Principal Component Analysis for Facial Recognition

Author: Henry J. Peters

Supervisor: Dr. Keshav R. Acharya

Embry-Riddle Aeronautical University

Abstract

The purpose of this experiment is studying the data science technique, Principal Component Analysis. We learn how to apply it to facial recognition in python to help create a deeper understanding of the technique when used on large data sets. Facial recognition has grown exponentially, driven by complex techniques such as artificial intelligence. There remains a need for simple, low-cost systems that can compare faces to pre-constructed data sets. Here we utilize the Principal Component Analysis (PCA) to process photos from the Yale Face Database.[3] These facial scans have numerous identifying variables so they are linearly reduced into a small matrix so that we can efficiently compare faces. We took the facial scan and processed it into large vectors. Then placed those vectors into a large matrix where each column represents one of the scanned faces, and each row represents a number pixel in the image. This matrix helps find the average face in the form of a column vector so that it can be deducted from the original face, this is to save unnecessary comparisons. The PCA reduces the matrix to reduce the image but keep most of the important data. We find the eigen values and their corelated eigen vectors (eigen faces) and use them as identifying values. For projections of a person whose face is altered from the original image in the data set we must compute the projection of their vector onto the set of eigen vectors obtained previously and use the mean face to reconstruct the missing attributes. This method is simple, fast, and low cost. It's intended for police stations, hospitals, or even surveillance cameras for the average homeowner.

Principal Component Analysis

Invented in 1901 by Karl Pearson, it was intended as an analogue for the principal axis theorem.[1] It's been used for many various fields of mathematics. Signal processing, empirical orthogonal functions, spectral decomposition, and more. The Principal Component Analysis in data science is a method of dimension reduction for data analysis, it prioritizes in data sets with large amounts of factors that contribute to some desired investigation.[2] Typically, the most important attributes or variables are used to create a correlation model. This is problematic because it loses out on so many other components that could have important contributing data. The PCA utilizes a covariance matrix that has "n" eigen vectors and "n" eigen values that are orthogonal to each other. The eigen vectors are the attributes or components that are important, the eigen values corresponding to those vectors describe its significance. There are thousands of pixels in each photograph and not all of them are necessary when we compare them. The PCA takes all contributing variables and combines them into Principal Components, new factors that generalize groups of data that prove to be the most significant. With just 2-3 Principal Components we can generalize most of the contributing variables.

Conclusion

The project successfully utilized the PCA to digest and prioritize a large set of images. To further this project and test how well it preforms one could do a percent recognition by using projections of altered version of images that are in the data set. For example, an image with glasses or maybe a hat would create an incomplete view of the persons face. The image would be processed all the same previous faces, then the mean is subtracted just as before so that we can center the image. By projecting the subtracted vector onto the previously calculated eigenvectors, we effectively decompose the partial face image into its principal components. Afterwards we add the mean face back to reconstruct the original image. The product is the altered image put back into its semi original state, no hat, no glasses.

Results

Before the covariance matrix is calculated we calculate the mean face. The mean face is the average face for all members of the data set. To save the PCA some time and energy we find obvious common attributes among all the faces; all participants have facial features such as a nose, mouth, and eyes. This is done by calculating the average face and subtracting it form all faces. In python *image_arrays_centered = image_arrays_reshaped - mean_Face.reshape(-1,1)*

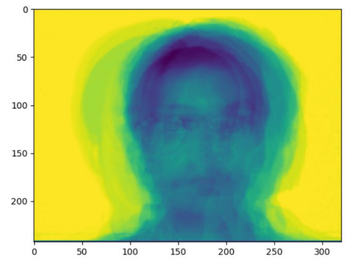

Figure 2 – The mean of all the faces within the data set.

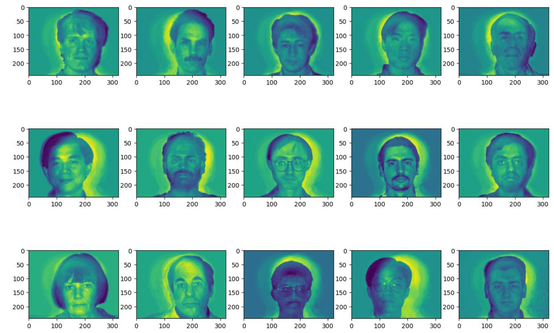

Figure 3 – The mean face was taken out of all the faces individually; yellow highlights represent the deductions while the dark green is what's left.

Each row of matrix P corresponds to feature or variable of the image. There are 77760 rows because there are that many pixels/variables per image. In order to become a Covariance Matrix, we must preform $P^T P$, bringing it from a 15x77760 matrix to a 15x15 matrix. In python it calculated buy suing the np.dot function. (*covariance_matrix = np.dot(P.T, P)*). From that 15x15 Covariance Matrix we can obtain the eigen values and eigen vectors that are shown below. To do this we use a built-in function *values, vectors = np.linalg.eig(covariance_matrix)*.

Eigenvalue	Value
λ_1	2.01933017×10^3
λ_2	5.18507280×10^2
λ_3	3.86963134×10^2
λ_4	-4.73480476×10^2
λ_5	2.93452751×10^2
λ_6	2.53106060×10^2
λ_7	-4.11755056×10^2
λ_8	-3.55059801×10^2
λ_9	-3.05239456×10^2
λ_{10}	1.52289602×10^2
λ_{11}	8.11369114
λ_{12}	-1.85017269
λ_{13}	-1.79265950×10^2
λ_{14}	-9.61179568×10^1
λ_{15}	-1.15017045×10^2

Figure 4 – These are the eigen values from the Covariance Matrix. These represent the importance of their corelated column vector.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
-0.24896	0.387287	-0.47833	0.390533	-0.13197	-0.4041	0.096834	0.391617	0.12598	-0.03581	0.081605	-0.19326	0.007149	-0.04506	0.003928
-0.25282	0.301811	-0.21833	-0.17224	-0.14283	0.407637	-0.18785	-0.02415	0.192271	0.588541	0.041545	0.228538	-0.08637	0.140826	0.283944
-0.18527	-0.2541	-0.29394	-0.39448	-0.26649	-0.24768	-0.06993	-0.3052	0.023178	0.028069	0.284023	-0.25934	-0.19135	0.348347	-0.34854
-0.23143	0.146386	0.334613	0.116933	-0.28522	0.206321	0.10799	-0.28144	0.575287	-0.12408	-0.01314	-0.2865	-0.07349	-0.34564	-0.17638
-0.25144	-0.51988	0.040423	-0.17472	-0.32405	-0.07313	0.0961	0.279713	-0.08495	0.09533	0.204631	-0.05519	0.148405	-0.4263	0.414158
-0.24372	0.379167	0.114907	-0.11972	-0.22107	-0.02788	0.162621	-0.25779	-0.25758	-0.3366	0.314405	0.341335	0.46947	0.087239	0.061108
-0.2571	0.187308	-0.01195	-0.09202	0.360548	-0.18942	-0.49079	-0.23011	-0.27775	0.063906	0.116657	-0.03445	-0.14861	-0.55175	-0.08926
-0.34379	-0.03951	0.460122	0.395127	0.0065	-0.05577	-0.11669	-0.03666	-0.24049	0.009567	0.139111	-0.19993	-0.32886	0.415956	0.309633
-0.25165	-0.20589	0.088353	0.220387	0.30534	-0.11927	0.477993	-0.04341	0.078732	0.374102	0.22774	0.412436	-0.06704	-0.05673	-0.3525
-0.28297	-0.21178	-0.33273	0.336319	-0.20438	0.483973	0.01203	-0.10282	-0.42398	-0.11307	-0.30814	-0.00342	0.007875	-0.08021	-0.2656
-0.23616	0.004632	-0.08089	-0.1484	0.527407	0.214138	0.235908	-0.03216	0.004765	0.078816	0.01171	-0.55806	0.43884	0.122944	0.097161
-0.30724	-0.05292	-0.02123	-0.13924	0.014914	-0.41552	0.126005	-0.28598	0.092039	0.003563	-0.72184	0.150168	0.020372	0.056531	0.238178
-0.18321	0.135297	0.411582	-0.21607	-0.18176	-0.07478	-0.16734	0.471435	-0.15436	0.252563	-0.25512	-0.07316	0.231977	0.089119	-0.4726
-0.27836	-0.31405	-0.05819	0.122653	0.206489	0.049015	-0.50856	0.159843	0.434547	-0.31603	0.058117	0.278237	0.262311	0.185977	-0.05784
-0.2733	0.123472	-0.01333	-0.41743	0.191939	0.215728	0.248639	0.362818	-0.00405	-0.4324	-0.02501	0.116203	-0.50758	-0.00613	0.001916

Figure 5 - Each column is the column vectors obtained from the covariance matrix. These columns are the principal components of the data. Column 0 is related to eigen value λ_1, column 1 is related to λ_2 and so on.

Each column vector is a principal component, PC, of the data set. Some principal components are less valuable than o0th0ers, only contributing small amounts of data that would prove helpful in distinguishing faces apart. To find the most valuable PC's that so that we can prioritize we look at the eigen values. The largest are the most significant and the least significant are the values closest to zero.

Henry Peters
E-mail: Petersh@my.erau.edu
Ph: 845-796-8979

References:

- **[1] P. Belhumeur, J. Hespanha, D. Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1997, pp. 711-720. http://cvc.cs.yale.edu/cvc/projects/yalefaces/yalefaces.htm**
- **[2] Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points in Space". Philosophical Magazine. 2 (11): 559–572**
- **[3] Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. Philosophical transactions. Series A, Mathematical, physical, and engineering sciences, 374(2065), 20150202. https://doi.org/10.1098/rsta.2015.0202**