Images and Dimensionality Reduction (PCA)

Francesco Villi
francesco.villi@edu.unifi.it

Abstract

This project aims to explore the use of PCA in image retrieval; in particular without using image directly but extracting the feature from VGGFace and then applying dimensionality reductions to manage high dimensional data and investigate whether dimensionality reduction yields improvements in image retrieval accuracy.

1. Introduction

The data set used is "Labeled Faces in the Wild", after extracting the images with at least 70 samples I applied normalization and standardization according to the data given by the test set and applied also to the test set. Therefore I resized normalized and standardized the images and then gave them in input to a VGGFace to extract features. The distance used in Faiss index to retrieve elements is the L2 distance. I'm going to compare retrieval without reduction and a reduction maintaining the first 20, 100, 300 components and some tests without the first 2-3 components because these components are focused on low frequencies and I expect that they will not have an impact in discriminating images.

1.1. Principal Component Analysis

Principal component analysis (PCA) is a linear dimensionality reduction technique; the data is linearly transformed onto a new coordinate system such that the directions (principal components) capturing the largest variation in the data can be easily identified.

The principal components of a collection of points in a real coordinate space are a sequence of *p* unit vectors, where the *i-th* vector is the direction of a line that best fits the data while being orthogonal to the first *i-1* vectors.

The process of PCA involves several steps, including standardizing the data, computing the covariance matrix, calculating the eigenvalues and eigenvectors of this matrix, and finally, transforming the original data into the new coordinate system defined by the principal components. Practically the steps are:

- Given sample
$$D = \{x_1, \ldots, x_n\}, x_i \in \mathbb{R}^n$$
- Compute:
$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$
$$\Sigma = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^T$$
- Compute eigenvalues and eigenvectors of Σ, where:
$$\Sigma = \Phi\Lambda\Phi^T,$$
$$\Lambda = diag(\sigma_1^2, \ldots, \sigma_n^2),$$
$$\Phi^T\Phi = I$$
- Order eigenvalues $\sigma_1^2 > \ldots > \sigma_n^2$
- Select K eigenvalues and eigenvectors

1.2. Faiss library

Faiss is a powerful and efficient library designed for similarity search and clustering of high-dimensional vectors. It supports various distance metrics like L2 (Euclidean distance) and IP (inner product):

- **L2 distance** (Euclidean distance) measures the straight-line distance between two vectors in the vector space avoiding the square root due to speed reasons; so the result is an approximation:
$$A, B \in R^n$$
$$\text{L2 Dist.} = \sum_{i=1}^{n}(A_i - B_i)^2$$
- **Inner product distance** (IP), a metric commonly used in machine learning and information retrieval tasks. The norm of the query vectors does not affect the ranking of results (of course the norm of the database vectors does matter).In our case, this is cosine similarity, due to the vectors are normalized.
$$A, B \in R^n$$
$$\text{IP Dist.} = \mathbf{A} \cdot \mathbf{B} = \cos\theta$$

2. Project structure

- **Get face images**: Using LFW datasets from sklearn I downloaded all images in RGB where the lowest number of samples was 55, this dataset has 9 different faces.
- **Balancing data**: I addressed the dataset imbalance by implementing a balancing technique to mitigate the disparity between the underrepresented and overrepresented classes by selecting the minimum number of available samples for the underrepresented class.
- **Split dataset**: I partitioned the dataset into training and testing sets. Subsequently, I computed the mean and standard deviation of the training set.
- **Pixel standardization**: I scaled pixel values to have a zero mean and unit variance using the mean and std computed at the previous step
- **Extracting features**: I used VGGFace without the 3 fully connected layers at the top of the network. So for each sample, I got a tensor with shape (7, 7, 512).
- **Reduction and Testing**: I applied PCA reductions to the features extracted from VGGFace. I conducted tests comparing the original data to datasets subjected to PCA reductions, with preservation of 20, 50, and 100 initial components. Additionally, variations were explored by excluding 1, 2, or 3 of the first components out of the total 30.
  Then I tested the prediction in the recognition task varying the dataset dimension and

3. Equivalence of Cosine Similarity and L2 Distance for Normalized Vectors

Let $\mathbf{a}$ and $\mathbf{b}$ be normalized vectors, i.e., $\|\mathbf{a}\| = \|\mathbf{b}\| = 1$.

Cosine Similarity with Normalized Vectors:

$$\text{cosine\_similarity}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|} = \mathbf{a} \cdot \mathbf{b}$$

L2 Distance with Normalized Vectors:

$$\begin{aligned}\text{L2\_distance}(\mathbf{a}, \mathbf{b}) &= \|\mathbf{a} - \mathbf{b}\|_2 \\ &= \sqrt{(\mathbf{a} - \mathbf{b}) \cdot (\mathbf{a} - \mathbf{b})} \\ &= \sqrt{\mathbf{a} \cdot \mathbf{a} - 2\mathbf{a} \cdot \mathbf{b} + \mathbf{b} \cdot \mathbf{b}}\end{aligned}$$

3.1. Substitute $\|\mathbf{a}\| = \|\mathbf{b}\| = 1$:

$$\begin{aligned}&= \sqrt{1 - 2(\mathbf{a} \cdot \mathbf{b}) + 1} \\ &= \sqrt{2 - 2(\mathbf{a} \cdot \mathbf{b})}\end{aligned}$$

$$\text{cosine\_similarity}(\mathbf{a}, \mathbf{b}) = \mathbf{a} \cdot \mathbf{b}$$

$$\text{L2\_distance}(\mathbf{a}, \mathbf{b}) = \sqrt{2 - 2(\mathbf{a} \cdot \mathbf{b})}$$

We observe that both expressions are equivalent. Hence, for normalized vectors, cosine similarity is indeed equivalent to L2 distance.

4. Results and Discussion

4.1. Retrieval task

I'm going to compare results with varying numbers of components and removing 1,2,3 first eigenvectors. I'll use only IP distance due to the prop. 3.

ID	$P@10 - STD$				
	ALL	20	50	100	396
0	**1.00**	**1.00**	0.99	0.99	**1.00**
1	**0.93**	0.79	0.81	0.82	0.81
2	0.93	**0.94**	0.92	0.91	0.91
3	**0.86**	0.78	0.75	0.77	0.75
4	**0.88**	0.86	0.84	0.81	0.81
5	**0.94**	0.92	0.93	0.93	0.92
6	0.97	**0.98**	0.97	0.95	0.95
7	0.99	**1.00**	0.99	0.98	0.99
8	**0.85**	0.83	0.83	0.84	0.83
AVG	**0.92**	0.90	0.89	0.88	0.88

Table 1. Table displaying rounded P@10 scores for different PCA with 20,50,100,396 and 7x7x512 components with image standardization

ID	$P@10$				
	ALL	20	50	100	396
0	**0.99**	0.95	0.95	0.95	0.95
1	**0.92**	0.76	0.76	0.77	0.78
2	**0.91**	0.85	0.86	0.86	0.86
3	**0.82**	0.78	0.78	0.78	0.77
4	0.84	0.86	**0.87**	0.86	0.84
5	0.86	**0.91**	0.90	**0.91**	0.90
6	**0.95**	0.94	0.94	**0.95**	**0.95**
7	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
8	**0.84**	0.80	0.81	0.80	0.80
AVG	**0.90**	0.87	0.87	0.87	0.87

Table 2. Table displaying rounded Precision@10 scores for different PCA with 20,50,100,396 and 7x7x512 components without image standardization

Upon comparing tables 1 and 2, it becomes evident that employing image standardization yields notably improved

outcomes. Notably the algorithm demonstrates superior performance in both tables where ALL components are maintained.
However, it's worth highlighting that even in scenarios where resource constraints such as limited memory and CPU capacity are prevalent, good results are achieved applying PCA, in particular using only 20 components as evidenced by table 2. In this case, observing the result with ID 1 and 3 we notice that by simply applying PCA we get worse results in retrieving those images.

Then I decided, after applying PCA, to remove the first 1/2/3 components out of 20/50/396 as we can see in tables 9 and 10. Here the best results are archived without image standardization and removing the first 2 components, but we under-perform the previous approach.
In particular, results are worse than those in table 2 and 1. An observation removing firsts components we get worse results in retrieval face 1 but improvement in retrieval face 2; meaning that those first components are fundamental to correct recognize face 2 but they bring some noise for retrieval face 2.

STD PCA red.	P@5	P@10	P@20	P@ALL
20	0.93	0.90	0.82	**0.34**
50	0.93	0.89	0.82	0.28
100	0.93	0.89	0.81	0.26
396	0.93	0.89	0.80	0.27
0	**0.94**	**0.92**	**0.83**	0.14
-1/20	0.91	0.86	0.75	0.28
-2/20	0.90	0.86	0.77	0.26
-3/20	0.90	0.85	0.77	0.26
-1/50	0.91	0.86	0.74	0.24
-2/50	0.92	0.85	0.76	0.22
-3/50	0.91	0.85	0.75	0.21
-1/396	0.91	0.85	0.72	0.22
-2/396	0.92	0.85	0.74	0.20
-3/396	0.91	0.85	0.71	0.19

Table 3. Feature extracted has been standardized before to extract features and apply PCA.Precision as a mean value between all classes at different cut-off values for PCA components. 0 means no reduction

In tables 3 and 4 there is a comparison between all previous methods with different precisions (5,10,20,ALL). We notice that the best results are achieved by the test without PCA reduction excluding the case when we have set recall to 1 for retrieving elements, in that case, PCA with 20 components reaches the best performance.
Considering only PCA the best results are achieved using only 20 components, making this a good alternative if we don't have many computational resources.

PCA red.	P@5	P@10	P@20	P@ALL
20	0.90	0.87	0.79	**0.28**
50	0.91	0.87	0.78	0.25
100	0.92	0.87	0.77	0.24
396	0.92	0.87	0.77	0.24
0	**0.94**	**0.90**	**0.80**	0.13
-1/20	0.89	0.85	0.73	0.23
-2/20	0.90	0.88	0.81	0.30
-3/20	0.86	0.83	0.73	0.24
-1/50	0.90	0.85	0.71	0.21
-2/50	0.91	0.88	0.81	0.26
-3/50	0.88	0.83	0.70	0.20
-1/396	0.91	0.84	0.68	0.20
-2/396	0.91	0.87	0.78	0.24
-3/396	0.90	0.83	0.66	0.19

Table 4. Precision as a mean value between all classes at different cut-off values for PCA components. 0 means no reduction

4.2. Image recognition task

I analyzed face recognition using K-Nearest Neighbors (KNN) alongside Principal Component Analysis (PCA). For KNN, a parameter of $k = 9$ is utilized, and in the event of a tie, the selection is made based on the summation of distances grouped by label, with preference given to the closest distance.

Using a balanced dataset, we can observe the results presented in Table 5 and 6, it is evident that the highest accuracy is attained without PCA. Moreover, it is noteworthy that standardized images outperform non-standardized images.
Also using unbalanced dataset results are a little bit better than the previous case as we can see from table 7.
Eventually, I tested the algorithm setting $min\_faces\_per\_person = 30$, balancing the dataset and applying standardization to the images.
In this case, we have 34 different faces instead of 8. The accuracy gets a little bit worse wrt. the previous case but we achieved satisfactory results, in particular without PCA as we can notice from table 8.

5. Conclusion

In this study, we explored the extraction of features using VGGFace and the application of Principal Component Analysis (PCA) to reduce dimensionality. Through our experiments, we observed varying outcomes dependent on different configurations and techniques applied.

Firstly, employing image standardization notably enhanced the performance of our algorithm, as evidenced by the comparison between tables 1 and 2. Maintaining all components consistently demonstrated superior results across different scenarios.

However, when resource constraints such as limited

ID test	Accuracy w/ n components				
	ALL	20	50	100	396
test 1	**0.985**	0.971	0.964	0.967	0.975
test 2	**0.989**	0.960	0.964	0.975	0.978
test 3	**0.978**	0.971	0.971	0.964	0.971
test 4	**0.989**	0.946	0.953	0.957	0.967
test 5	**0.982**	0.943	0.950	0.960	0.960
test 6	**0.982**	0.950	0.957	0.950	0.957
test 7	**0.982**	0.946	0.953	0.960	0.957
test 8	**0.982**	0.957	0.960	0.967	0.971
test 9	**0.989**	0.960	0.971	0.975	0.978
AVG	**0.984**	0.956	0.960	0.964	0.968

Table 5. Accuracy in face recognition with different train-test splits and using KKN with k=9

ID test	Accuracy w/ n components STD				
	ALL	20	50	100	396
test 1	0.979	**0.989**	**0.989**	0.969	0.979
test 2	**0.989**	0.949	0.939	0.959	0.969
test 3	**1.000**	0.989	0.989	**1.000**	0.989
test 4	0.989	0.989	0.989	**1.000**	0.979
test 5	**1.000**	0.979	0.989	0.989	**1.000**
test 6	**0.989**	0.949	0.959	0.959	0.969
test 7	**0.979**	0.969	0.969	0.969	0.969
test 8	0.989	**1.000**	0.979	0.989	**1.000**
test 9	**0.979**	**0.979**	**0.979**	**0.979**	0.969
AVG	**0.988**	0.977	0.976	0.979	0.980

Table 6. Accuracy in face recognition with different train-test splits and using KKN with k=9. All images have been pre-processed with standardization and a balanced dataset

ID test	Accuracy w/ n components STD Unb.				
	ALL	20	50	100	396
test 1	**0.996**	0.971	0.978	0.975	0.978
test 2	0.989	0.985	0.989	**0.992**	**0.992**
test 3	0.992	**1.000**	**1.000**	0.996	**1.000**
test 4	0.992	0.989	**0.996**	**0.996**	**0.996**
test 5	**0.992**	0.975	0.978	0.978	0.982
test 6	**0.992**	0.967	0.978	0.982	0.989
test 7	**0.989**	0.967	0.975	0.975	0.978
test 8	**0.989**	0.975	0.978	0.978	0.982
test 9	**0.989**	0.971	0.975	0.975	0.985
AVG	**0.991**	0.978	0.983	0.983	0.987

Table 7. Accuracy in face recognition with different train-test splits and using KKN with k=9. All images have been pre-processed with standardization using an unbalanced dataset

memory and CPU capacity were taken into consideration, PCA offered a viable solution. Notably, employing only 20 components in PCA yielded commendable results, particularly in scenarios where computational resources were scarce, as indicated in tables 2 and 1.

Further experimentation involved the removal of the first

ID test	Accuracy w/ n components STD Unb.				
	ALL	20	50	100	396
test 1	0.965	0.901	0.941	0.960	0.960
test 2	0.970	0.911	0.926	0.936	0.950
test 3	0.990	0.931	0.960	0.960	0.970
test 4	0.980	0.931	0.950	0.960	0.965
test 5	0.975	0.955	0.965	0.960	0.960
test 6	0.985	0.965	0.960	0.965	0.980
test 7	0.980	0.926	0.975	0.980	0.985
test 8	0.990	0.936	0.965	0.970	0.980
test 9	0.980	0.960	0.955	0.960	0.975
AVG	**0.979**	0.934	0.955	0.961	0.970

Table 8. Accuracy in face recognition with different train-test splits and using KKN with k=9. All images have been pre-processed with standardization using a balanced dataset and min_faces_per_person=30

1/2/3 components out of 20/50/396 after applying PCA. Interestingly, the best results were achieved without image standardization and by removing the first 2 components. However, it's noteworthy that this approach underperformed compared to the previous method, as observed in tables 9 and 10. This suggests that while removing certain components can reduce noise, it can also lead to a degradation in overall performance.

Moreover, our comparison of different precision values (5, 10, 20, and all) across various PCA configurations highlighted the effectiveness of employing fewer components, particularly 20 components, in achieving satisfactory results while conserving computational resources. In tables 3 and 4, a comparison between all previous methods with different precisions (5,10,20,ALL) further corroborates these findings. The best results are generally achieved without PCA reduction, except in cases where recall is set to 1 for retrieving elements, where PCA with 20 components reaches optimal performance.

Considering only PCA, the best results are consistently achieved using only 20 components, making this a good alternative when computational resources are limited.

In addition to feature extraction, we analyzed face recognition using K-Nearest Neighbors (KNN) alongside PCA. With a parameter of $k = 9$ for KNN and tie-breaking based on the summation of distances grouped by label.
Standardized images generally outperformed non-standardized images, as indicated by the results in Table 5 and 6. Even with an unbalanced dataset, the results were slightly improved, as seen in Table 7.
Furthermore, balancing the dataset and applying standardization to the images with $min\_faces\_per\_person = 30$ resulted in 34 different faces instead of 8, with a slight decrease in accuracy compared to the previous case. However, satisfactory results were still achieved, particularly without PCA, as evident from Table 8. In conclusion,

ID	$STD - P@10$								
	$-1/20$	$-1/50$	$-1/396$	$-2/20$	$-2/50$	$-2/396$	$-3/20$	$-3/50$	$-3/396$
0	**1.00**	0.96	**1.00**	0.92	**0.99**	0.98	**0.96**	0.92	0.92
1	0.65	**0.67**	0.65	0.64	0.64	**0.72**	**0.67**	0.64	0.64
2	0.90	**0.92**	0.90	0.94	**0.96**	0.92	0.92	**0.94**	**0.94**
3	0.72	**0.76**	0.72	**0.79**	0.77	0.76	0.76	**0.79**	**0.79**
4	**0.81**	**0.81**	**0.81**	0.84	**0.85**	0.81	0.81	**0.84**	**0.84**
5	**0.91**	0.88	**0.91**	**0.88**	0.87	0.87	**0.88**	**0.88**	**0.88**
6	**0.95**	0.91	**0.95**	0.92	0.91	**0.93**	0.91	**0.92**	**0.92**
7	0.98	**0.99**	0.98	0.98	**1.00**	**1.00**	**0.99**	0.98	0.98
8	**0.80**	0.79	**0.80**	**0.79**	**0.79**	**0.79**	**0.79**	**0.79**	**0.79**
AVG	0.85	0.85	0.85	0.85	**0.86**	**0.86**	0.85	0.85	0.85

Table 9. Table displaying rounded Precision@10 scores for different PCA settings; "-X/K" denotes the removal of the first X out of K components in PCA reconstruction.

ID	$P@10$								
	$-1/20$	$-1/50$	$-1/396$	$-2/20$	$-2/50$	$-2/396$	$-3/20$	$-3/50$	$-3/396$
0	0.95	0.95	0.95	**0.98**	0.97	0.97	0.89	0.88	0.86
1	0.66	**0.70**	0.65	0.67	**0.70**	**0.70**	0.58	0.59	0.58
2	0.84	0.84	0.86	0.95	0.94	0.91	**0.96**	**0.96**	0.93
3	0.76	0.75	0.72	**0.82**	0.81	0.81	0.74	0.75	0.72
4	0.86	0.85	0.83	0.88	0.87	**0.89**	0.87	0.88	0.88
5	0.87	0.85	0.84	**0.89**	0.88	0.86	0.86	0.86	0.85
6	0.94	0.94	0.94	0.94	**0.96**	0.95	0.88	0.93	0.90
7	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
8	0.79	0.77	0.78	0.79	0.79	**0.80**	0.74	0.74	0.75
AVG	0.85	0.85	0.84	**0.88**	**0.88**	**0.88**	0.84	0.84	0.83

Table 10. Feature extracted has been standardized before to extract features and applying PCA. "-X/K" denotes the removal of the first X out of K components in PCA reconstruction.

our findings underscore the importance of considering both computational efficiency and performance trade-offs when applying feature extraction techniques such as PCA in facial recognition systems. Additionally, the impact of preprocessing steps such as image standardization can significantly influence algorithm performance.