Assignment 2 : The one-stop PCA Center

Suwaphit Buabuthr
Department of Computer Science
Texas Tech University
Suwaphit.Buabuthr@ttu.edu

Abdul Serwadda
Department of Computer Science
Texas Tech University
Abdul.serwadda@ttu.edu

***Abstract*—This assignment proposes a dashboard by using a shiny library to represent understanding in Principal Component Analysis (PCA) and eigenfaces on the website platform. The project would demonstrate the applying PCA with a face dataset from Kaggle[1] as a scenario from the raw data until get the output from the PCA technique. Then the author would apply and compare the output from PCA to do face classification also including the improvement method to produce a better result.**

I. Dataset

The data set is used in this project are open data which was provided by Kaggle[1]. This data set consists of a list of training 7049 images (Each row contains the (x,y) coordinates for 15 key points and image data as a row-ordered list of pixels.) and a list of 1783 testing images (Each row contains ImageId and image data as a row-ordered list of pixels). Fig. 1 represents a data sample from this data set. This project would select the first 100 images to do a PCA.

Fig. 1. A sample data from the data set that was used in this project

II. Introduction

PCA is a useful technique for exploratory data analysis. From a lesson in class, PCA is applied for reducing the dimensionality of data by not losing important information and decreasing multicollinearity in variables because each component would not be related to the other. Dimensional reduction can happen by finding the directions where the data sample is the most variance, the directions where the data is most spread out. So, we try to find the best straight line that the data can project along with it. To derive new variables from the original variables that preserve most of the information given by their variances, we will find a covariance matrix from the data sample to perform PCA. Then we will get eigenvalues and eigenvector from these co-variance matrices. The eigenvector represents the principal components of this data set. The eigenvalues of the data set are used to find the proportion of the total variance explained by the components in which each new component would not relate with the other. You would see from the fig. 2 which represent the relation between variance and number of component. A scree graph illustrates the proportion of variance explained by each subsequential eigenvalue. Moreover, you would see from cumulative graph that the first 50 components can explain variance of the population around 95% and if we use the first 100 components, it can explain almost 100%

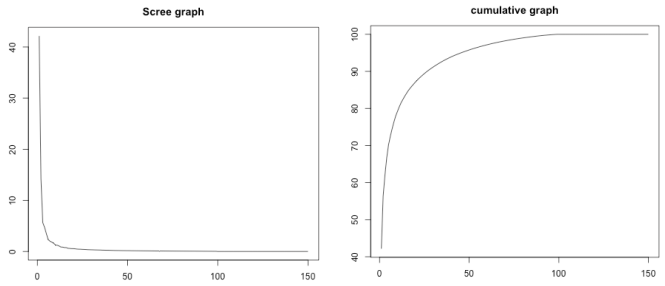

Fig. 2. A scree and cumulative graph of the eigenvalues

The first one hundred eigenvectors can illustrate as shown in fig.3

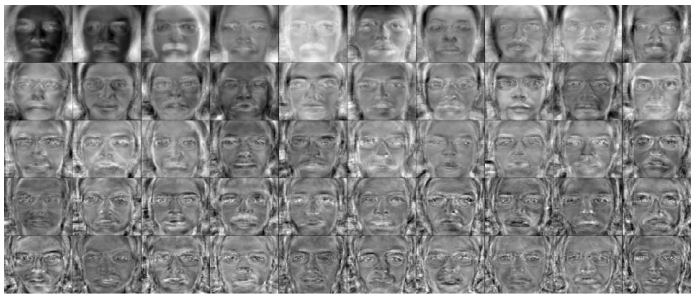

Fig. 3. The eigent face of this data set

The figure above shows the eigen face which is a set of eigenvectors of this data set. We would use the **eigen face** term when used in the computer vision problem of human face classification.

Then we use these eigenvectors to classify a face data set. In this project, we illustrate a reconstructed face by varying the

number of PCA components in 3 values: 20, 50, and 100.

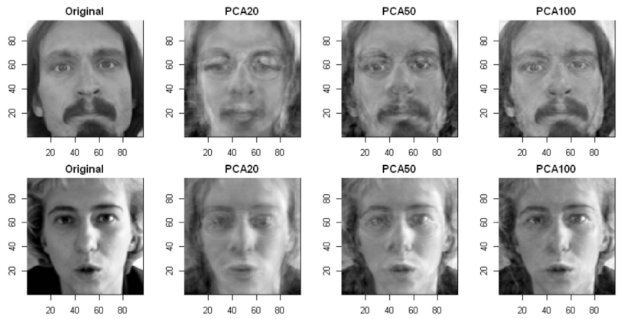

Fig. 4. Using eigen face to reconstruct a face by comparing between a different number of PCA in 20, 50, and 100

You would see in fig.4 that even we try to use 100 components to reconstruct and classify the face. There is some part of the picture unsmooth.
So I try to improve results by subtracting the original data by a mean of these data set as shown in fig.5. Then we use subtracted data to reconstruct a face again as shown in fig.6

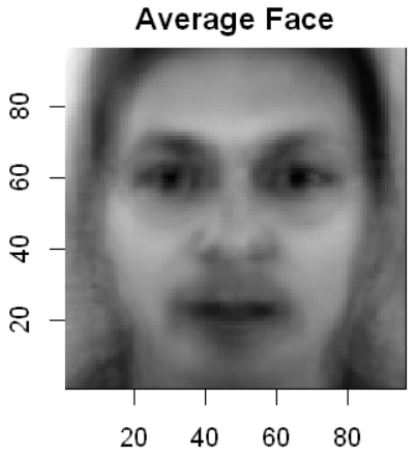

Fig. 5. A mean value of data set

Fig.6 would represent the result after subtracting each face by a mean value of this data set. So, If you compare between 4 and 6, you would see the result improvement.

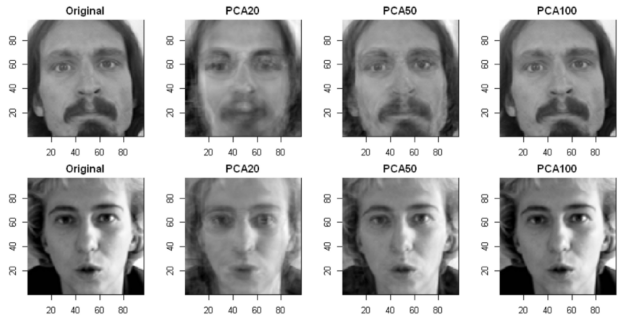

Fig. 6. Using eigen face after subtracting with mean value to reconstruct a face by comparing between a different number of PCA in 20, 50, and 100

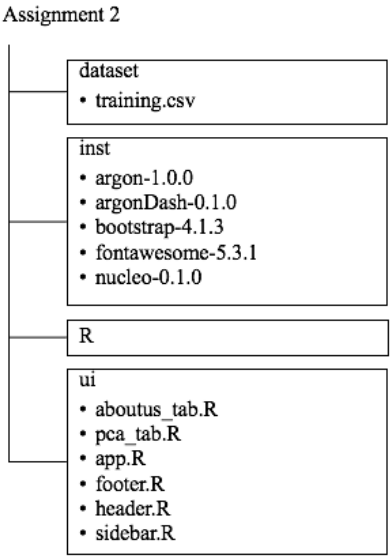

Fig. 7. Project structure

III. Project structure and Features

This project was implemented by using **shiny** library to represent in a web-based platform. The author also use **argonDash**[2] which is a open-source HTML library to demonstrate better visualization. So the structure of this project is represented in the figure 7.

1) dataset: the location to store a dataset that the author use in this project
2) inst: this location would provide all library that this project use for customizing the visualization such as bootstrap would provide a library for HTML structure, fontawsome would provide a special character that we use it as an icon, etc.
3) R: default R script provided bt argonDash library which uses to the config user interface
4) UI : this folder would provide all files that use to implement the user interface file.
   - aboutus_tab.R: user interface under About us menu tab
   - pca_tab.R: user interface under PCA Scenario menu tab
   - app.R: the main file that combines each part of the web both of the user interface and back-end part.
   - footer.R: user interface for customizing the bottom part of the website
   - header.R: user interface for customizing the top part of the website
   - sidebar.R: user interface for customize the side part of the website

IV. Project features

The project propose 2 features in 2 menus : a **PCA scenario** and **About us**

1) PCA scenario: the content under this menu would explain the PCA process in each step as storytelling. Including face recognition by using the output of PCA (eigen face) by comparing the difference between each result when we pick a different number of components from the PCA technique.
2) About us: this menu would show a channel to access the source code on github[3][4] and a demo video[5] of this project

V. Future work

Some features in this project take more than 5 seconds to execute. We can reduce the time in each rendering to be lower for a better user experience.

References

[1] Y. Bengio, *Facial Keypoints Detection*, 2016 (accessed October 23, 2020). [Online]. Available: https://www.kaggle.com/c/facial-keypoints-detection/data

[2] D. Granjon, *argonDash*, 2019 (accessed October 23, 2020). [Online]. Available: https://github.com/RinteRface/argonDash

[3] S. Buabuthr, *assignment2*, 2020 (accessed November 4, 2020). [Online]. Available: https://github.com/Bellypoly/eigen_face_pca.git

[4] ——, *assignment2*, 2020 (accessed November 4, 2020). [Online]. Available: shorturl.at/cfCG4

[5] ——, *video demo*, 2020 (accessed November 4, 2020). [Online]. Available: https://youtu.be/0TVNZ67xNdI