EE499 HW2 PART2

Deniz Soysal 2305332

November 2024

Questions

a)

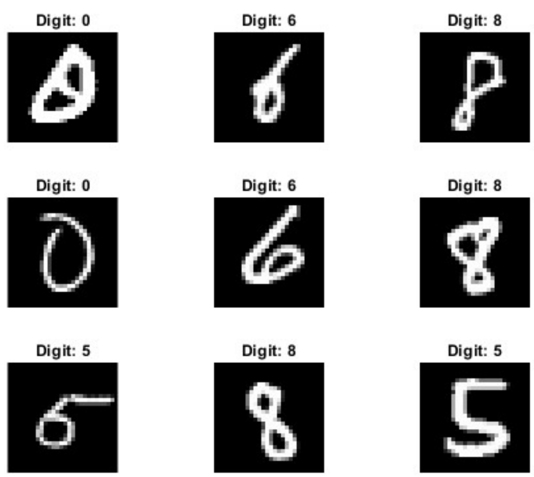

Figure 1: Samples from the dataset.

Samples of the MNIST dataset are handwritten shapes of numbers in 28*28 pixel image format.

b)

The code script for myPCA(x) function is in the appendix.

c)

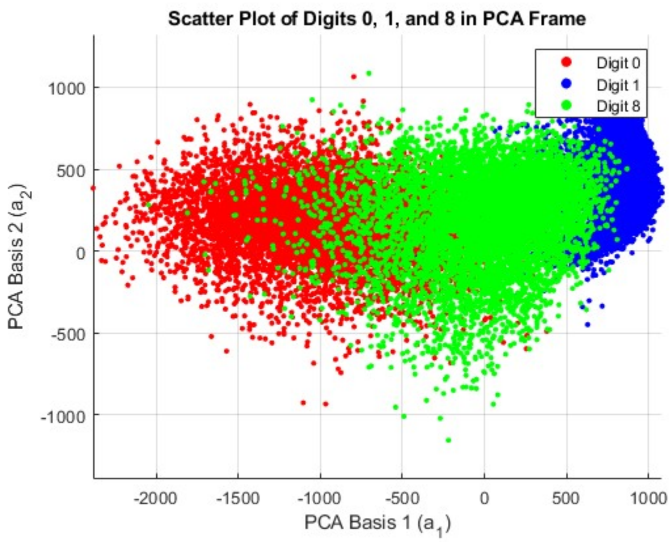

Figure 2: sample data points of numbers 0, 1, and 8 mapped under two most prominent eigenvalues.

Even though we can observe a clear pattern, data points can not be classified accurately as there are a lot of overlap. We can see that a large margin of number 0 does not overlap with other numbers. This can be interpreted as the shape of the number 0 is rather unique in its wide and circular form

d)

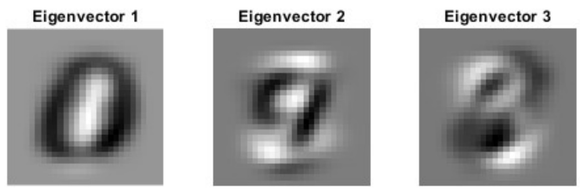

Figure 3: visualizations of the most important three eigenvectors .

These three Eigenvectors are the ones that correspond to the three of the greatest eigenvalues. The first one resembles 0 and 1 while the second one is somewhere between 8 and 7 with a circular background. the third eigenvector also resembles 8, 9, and 3. Those three eigenvectors must be the ones with the highest amount of discernible information of the number shapes which is the reason behind their visual image.

e)

The code script for myPCAdimreductor(X, K') function is in the appendix.

f)

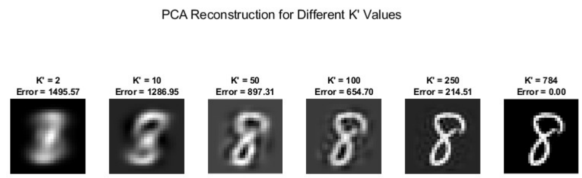

Figure 4: visualizations of the reconstructed data with increasing K'.

We see that with increasing K' value we get closer to the original data. However, we can see a clear distinction of the information in the data starting from K'=50. we would again see an increasing imge quality however sufficient number of K' to discern the information would be higher

g)

The code script for part g) is in the appendix.

h)

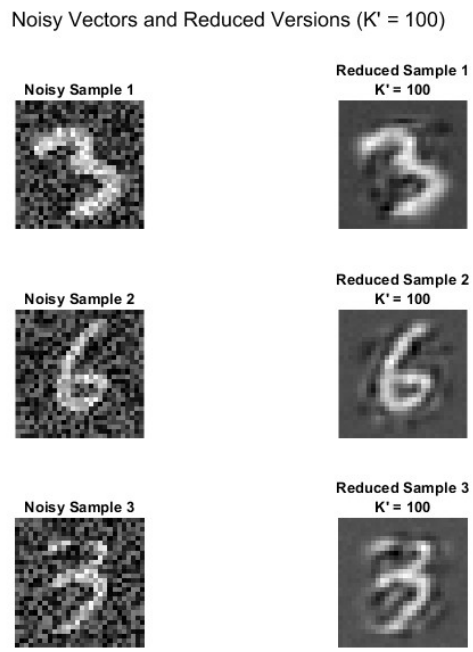

Figure 5: Visualizations of reconstructed images with their noisy originls.

I think all the numbers are pretty discernible. reduced ones are more discernible this shows that not all the variability is representative of our data thus our reduction works well with denoising.

Appendix

a)

```
close all

load('mnist.mat');

num_samples = 9;
random_indices = randperm(size(X, 2), num_samples);

% Create a 3x3 grid of subplots
figure;
for i = 1:num_samples

    idx = random_indices(i);
    img = reshape(X(:, idx), [28, 28]);
    img = img';

    % Display the image
    subplot(3, 3, i);
    imshow(img, []);

    title(['Digit: ', num2str(labels(idx))]);
end
```

b)

```
function [d, V, A] = myPCA(X)
    % myPCA: Perform Principal Component Analysis (PCA)

    meanX = mean(X, 2);
    X_centered = X - meanX;

    C = (X_centered * X_centered') / (size(X, 2) - 1); % Covariance matrix

    [V, D] = eig(C); % V: Eigenvectors, D: Diagonal matrix of eigenvalues

    d = diag(D); % Eigenvalues as a column vector

    [d, idx] = sort(d, 'descend'); % Sort
    V = V(:, idx); % Reorder eigenvectors accordingly

    A = V' * X_centered; % Project data onto the principal components
    X_reconstructed = meanX + V * A; % Reconstruct the data using PCA
    reconstruction_error = norm(X - X_reconstructed, 'fro');
```

```
    fprintf('Reconstruction error: %.2e\n', reconstruction_error);
    assert(reconstruction_error < 1e-8, 'Reconstruction failed!');
end
```

c)

```
close all

% Load the MNIST dataset (Assuming X and labels are loaded)
load('mnist.mat'); % X: 784x70000, labels: 1x70000

% Perform PCA
[d, V, A] = myPCA(X);

% Filter indices for digits 0, 1, and 8
indices_0 = find(labels == 0);
indices_1 = find(labels == 1);
indices_8 = find(labels == 8);

% Extract first two rows of A for these digits
A_0 = A(1:2, indices_0);
A_1 = A(1:2, indices_1);
A_8 = A(1:2, indices_8);

% Plot scatter plots
figure;
hold on;
scatter(A_0(1, :), A_0(2, :), 10, 'r', 'filled', 'DisplayName', 'Digit 0');
scatter(A_1(1, :), A_1(2, :), 10, 'b', 'filled', 'DisplayName', 'Digit 1');
scatter(A_8(1, :), A_8(2, :), 10, 'g', 'filled', 'DisplayName', 'Digit 8');
xlabel('PCA Basis 1 (a_1)');
ylabel('PCA Basis 2 (a_2)');
title('Scatter Plot of Digits 0, 1, and 8 in PCA Frame');
legend('show');
axis equal;
grid on;
hold off;
```

d)

```
% Assume V is the matrix of eigenvectors obtained from myPCA
figure;

for i = 1:3
    % Reshape the i-th eigenvector into a 28x28 grid and transpose for visualization
```

```
    eigenvector_image = reshape(V(:, i), 28, 28)';

    % Create subplot for each eigenvector
    subplot(1, 3, i);
    imshow(eigenvector_image, []);
    title(['Eigenvector ', num2str(i)]);
end
```

e)

```
function Xhat = myPCAdimreductor(X, Kprime)
    % myPCAdimreductor: Perform PCA-based dimensionality reduction and reconstruction

    meanX = mean(X, 2);
    X_centered = X - meanX;

    % Perform PCA
    [d, V, A] = myPCA(X);

    V_Kprime = V(:, 1:Kprime);    % Select the top Kprime eigenvectors
    A_Kprime = A(1:Kprime, :);    % Select the top Kprime rows of the coefficients matrix

    % Step 5: Reconstruct the dataset
    Xhat = meanX + V_Kprime * A_Kprime;
end
```

f)

```
% Load MNIST data
load('mnist.mat'); % Assume this loads X and labels

n = randi(size(X, 2)); % Random index
xn = X(:, n); % Randomly selected vector

Kprime_values = [2, 10, 50, 100, 250, 784];

figure;
num_plots = length(Kprime_values);
for i = 1:num_plots
    Kprime = Kprime_values(i);

    % Reconstruct
    Xhat = myPCAdimreductor(X, Kprime);
    xhat_n = Xhat(:, n);
```

```
    % Calculate reconstruction error
    reconstruction_error = norm(xn - xhat_n);

    % Reshape to 28x28 for visualization
    xhat_image = reshape(xhat_n, 28, 28)';

    % Plot
    subplot(1, num_plots, i);
    imshow(xhat_image, [], 'InitialMagnification', 'fit');
    title(sprintf('K'' = %d\nError = %.2f', Kprime, reconstruction_error));
end

% Overall title
sgtitle('PCA Reconstruction for Different K'' Values');
```

g)

```
load('mnist.mat');

% Generate the noisy dataset Y
[N, K] = size(X);
W = 256 * rand(N, K);
Y = X + W;
```

h)

```
close all

load('mnist.mat');

% Generate noisy dataset
N = size(X, 2);
noise = randi([0, 256], size(X));
Y = X + noise;

% Randomly sample three indices
indices = randi(N, 1, 3);
yn_samples = Y(:, indices);

% K' value
Kprime = 100;

% Apply PCA
Yhat = myPCAdimreductor(Y, Kprime);
```

```
figure;
for i = 1:3
    yn = yn_samples(:, i);
    yhat_n = Yhat(:, indices(i));

    % Reshape
    yn_image = reshape(yn, 28, 28)';
    yhat_image = reshape(yhat_n, 28, 28)';

    % Plot
    subplot(3, 2, 2*i-1);
    imshow(yn_image, [], 'InitialMagnification', 'fit');
    title(sprintf('Noisy Sample %d', i));

    subplot(3, 2, 2*i);
    imshow(yhat_image, [], 'InitialMagnification', 'fit');
    title(sprintf('Reduced Sample %d\nK'' = %d', i, Kprime));
end

sgtitle(sprintf('Noisy Vectors and Reduced Versions (K'' = %d)', Kprime));
```