ITE4052 Computer Vision

Final Term Project - PCA

2016026080 한다진

**# Q1 – Calculate Eigenfaces**

Eigenfaces are 2-dimentional version of eigenvectors. With these eigenfaces, we can express image A as a point on specific coordinate system. In this coordinate system, each eigenfaces are the coordinate axes. In this project, the Goal is to transform the coordinate system from with higher number of coordinate axes to smaller number of coordinate axes. Of course, there are information loss during coordinate system conversion. So the Key Idea is to loss small information. So We Calculate the eigenvectors, then use eigenvectors with higher eigenvalue as a new coordinate axes. Ths decrease the loss of information during conversion. The original Image is 28x28 so it is 784 dimension point. The converted image is expressed with 300 dimension point.

First normalize x\_train dataset. “phi = x\_train – np.mean(x\_train)”

Second, Create Covariance C with np.cov function.

Then I can calculate eigenvalues and eigenvectors.

So <Q1 Top 40 Eigenfaces> is the visualization of 40 eigenvectors with high eigen value.

**# Q2 – Image Approximation**

With eigenvector and eigenvalues, we can express image with 300 dimensional point. Now calculate K with two condition.

First, “Smaller K, more meaningful transformation”

Second “Bigger K, more accurate expression”

So I use eigenvalue to which eigenvector is better for coordinate axes. I Select K eigenvectors with highest eigenvalue. The sum of eigenvalues of selected eigenvectors should be bigger than the 85% of total eigenvalue sum. So K is calculated as 45. Then Calculate the MSE. When I increase K, MSE decreses.

**# Q3 – Fast approach**

In Q1, I make C with 300 x 784 matrix(A). 300 is the number of images in dataset. 784 is the number of pixels in single image. ATA is 784 x 784 matrix. This time, I make C with AAT. So the shape is 300x300. This is faster than the Q1 algorithm. Often the number of dataset is smaller than the total pixel in single image because nowadays many study use the high resolution image as a dataset. So Using AAT as a C matrix and then calculate eigenvalue and eigenvectors in 300x300 dimension. Then Select K eigenvectors, so selected eigenvectos are Kx300 dimension. Now convert Kx300 dimension to Kx784 dimension using 300x784 dataset matrix.

Then I compare the time, Q3 algorithm is 4 times faster than Q1 algorithm.

Other explanation is commented on my pca.ipynb jupyter file.

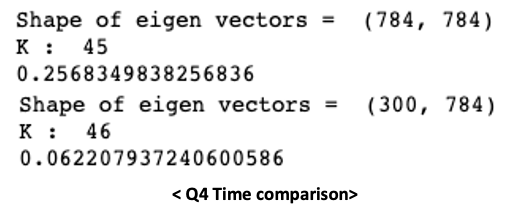
**# Results**

**A picture containing text

Description automatically generatedCalendar

Description automatically generated**

Graphical user interface, text, application

Description automatically generated****

A picture containing text

Description automatically generated