DS5230

David Dadiomov

Brendon Hahm

Facial Recognition using Eigenfaces

**Introduction:**

Facial recognition is used in many applications in everyday life such as security systems, access control, and biometric identification. This project will explore this task by using Eigenfaces, a widely used technique in facial recognition, to identify faces in images. Specifically, the project will investigate the effectiveness of Eigenfaces in recognizing faces in real-world scenarios, where lighting conditions and facial expressions are not controlled. While CNN has historically proven to be a powerful tool for facial recognition, eigenfaces have a few extremely strong advantages compared to CNN being computationally more efficient, giving a clear understanding of characteristic features that can be visualized and is more interpretable. Eigenfaces method is relatively intuitive, and if given a simpler method for recognition,, can only require a single image in “training” per person to recognize an individual. In our project we explore and use the Labeled Faces in the Wild (LFW) dataset . The LFW dataset is a collection of labeled face images designed for studying the problem of unconstrained face recognition. We used this data to train our model, while doing some preprocessing to our data. The accuracy of our model was around .74 which was not as high as we expected from our model. Once we trained our model we added two functions where you can add a folder of photos that includes images of people (as long as they are the same people as the model was trained on) that will take these photos, crop the face and then predict it based on our model. We found out that this was very hard to do accurately and no matter the amount of preprocessing we did to our images it would only minimally affect in improving our accuracy. From this project, we have learned alot about facial recognition technology, including the identification of available libraries that facilitate image cropping and resizing. Moreover, we have developed an understanding of the Eigenfaces method and its potential for achieving higher levels of accuracy in comparison to alternative approaches.

**Dataset:**

A well-known and well-liked dataset for researching face recognition tasks in unrestricted settings is the Labeled Faces in the Wild (LFW) dataset. It consists of more than 13,000 pictures of more than 5,000 people that were taken from the internet, with a range of image quality, lighting, facial expressions, and poses. The LFW dataset is a great option for assessing how well facial recognition algorithms function in difficult situations because of the variety of photos it contains, which resembles real-world conditions. Researchers and developers have made considerable use of the dataset to benchmark their facial recognition algorithms and assess how well they perform in comparison to other cutting-edge methods.The LFW dataset also provides a foundation for the development of more sophisticated and reliable facial recognition algorithms that can handle a variety of events and conditions, ultimately enhancing the performance and usefulness of these systems in real-world applications and why we decided to use it to train our model.

Since we wanted to maintain at least 70 images per person, we ended up with 1288 observations and resized the images such that they were 75 x 56 images. This means that the size of the images dataset is (1288, 4200). When performing PCA on this data, we found that when k, the number of principal components, is equal to 38, our principal components still explain 80% of the variation in the original image data. So after performing PCA with k=38, our eigenfaces data is (1288, 38), which is a significantly smaller feature space.

Since we want to test how generalizable our eigenfaces can be, we collected “random” images of the people to a separate dataset which we will call the external validation set. The seven people in both this data and the LFW used are:

1. Ariel Sharon
2. Colin Powell
3. Donald Rumsfeld
4. George W Bush
5. Gerhard Schroeder
6. Hugo Chavez
7. Tony Blair

We collected 5 images per person for 35 images total. We used a pre-trained facial detection algorithm to find the locations of the faces and to crop them. We then preprocess the image and resize the images so they can be projected on the eigenspace and used for facial recognition.

**Methods:**

In our project, we aimed to evaluate the effectiveness of the eigenfaces method for face recognition tasks, finding the best classifier with eigenfaces, and its effectiveness on images more varied. First, we compared the performance of a 1NN algorithm which utilized eigenfaces with the performance 1NN on the raw data. By projecting the original image data onto the eigenspace, we can obtain a lower-dimensional representation of the images. This new representation emphasizes characteristic features for distinguishing the facial images while discarding less important variations. The lower dimensional space, formed by the eigenfaces, is used for classification in facial recognition tasks. We divided the data into training and testing sets, maintaining a 90-10 split. Data standardization was performed using the StandardScaler method to achieve a mean of zero and unit variance for the features. For the eigenfaces-based method, PCA was applied to the standardized data to extract the top 38 eigenfaces, which represent the most significant variations in facial features denoted by the magnitude of the corresponding eigenvalues. These eigenfaces were used as an orthonormal basis for projecting the input data into a lower-dimensional space.. In contrast, the regular method did not employ PCA, and the classification was performed using the original feature space. We used the 1NN for classification in both cases, calculating the Euclidean distance between feature vectors in the respective feature spaces. By comparing the accuracy scores of the eigenfaces-based method and the regular method, we assessed the performance and effectiveness of PCA in combination with eigenfaces for face recognition tasks.

While we use 1NN as a benchmark to measure the efficacy of the eigenfaces method, we also wanted to see if other ML models could perform better for classification. We also used KNN, logistic regression, SVM, and Random Forest to see which ML method performed the best on the eigenfaces. We used two repeated stratified 10-Fold for cross validation and randomized grid search for hyperparameter tuning. These methods allowed us to find the best classification model with the eigenfaces method by comparing accuracy scores.

For our function for predicting a face in a photo from our own input, we developed a methodology for face recognition using a combination of preprocessing techniques, [1]RetinaFace for face detection, and a machine learning model for classification. For face detection, given an input photo, RetinaFace was employed to detect faces in the image. If no faces were detected, the image was discarded. If multiple faces were detected, the largest face was selected for further processing. The detected face was resized to match the dimensions (w, h) of the images in the training dataset, and then the image was then converted to grayscale, followed by bilateral filtering to reduce noise while preserving edges.

[3]Grayscale conversion is the process of converting a colored image into shades of gray. It involves removing the color information from an image, leaving only the intensity values. This simplifies the image data, reducing the computational complexity, and sometimes helps improve the performance of algorithms that primarily rely on intensity variations for example edge detection or pattern recognition. The conversion can be done using several methods, but a common approach is to calculate the weighted average of the red, green, and blue. Bilateral filtering is a non-linear, edge-preserving, and noise-reducing smoothing filter for images. It replaces the intensity value of each pixel with a weighted average of intensity values from nearby pixels. The weight calculation considers both the spatial distance and the intensity difference between the center pixel and its neighbors. This dual consideration allows the filter to preserve edges, as pixels with significantly different intensity values will receive low weights, ensuring that the edge information is not lost during filtering.

Subsequently, histogram equalization was applied to improve contrast and normalize the intensity distribution. The main idea behind histogram equalization is to transform the image in such a way that the resulting distribution of pixel intensities becomes approximately uniform across the entire intensity range. For our feature extraction the preprocessed image was flattened to create a one-dimensional feature vector. This feature vector was then standardized using the same scaler used during training. PCA transformation was applied to the standardized feature vector to project it onto the eigenfaces orthonormal basis, which was obtained during the training phase.

**Results:**

For the 1NN benchmark comparison of using eigenfaces and using the raw image data, the eigenfaces method was far quicker, requiring only 0.918 seconds to predict. Furthermore, we reached an accuracy score of 0.713 on the testing set. The original image data is very high dimensional compared to the eigenfaces. As a result, the execution time for creating predictions takes 69.039 seconds, which is nearly 69 times as long. The accuracy score was 0.628 for the testing set. So the eigenfaces-based method surpasses the conventional method in terms of both accuracy and execution time. The dimensionality reduction obtained by PCA allows the eigenfaces-based technique to produce a faster prediction time. Applying PCA to the dataset allows for the most information to be retained while capturing the most important features and compressing the data into a lower-dimensional space. By reducing the amount of features in the model, this compression also lowers the computational complexity and why it is so much quicker than the regular method.

With runtime and accuracy improvements found in our benchmark case, we looked at our other models for the best performing model The results of the accuracy of the models are shown below:

| Eigenface Model: | 1NN | KNN | Random Forest | Logistic Regression | SVM |
| --- | --- | --- | --- | --- | --- |
| Best Accuracy: | 0.713 | 0.737 | 0.672 | 0.797 | 0.827 |
| Best Considered Hyperparameter Combinations: | N/A | K = 5 | # trees = 87,  Max depth = 16 | Penalty = L2,  Multinomial dist,  C = 0.234 | Kernel = rbf,  Gamma = scale,  Degree = 2,  C = 0.95 |

Eigenfaces SVM performed the best with 0.827 testing accuracy. Surprisingly, Random Forest performed the worst with 0.672. With this in mind, we then used the models with said parameters to try and recognize our processed faces from the external validation set.

We found that for our external validation set, we had much poorer results with an accuracy rating of just 0.31 for the best performing model actually being the 1NN classifier. The results on the external validation set are found here:

| Eigenface Model: | 1NN | KNN | Logistic Regression | SVM |
| --- | --- | --- | --- | --- |
| Accuracy on External Validation Set: | 0.312 | 0.143 | 0.171 | 0.257 |

**Related Implementations:**

Looking into other people have done with our data set on kaggle [5] had similar results to our project when they use classic CNN algorithm for facial recognition. Their accuracy for their model was around 80% which was just a little better than our model, however they did not include a function or process photos that can be pulled form the internet. It does give insight into other ways to approach the problem we are talking and using the same dataset.

For this particular implementation from kaggle [6] , the model's basis model for extracting features from the input images is the Xception architecture with pre-trained ImageNet weights. Since Xception is a strong and well-known architecture, using it as the base model may result in greater performance than using a customized or more straightforward CNN design. To train this model, you would need to use pairs of face photos with labels indicating whether they are of the same person (similar) or two different people (dissimilar). For the same individual and for different people, the model would learn to produce feature vectors that are near together in the feature space. Which is highly more effect than our current model implementation.

In the study [4], our results were actually significantly better than of this published report. When using eigenvalues with PCA it seems are model does do very well with the trained dataset of LFW but in comparison to other models for facial recognition, are model seems to fall short. In conclusion, supervised learning, input-output pairs of labeled training data are given to the model. A label indicating whether the input pair is similar or dissimilar is produced by Siamese networks when a pair of pictures or data points are used as the input and output, respectively.

**Discussion:**  
 The results of our project were insightful for many reasons. We achieved a model performance of 0.82 accuracy with SVM when cross validated with LFW data. However, when cross validated with the external validation set, the testing accuracy reduced to 0.257. For all models, cross validation with LFW data was significantly higher but 1NN actually performed the best on the external validation set. These facts point to the complexities of the models for why 1NN performs the best. The 1NN algorithm is arguably the simplest model for classification. More complex models tend to overfit more easily and notably the preprocessing of our data on the external validation set is likely not the same as the LFW data. So what is likely occurring is that the preprocessing differences in the LFW and external validation set are resulting in models that are overtrained for LFW and not generalizable to images outside LFW or preprocessed exactly like LFW. This exemplifies the key importance of preprocessing and its effects on eigenface methods.

It is also curious to find that for our benchmark comparison of 1NN on eigenfaces and 1NN on raw data, the eigenfaces had improvements for both accuracy and runtime. The runtime improvement is not surprising given the lower dimensional feature space. With a lower dimensional feature space, the runtime complexity of ML algorithms decreases. However, what is interesting is that accuracy increased. This is especially interesting given that we chose a value of k that explains 80% of the variance in our feature matrix. If our feature space has less variance explained, then how can it perform better? PCA is completely invariant to our response variable, so it actually is completely possible to improve accuracy since PCA is a procedure that is completely ignorant of our response variable. Specifically, accuracy can improve when many features are uninformative to predicting the response variable. Since our features are pixels of images, many pixels won't help in recognizing faces since face images will always have pixels pertaining to non facial features. Since we select our k principal components based on the proportion of their explained variance, any n - k principal components we remove will be the least explanative n-k principal components to the variance in our feature matrix. This shows a powerful property of PCA for facial recognition. PCA with an appropriate level of K can essentially (but not literally):

* Perform feature selection (reducing feature space dimensionality)
* Make our models more generalizable (less complex models)
* Establish characteristic features of facial images

There are many aspects of the project that could be improved for future work in theory. Preprocessing differences leading to overfitting on LFW preprocessing is likely why our model failed to generalize well to outside data. We also could try to control for lighting variation in our eigenspace. A typical approach for this is to actually remove the first three principal components which are sometimes attributable to lighting variation. This can improve the model but this is not guaranteed. However, any improvements to the accuracy of our models would require us to know more on the preprocessing and data retrieval methods used which we did not have access to at the time of doing this project. Eigenfaces is one of the earliest methods for facial recognition and overall, the performance of our models was relatively strong given the inherent limitations of the method and comparisons with other implementations of it.

**Discussion on Other Methods**

In research paper [2], which was revised in 2019, a comprehensive analysis is presented on leading corporations and their advancements in facial recognition technology. The research also addresses certain limitations in the training data currently being utilized LFW, which contains merely 13,233 images. In contrast, other models, such as Google's FaceNet, employ not only the LFW dataset but also YouTubeDB and the proprietary Google Face Data, granting access to over 200 million photographs and yielding an accuracy rate exceeding 99% for their model. Regrettably, the Google Face Dataset remains inaccessible to the public for research purposes; however, it offers valuable insights into potential improvements for our model, emphasizing the necessity of training on considerably larger datasets.

[2] asserts, "Deep learning has already been integrated into most state-of-the-art facial recognition pipelines. This shift has led to a massive increase in the accuracy of facial recognition systems and has caused the current 'standard' benchmark for face recognition, LFW, to become saturated. In addition, the data requirements for deep networks highlight the need for a new, very large scale (tens of millions of images), public dataset for face recognition research." This statement reinforces the lack of extensive datasets available for enhancing the accuracy of our model through training.

**Conclusion:**

Using the dataset Labeled Faces in the Wild (LFW), we evaluated the performance of the Eigenfaces approach for facial identification throughout this study. Our investigation shows that the Eigenfaces technique, which uses Principal Component investigation (PCA) for dimensionality reduction, performs better than the traditional method that doesn't use PCA. We found that the eigenface method had both accuracy and runtime improvements in our benchmark case. These results highlight the benefits of utilizing PCA and Eigenfaces for face recognition tasks, as it enables the model to focus on the most crucial features while discarding less important variations.

We also observed the accuracy decreased to just 31% with our best model when used with our own input photographs for a unique face detection program. The resolution of the input samples, noise, or model overtraining could all be blamed for this decline in performance. Despite these difficulties, this project has offered insightful knowledge on the benefits of Eigenfaces and facial recognition technologies.

In conclusion, the Eigenfaces technique has demonstrated potential in enhancing the precision and speed of facial recognition tasks when combined with PCA. Even though the performance on the customized input images fell short of expectations, more study and model development could produce outcomes that perform better in practical applications. Future research in facial recognition technology can benefit from the lessons learnt from this experiment, which will also improve knowledge of Eigenfaces and its potential advantages over competing methods.

**References:**

[1] Serengil, S. I. (n.d.). retina-face: RetinaFace: Deep Face Detection Framework in TensorFlow for Python. PyPI. Retrieved April 25, 2023, from https://pypi.org/project/retina-face/

[2] Balaban, S. (2015). Deep learning and face recognition: the state of the art. In arXiv preprint arXiv:1902.03524.

[3] "Python | Grayscaling of Images using OpenCV." GeeksforGeeks. (2019, April 15). <https://www.geeksforgeeks.org/python-grayscaling-of-images-using-opencv/>.

[4] A Study about Principle Component Analysis and Eigenface for Facial Extraction. (2019). In Journal of Physics: Conference Series (Vol. 1196, No. 1, p. 012010). IOP Publishing. <https://iopscience.iop.org/article/10.1088/1742-6596/1196/1/012010/pdf>

[5] Face detection using CNN with the LFW dataset. (n.d.). Kaggle.com. Retrieved April 26, 2023,<https://www.kaggle.com/code/jake126/face-detection-using-cnn-with-the-lfw-dataset>

[6] Face Detection using Siamese Networks. (n.d.). Kaggle.com. Retrieved April 26, 2023, from https://www.kaggle.com/code/stoicstatic/face-detection-using-siamese-networks

**Appendix:**

https://colab.research.google.com/drive/1yPkiWiiNWTMwxF7PnLN-pY9oXjVVwGLs?usp=share\_link

Eigenfaces 1NN Method:

* Execution time: 0.865 seconds
* Accuracy score: 0.713’

Regular Method:

* Execution time: 69.577 seconds
* Accuracy score: 0.628

Trest Training Results:

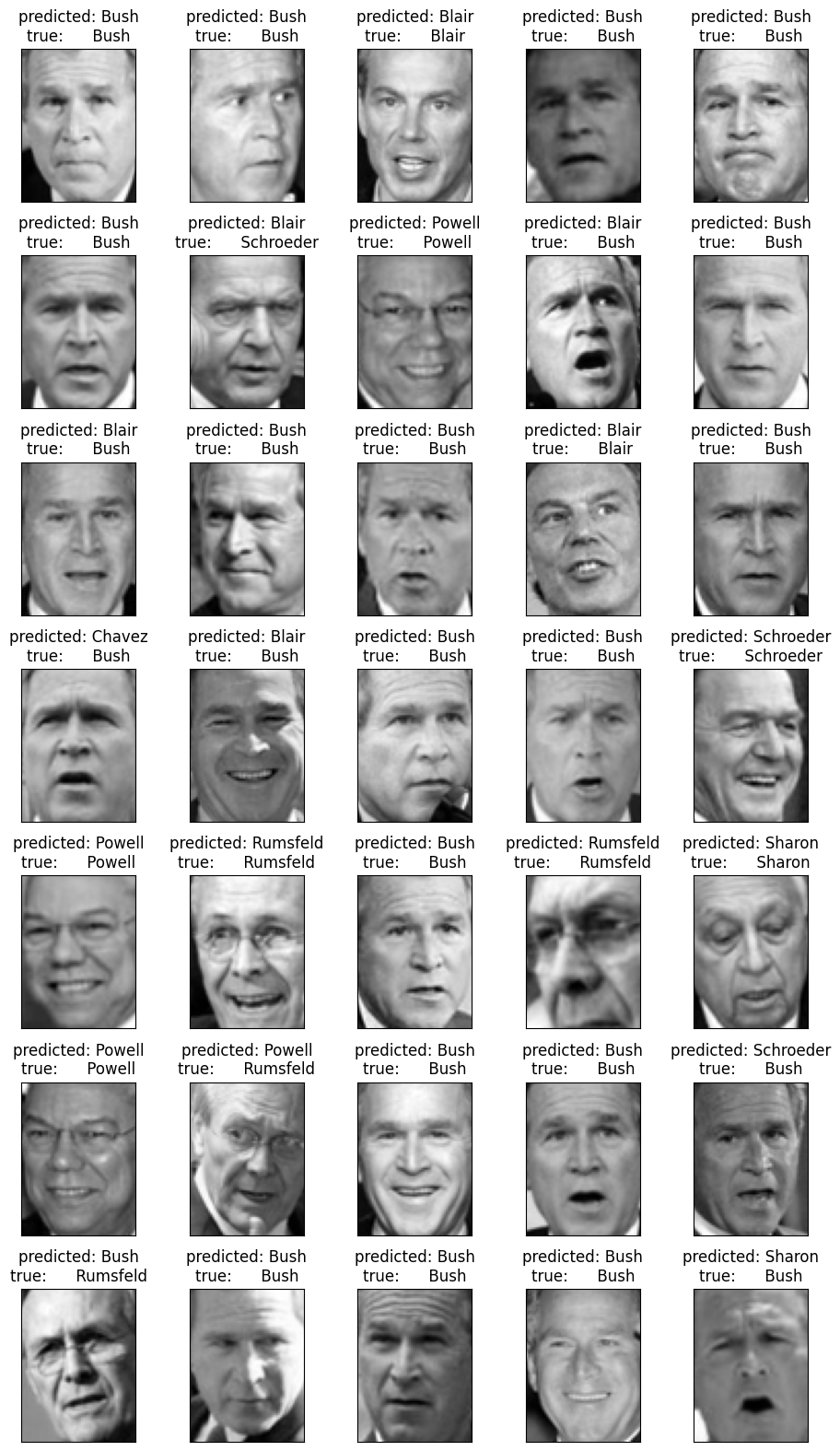


Photo Input Prediction:



**Statement of Contributions:**

Brendon

* LFW PCA and EDA
* LFW ML modeling
* Presentation
* Report

David

* Validation data preprocessing
* General functions for plotting and combining code
* Presentation
* Report

‌