of the low rank approximation, k. We expect truncated SVD to have non-optimal compute time.

2.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) only exists for data with class labels (supervised). In order to describe LDA, we first define within-class covariance (S_w) and between-class covariance (S_b).

Given a training set of K classes, set of examples of class k, C_k, number of samples from class k, N_k, total set of all training samples, $C = \cup_{k=1}^{K} C_k$ and total number of training samples, $N = \sum_{k=1}^{K} N_k$

$$\bar{a_k} = \frac{1}{N_k} \sum_{a \in C_k} a$$
$$\bar{a} = \frac{1}{N} \sum_{a \in C} a$$
$$S_k = \frac{1}{N_k} \sum_{a \in C_k} (a - \bar{a}_k)(a - \bar{a}_k)^T$$
$$S_w = \sum_{k=1}^{K} \frac{N_k}{N} S$$
$$S_b = \frac{1}{N} \sum_{k=1}^{K} N_k (\bar{a_k} - \bar{a})(\bar{a_k} - \bar{a})^T$$

S_k is the covariance matrix from class k, S_w is the weighted average of the class covariance matrices S_k. S_b is the between class covariance matrix. The total covariance $S = S_w + S_b$ is equivalent to the definition in Truncated SVD section.

The primary objective of Linear Discrminant analysis is to find directions that maximize linear separability. This is done by maximizing the between class covariance and minimizing the within class covariance. Intuitively, this can be thought of as finding directions that cluster points of the same class tightly, while maximizing the distance between each cluster. Mathematically, following is the objective for finding the kth LDA vector, v_k

$$\begin{aligned} v_k &= \arg\max_x \frac{x^T S_b x}{x^T S_w x} \\ \text{such that } v_i^T S_w x &= 0, \quad i = 1, \cdots k-1 \end{aligned}$$

Without proof, another interpretation (although not immediately mathematically equivalent) of LDA would be to find

$$\begin{aligned} V &= \arg\max_X \frac{tr(X^T S_b X)}{tr(X^T S_w X)} \\ &= \arg\max_{tr(X^T S_w X)=1} (X^T S_b X) \end{aligned}$$

for $V \in \mathbb{R}^{n\times k}$. Notice that the $tr(V^T S_b V) = \sum_{i=1}^K v^T S_b v$ and $tr(V^T S_w V) = \sum_{i=1}^K v^T S_w v$. With the constraint that the covariance matrice S_w is positive definite (and both covariance matrices are symmetrical, by definition), we compute the Cholesky factorization of $S_w = R^T R$ and make the change of variables $Y = RX$, $X = R^{-1}Y$

$$\begin{aligned} tr(X^T R^T R X) &= tr(Y^T Y) = 1 \\ tr(X^T S_b X) &= tr(Y^T R^{-T} S_b R^{-1} Y) \end{aligned}$$

Therefore, the constrained maximization problem becomes

$$\max_{tr(Y^T Y)=1} tr(Y^T R^{-T} S_b R^{-1} Y) = \lambda_{max}(R^{-T} S_b R^{-1})$$

where the columns of Y are the k eigenvectors of the matrix $R^{-T} S_b R^{-1}$. Therefore, $V = R^{-1}Y$.

Reduced Data Matrix
The reduced data matrix $\tilde{A}$ can be computed from the new basis

$$\tilde{A} = AQ = AV$$

Complexity
The complexity for calculating covariance matrices is $\mathcal{O}(4mn^2)$ for calculating S, S_w, S_b. Cholesky factorization $\mathcal{O}(\frac{1}{3}n^3)$ and eigendecomposition is $\mathcal{O}(n^3)$. Therefore the total complexity is $\mathcal{O}(\frac{4}{3}n^3 + 4mn^2)$

2.3 Truncated QR

Without proof, we state that any $A \in \mathbb{R}^{m\times n}$, with $rank(A) = r$ can be factorized as

$$A = QRP$$

Where $Q \in \mathbb{R}^{m\times r}$, $R \in \mathbb{R}^{r\times n}$, $P \in \mathbb{R}^{n\times n}$. Q has orthonormal columns, $R = [R_1\ R_2]$ where $R_1 \in \mathbb{R}^{r\times r}$ is an upper triangular matrix, and P is a permutation matrix. Note that the columns of Q form an orthonormal basis, which will later be used in the low rank approximation.

Modified Gram-Schmidt with pivoting
The modified Gram-Schmidt algorithm provides a simple way to compute the pivoted QR factorization of matrix A. Following is the pseudocode:

```
**Data:** $A \in \mathbb{R}^{m\times n}$
$B_0 = A$, $R = I$, $P = I$ $k = 0$;
**while** $||B_k||_F^2 < \epsilon$ **do**
    $k = k + 1$;
    reorder $B_{k-1}$ such that largest column is placed first;
    update $P$ to reflect reordering;
    $R_{kk} = ||b||$, $q_k = \frac{1}{R_{kk}} b$, $b$ is the first column of the reordered $B_{k-1}$ ;
    $[R_{k,k+1} \cdots R_{k,n}] = q_k^T \hat{B}$  $\hat{B}$ is $B_{k-1}$ with first column removed;
    $B_k = \hat{B} - q_k[R_{k,k+1} \cdots R_{k,n}]$ ;
**end**
return $P, Q = [q_1 \cdots q_k], R = R_{1:k,1:n}$
```

Algorithm 1: Modified Gram-Schmidt with Pivoting

At iteration k, the partial factorization is

$$\begin{aligned} AP_k^T &= \begin{bmatrix} q_1 \cdots q_k & B_k \end{bmatrix} \begin{bmatrix} R_1 & R_2 \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \\ &= \begin{bmatrix} q_1 \cdots q_k \end{bmatrix} \begin{bmatrix} R_1 & R_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0} & B_k \end{bmatrix} \end{aligned}$$

Note that since the columns are reordered, q_i is in the direction of the ith largest column.

Truncated QR Factorization
With the pivoted gram-schmidt factorization, we can compute the QR factorization of A^T at iteration k

$$\begin{aligned} A^T P_k^T &= \begin{bmatrix} q_1 \cdots q_k \end{bmatrix} \begin{bmatrix} R_1 & R_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0} & B_k \end{bmatrix} \\ A^T P^T &= Q \begin{bmatrix} R_1 & R_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0} & B \end{bmatrix} \\ PA &= \begin{bmatrix} R_1^T \\ R_2^T \end{bmatrix} Q^T + \begin{bmatrix} \mathbf{0} \\ B^T \end{bmatrix} \end{aligned}$$

Where in the second line, we defined $P = P_k$, $Q = \begin{bmatrix} q_1 \cdots q_k \end{bmatrix}$, $B = B_k$ Given the partial factorization at iteration k,

$$PA \approx \begin{bmatrix} R_1^T \\ R_2^T \end{bmatrix} Q^T$$
$$A \approx P^T \begin{bmatrix} R_1^T \\ R_2^T \end{bmatrix} Q^T = \tilde{A}Q^T$$

Therefore, the low rank approximation of matrix $A \approx \tilde{A}Q^T$ has been found. However, note that this is not the solution that minimizes the Frobenius norm of $A - \tilde{A}Q^T$.

Geometric Interpretation
q_1 is in the direction of the largest row in A, since with pivoting we have reordered B_k such that the largest column is placed first (note that we have calculated the partial factorization of A^T, not A). Without derivation, subsequent values of q_i is the largest row of A projected recursively on all previous q_j, $j = 1 \cdots i - 1$.

Complexity
The complexity of the truncated QR factorization is $\mathcal{O}(2mk^2)$. Note that this is almost linear for small values of k, which is the prime advantage of using the truncated QR factorization method.

2.4 k-means Clustering

k-means as low rank approximation
K-means is typically not interpreted as a method of low rank approximation. Following is an alternative interpretation of k-means.

Given data matrix $A \in \mathbb{R}^{m\times n}$, $b_1,\ b_2, \cdots b_k$ group representatives for clusters $c_1,\ c_2 \cdots c_k$, the cost function that k-means minimizes is

$$\begin{aligned} J &= \frac{1}{n}\sum_{j=1}^{n} ||a_j - b_{c_j}||^2 \\ &= \frac{1}{n}\sum_{j=1}^{n} a_j^T a_j + b_{c_j}^T b_{c_j} - 2a_j^T b_{c_j} \\ &= \frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m} a_{ij}^2 + b_{c_j i}^2 - 2a_{ij}b_{c_j i} \\ &= \frac{1}{n}\sum_{j=1}^{n}\sum_{i=1}^{m} (a_{ij} - b_{c_j i})^2 \end{aligned}$$

where c_j is the index of the cluster that column a_j is assigned to. Note that $b_{c_j} \in \{b_1, b_2 \cdots b_k\}$ such that c_j is the index of the cluster that a_j is assigned to. We define $B \in \mathbb{R}^{m\times k}$, $B = [b_1\ b_2 \cdots b_k]$ and $C \in \mathbb{R}^{k\times n}$.

$$c_{ij} = \begin{cases} 1 & \text{if } a_j \text{ is assigned to } c_i \\ 0 & \text{otherwise} \end{cases}$$

Since each a_j belongs to only one $c_i \in \{c_1 \cdots c_k\}$, each column in C has one 1 and all 0s. Therefore,

$$b_{c_j i} = \sum_{l=1}^{k} b_{li} c_{lj}$$

Substituting into the cost function,

$$\begin{aligned} J &= \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{m} (a_{ij} - b_{c_j i})^2 \\ &= \frac{1}{n} \sum_{j=1}^{n} \sum_{i=1}^{m} (a_{ij} - \sum_{l=1}^{k} b_{li} c_{lj})^2 \\ &= \frac{1}{n} ||A - BC||_F^2 \end{aligned}$$

Therefore, the objective of k-means clustering is to find

$$\arg\min_{B,C} ||A - BC||_F^2$$

If $rank(B) = k$, BC is a rank-k approximation of A

Reduced Data Matrix from Low Rank Approximation

Applying k-means clustering to the rows of A, the low rank factorization is

$$A^T \approx BC = QRC$$

Where $B \in \mathbb{R}^{n\times k}$ and $C \in \mathbb{R}^{k\times m}$

Further improvements can be made on BC by solving the least squares minimization problem

$$\min ||A^T - BC||_F^2 = \min ||A - C^T B^T||_F^2$$
$$\hat{C}^T = AB(B^T B)^{-1}, \quad \hat{C} = (B^T B)^{-1} B^T A^T$$

Therefore, the best low rank factorization of A using k-means cluster is

$$A \approx \hat{C}^T R^T Q^T = \tilde{A} Q^T$$

The reduced data matrix is

$$\tilde{A} = \hat{C}^T R^T$$

Complexity
The complexity of initial k-means clustering (k-medioids) of the rows of A is $\mathcal{O}(m^2k^2)$. QR factorization of B is $\mathcal{O}(2nk^2)$. Least squares solution for optimal C is $\mathcal{O}(m(2nk+k^2))$.

Therefore the total number of flops is $\mathcal{O}(m^2k^2 + m(2nk+k^2))$. For large m, n, the complexity is $\mathcal{O}(m^2 + mn)$

3 Experimental Result on Facial Recognition

Dimensionality reduction techniques were used on labeled dataset, Labeled Faces in the Wild. Naive Bayes classification algorithm was used. Naive Bayes finds the between class covariance of the classes in the (dimensionality reduced) training set. The classification, given test image a is assigned as

$$k = \arg\max_k \mathcal{N}(a|\mu_k, S_b)$$

Overview
In the dataset used for the project, there were a total of 34 classes (distinct individuals) with a total of 2370 samples. Each face was flattened to vector $a \in \mathbb{R}^{1850}$, with each point a_i representing the intensity of the pixel in grayscale (255). The total data matrix $A \in \mathbb{R}^{2370\times 1850}$. Following is an example of three random faces in the dataset.

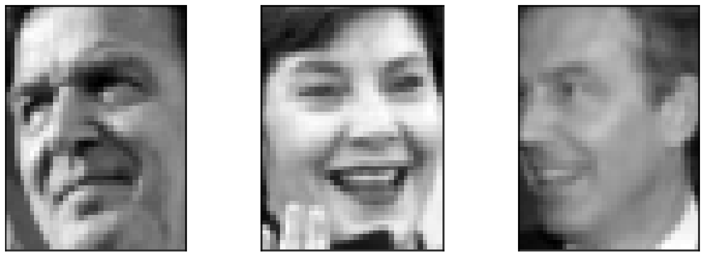

Figure 1: 3 Random Faces from Labeled Faces in the Wild

The Dimensionality reduction techniques that will be used are Truncated SVD without centering, Truncated SVD with centering (PCA), LDA, Truncated QR, k-means. Throughout the rest of the project, Truncated SVD will refer to dimensionality reduction without centering, and PCA will refer to dimensionality reduction with centering. Figure 3 examples of first components of each reduced rank basis. It's interesting to see that the pixel intensity separation is much greater for the truncated SVD than PCA, which is due to centering the Data Matrix. Also, note that the first direction of the truncated QR factorization is the exact replica of the training face with the highest norm.

Figure 4 shows the reconstruction of a random face from rank 10, 50, 100

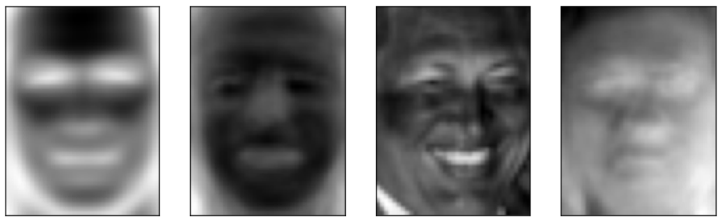

Figure 2: left to right: Truncated-SVD, PCA, Truncated QR, k-means direction

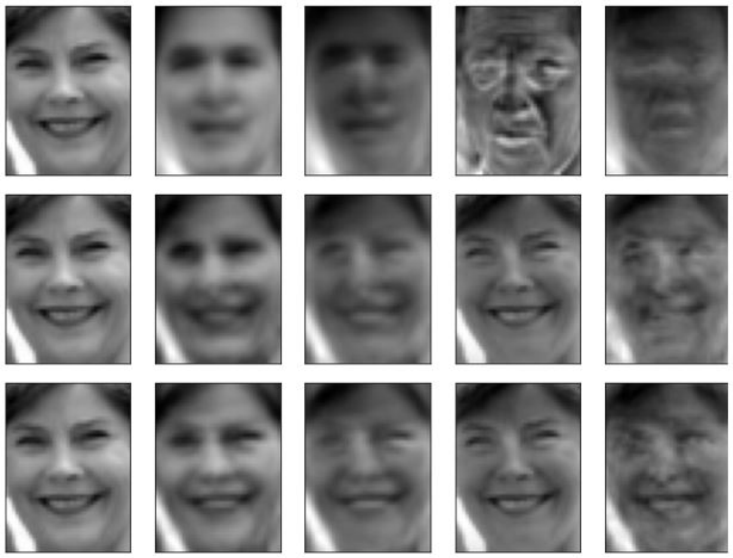

Figure 3: Image Reconstruction. From top to bottom, $k = 10, 50, 100$. From left to right: Original, Truncated-SVD, PCA, Truncated QR, k-means

approximations. It is interesting to note that the truncated PCA has qualitatively higher fidelity to the original image. K-means has the worst fidelity to the original image. Also note that in lower dimensinos, truncated QR resembles the image with the highest norm in the training set (Figure 2), as is expected.

Classification Accuracy
Figure 3 shows the training and test accurracies for all the dimensionality reduction techniques, except for LDA. LDA was excluded because the maximum dimensionality for LDA is $K - 1 = 33$, where K is the total number of separate classes within the dataset.

As expected, PCA has the best performance, followed by truncated SVD, truncated QR and k-means. The performance between centered (PCA) and non-centered truncated SVD is minimal, which may be due the relatively small size of the data-vector (1850). Unexpectedly, truncated QR factorization accuracy exceeds that of k-means. Again, this may be attributed to the relatively small size of the data-vector. The maximum accuracy for PCA using Naive Bayes is 0.71, which is a 23.8 times improvements on random guessing. Note that the objective of the project is to measure dimensionality reduction effectiveness, so further improvements can be made in model selection and hyperparameter selection to increase accuracy.

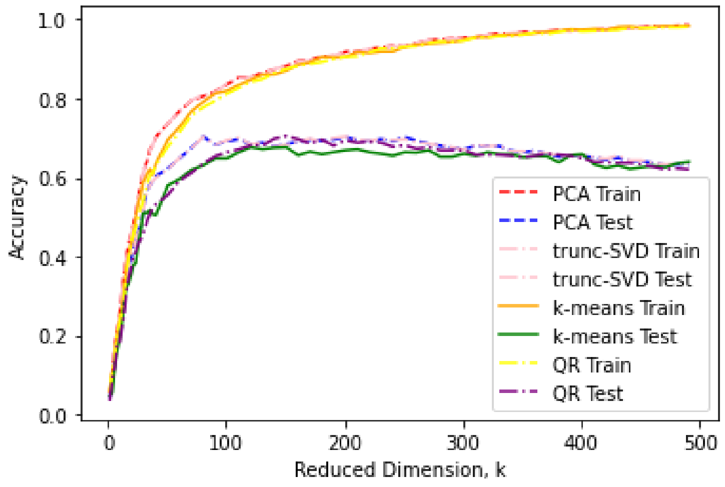

Figure 4: Training and Test Accuracy for all Dimensionality reduction techniques

Figure 4 shows the training and test accuracies for all dimensionality reduction techniques including LDA for up 30 features. The low training accuracy for LDA is due to failed implementation of obtaining the reduced data matrix $\tilde{A}$, resulting in suboptimal image reconstruction. However, the training accuracy, which was tested on sklearn's LDA package (as opposed to personal implementation), reaches close to 1.0 accuracy in training.

Notes on Complexity and Source Code Theoretically, we expected the complexity of the low rank approximations to be in the order of SVD, LDA, k-means, QR. This is explained in the complexity section of each techqniue. However, this was hard to demonstrate in equivalent conditions experimentally, as portions of my sourcecode relied on highly optimized numpy/scipy libraries on Python, whereas others were implemented by non-optimized methods with personalized algorithms.

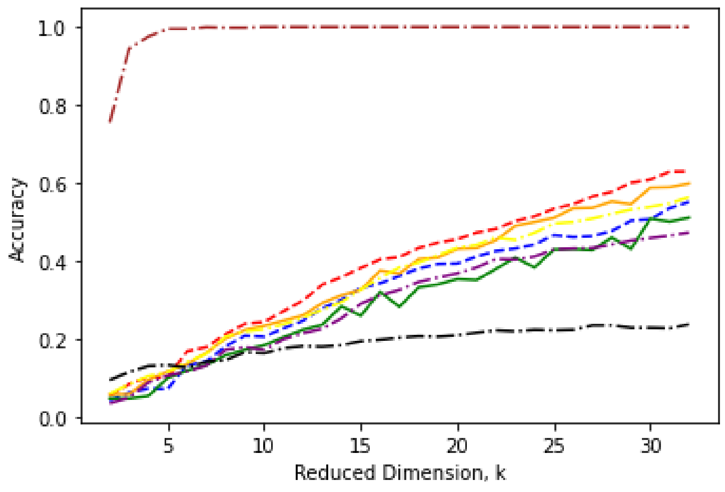

Figure 5: Training and Test Accuracy for LDA, compared with other techniques

4 Conclusion

In this Project, we have analyzed 5 dimensionality reduction techniques, Truncated-SVD (non-centered), PCA (centered), Truncated QR and k-means, and implemented the algorithms in a facial recognition problem. Note that all of these dimensionality reduction techniques ultimately collapse to some type of optimization problem with constraints, and can be well characterized by min-max properties of singular and eigenvalues. In this particular data-set, we found that PCA has the highest testing accuracy, while k-means has the lowest, and that centering the data matrix prior to dimension reduction had very little effect on testing accuracy. Further improvements can be made by optimizing the complexities of the reduction techniques and robust implementation of LDA. The source code for my project is available here.