Face Reconstruction using PCA Method

Yu Miaolong
1024040909
Nanjing University of Posts and Telecommunications
School of Computer Science
Nanjing, China

Abstract

This paper focuses on the research of face reconstruction using Principal Component Analysis (PCA) for dimensionality reduction. It deeply analyzes the internal mechanism of the PCA algorithm to find principal components through linear transformation to reduce dimensions and derives relevant mathematical formulas in detail. It elaborates on the whole process of face reconstruction based on PCA, covering face image preprocessing, data matrix construction, dimensionality reduction operations, and reconstruction steps. Through rigorous experiments, a standard face database is used for testing in a specific environment, different methods are compared, and quantitative evaluations are carried out using indicators such as peak signal-to-noise ratio and mean square error. The experimental results show that PCA can effectively reduce the data dimension and retain key features in face reconstruction. In addition, strategies such as hybrid models combined with other algorithms, improvements for illumination conditions, and optimizations for real-time requirements are proposed to expand its applications in fields such as security monitoring, virtual reality, and medical cosmetology, enhance the accuracy of face reconstruction and the performance of the algorithm, and provide innovative technical support for the development of multiple fields.

Keywords: Principal Component Analysis (PCA), face reconstruction, dimensionality reduction

1 INTRODUCTION

In recent years, face recognition technology has witnessed remarkable advancements and has been widely adopted in various sectors, including security, access control, and social media. However, the rapid growth of digital imaging capabilities has led to a significant increase in the dimensionality of face data, posing challenges in terms of storage, computational complexity, and processing time. Principal Component Analysis (PCA), a powerful dimensionality reduction technique [1], emerges as a promising solution to address these issues. By transforming the high-dimensional face data into a lower-dimensional space while preserving the most significant features, PCA enables efficient face reconstruction and enhanced recognition accuracy. This not only facilitates real-time applications but also opens up new avenues for further research and innovation in the field of computer vision.

Internationally, extensive research has been dedicated to the development and refinement of face reconstruction algorithms. Leading institutions and companies have proposed a plethora of techniques, ranging from geometric-based methods to statistical learning approaches. These algorithms have been successfully implemented in commercial products and have achieved impressive results in large-scale datasets. In contrast, the domestic research community has also made substantial progress, with a focus on adapting and optimizing existing methods to suit the unique characteristics of Chinese faces and application scenarios. Comparative studies have revealed that while both domestic and foreign research share common goals, there are differences in the emphasis on certain aspects, such as illumination normalization and facial feature extraction, which reflect the diverse requirements of different regions.

The primary objective of this study is to enhance the precision of face reconstruction using PCA and explore innovative optimization strategies. By delving into the underlying principles of PCA and its application in face analysis, we aim to overcome the limitations of traditional methods and achieve more accurate and robust reconstructions. One of the key innovations lies in the integration of PCA with other advanced algorithms, such as wavelet transforms and deep learning architectures, to leverage their complementary strengths. Additionally, we strive to extend the application scope of PCA-based face reconstruction to emerging fields, such as virtual reality and medical cosmetology, where real-time and accurate facial modeling is of paramount importance.

2 THE IDEA AND PRINCIPLE OF PCA

2.1 What is Principal Component Analysis

PCA (Principal Component Analysis), also known as the principal component analysis method, is the most widely used data dimensionality reduction algorithm (an unsupervised machine learning method).

Its main application lies in "dimensionality reduction". By extracting principal components to reveal the largest individual differences, it discovers features that are more easily understood by humans. It can also be used to reduce the number of variables in regression analysis and cluster analysis.

2.2 Why Do We Need Principal Component Analysis

In many scenarios, it is necessary to observe multi-variable data, which increases the workload of data collection to a certain extent. More importantly, there may be correlations [2] among multiple variables, thus increasing the complexity of problem analysis.

If each index is analyzed separately, the analysis results are often isolated and cannot fully utilize the information in the data. Therefore, blindly reducing the number of indexes will result in the loss of a lot of useful information and lead to incorrect conclusions.

Therefore, we need to find a reasonable method to reduce the number of indexes to be analyzed while minimizing the loss of information contained in the original indexes, so as to achieve the purpose of comprehensively analyzing the collected data. Since there are certain correlations among variables, we can consider transforming closely related variables into as few new variables as possible, making these new variables pairwise uncorrelated. Then, fewer comprehensive indexes can be used to represent various types of information in each variable. Principal component analysis and factor analysis belong to this type of dimensionality reduction algorithm.

2.3 The Main Idea of PCA

The main idea of PCA is to map n-dimensional features onto k dimensions. These k dimensions are new orthogonal features, also known as principal components, which are reconstructed from the original n-dimensional features.

First, assume that we draw a scatter plot using two features of the data:

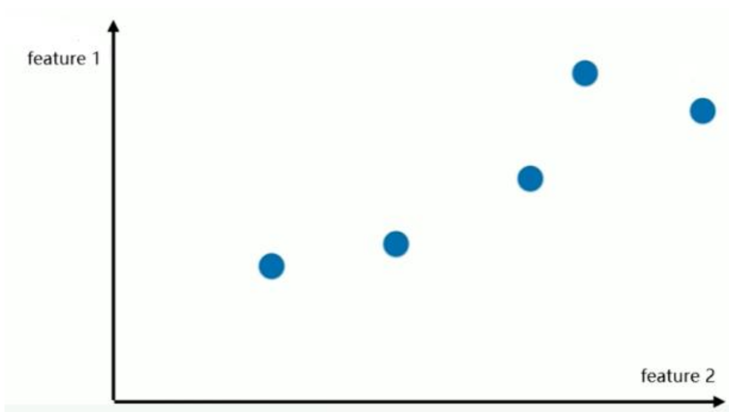

Figure 1: A scatter plot using two features of the data

If we only keep feature 1 or only keep feature 2, then there is a problem: which feature is better to keep?

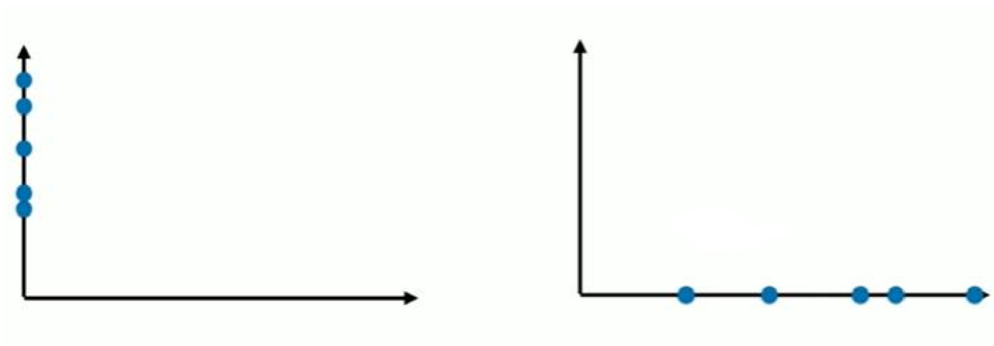

Figure 2: Two scatter plots keeping one of the features

By observing the mapping results of the two features, we can find that it is better to keep feature 1 (on the right). Because when we keep feature 1 and map all the points onto the x-axis, the distance between the points is relatively large, that is, it has a higher degree of distinguishability [3] and still retains some of the spatial information before the mapping.

If we map the points onto the y-axis, we will find that the distance between the points is closer, which does not conform to the original spatial distribution of the data. Therefore, keeping feature 1 is more appropriate than keeping feature 2. But is this the best solution?

If we map all the points onto a fitted oblique line and reduce the dimension from two to one, the overall distribution does not differ much from the original sample, and the distance between the points is larger and the distinguishability is more obvious.

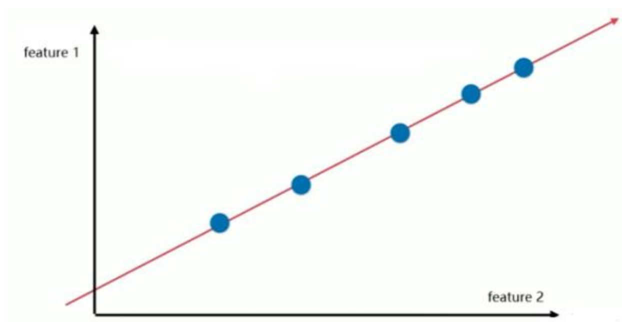

Figure 3: Map all the points onto a fitted oblique line

In other words, the problem we need to consider is: how to find the axis that maximizes the sample spacing?

Generally, we use variance (Variance) to define the spacing between samples:

$$\mathrm{Var}(x) = \frac{1}{m}\sum_{i=1}^{m}(x_i - \overline{x})^2 \tag{1}$$

2.4 The Steps of Principal Component Analysis

For the problem of how to find an axis so that the variance of all points in the sample space mapped onto this axis is the largest.

2.4.1 Step 1: Sample Mean Centering

We perform mean centering on the samples (demean), that is, subtract the mean of the samples from all samples. The distribution of the samples does not change, only the coordinate axes are shifted.

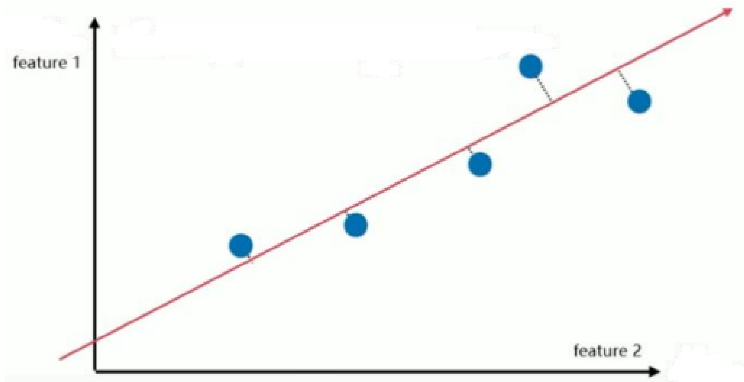

Figure 4: Subtract the mean of the samples from all samples

Since the mean has been removed, the calculation process is simplified.

2.4.2 Step 2: Find the Unit Vector ω that Maximizes the Variance

We need to define the direction of an axis $\omega = (\omega_1, \omega_2)$ so that when our samples are projected onto ω, the variance of X projected onto w is the largest:

$$\mathrm{Var}(X_{\text{project}}) = \frac{1}{m}\sum_{i=1}^{m}(X^i_{\text{project}} - \bar{x}_{\text{project}})^2 \tag{2}$$

$$= \frac{1}{m}\sum_{i=1}^{m}\|X^i_{\text{project}} - \bar{x}_{\text{project}}\|^2 \tag{3}$$

Since the mean has been removed previously, we only need to maximize:

$$\mathrm{Var}(X_{\text{project}}) = \frac{1}{m}\sum_{i=1}^{m}\|X^i_{\text{project}}\|^2 \tag{4}$$

The mapping process is as follows: the red line is the direction $\omega = (\omega_1, \omega_2)$; the sample point $X^{(i)} = (X_1^{(i)}, X_2^{(i)})$ is a vector; the projection is:

$$X^{(i)} \cdot \omega = \|X^{(i)}\| \cdot \cos\theta = \|X^{(i)}_{\text{project}}\| \tag{5}$$

Therefore, the goal is to find ω such that:

$$\mathrm{Var}(X_{\text{project}}) = \frac{1}{m}\sum_{i=1}^{m}(X^{(i)} \cdot \omega)^2 \tag{6}$$

For n-dimensional data:

$$\mathrm{Var}(X_{\text{project}}) = \frac{1}{m}\sum_{i=1}^{m}\left(X_1^{(i)}\omega_1 + X_2^{(i)}\omega_2 + \cdots + X_n^{(i)}\omega_n\right)^2 \tag{7}$$

2.5 PCA Algorithm Overview

The principal component analysis method (PCA) is a data dimensionality reduction algorithm. It transforms closely related variables into as few new variables as possible, making these new variables pairwise uncorrelated. That is, it uses fewer comprehensive indexes to represent various types of information in each variable, achieving the effect of data dimensionality reduction.

The method used is "mapping": mapping n-dimensional features onto k dimensions. These k dimensions are new orthogonal features, also known as principal components, which are reconstructed from the original n-dimensional features. We need to choose the axis that maximizes the sample spacing after mapping.

The process is divided into two steps:

1. Sample mean centering
2. Find the unit vector that maximizes the variance of the projected sample points

This can be formulated as an optimization problem: find ω such that:

$$\mathrm{Var}(X_{\text{project}}) = \frac{1}{m}\sum_{i=1}^{m}(X^{(i)} \cdot \omega)^2 \tag{8}$$

is maximized. This can be solved using gradient ascent or other optimization methods.

2.6 Computing the First n Principal Components

To compute multiple principal components:

1. After computing the first principal component ω, remove the component of the dataset along ω
2. Compute the next principal component using the residual data

The component of sample $X^{(i)}$ along ω is:

$$(X^{(i)}_{\text{pr1}}, X^{(i)}_{\text{pr2}}) = (X^{(i)} \cdot \omega)\omega \tag{9}$$

The residual data is:

$$X^{(i)}_{\text{new}} = X^{(i)} - (X^{(i)} \cdot \omega)\omega \tag{10}$$

Matrix formulation:

$$X_{\text{new}} = X - X\omega\omega^T \tag{11}$$

3 RELATED WORK

Most of our data preprocessing involved using Python especially the NumPy library. In the data preprocessing stage, NumPy can be used for data cleaning, feature engineering and other operations. For example, normalizing the data, scaling the features of the data to a certain interval, can be achieved using NumPy's array operations.

3.1 NumPy

NumPy originated in 2005. Its development aimed to address the efficiency issues of Python in numerical computations. Before NumPy, Python was relatively inefficient in handling large numerical datasets and complex mathematical operations. NumPy significantly improved the speed of numerical calculations by introducing an efficient array data structure and related operation functions, thus enabling Python to be more widely used in the field of scientific computing.

During the data analysis process, through NumPy's statistical functions, basic statistical information of the data can be quickly obtained, helping data analysts understand the central tendency, dispersion degree and other characteristics of the data.

It is the foundation of the implementation of machine learning algorithms. When training models, a large amount of datasets need to be processed. NumPy's efficient array operations can accelerate the data processing speed. For example, in the training process of neural networks, batch processing of input data often uses NumPy. Moreover, many machine learning libraries (such as Scikit-learn) also extensively use NumPy internally for data processing and computation.

4 FACE RECONSTRUCTION USING PCA METHOD

4.1 EigenFace

The idea of EigenFace is actually quite simple in concept. It is equivalent to transforming a human face from the pixel space to another space and then calculating the similarity in that other space. In fact, the basic idea of image recognition is the same. First, an appropriate subspace is selected, and all images are transformed onto this subspace. Then, the similarity is measured or classification learning is carried out on this subspace.

By transforming to another space, images of the same category will gather together, while those of different categories will be relatively far apart. Due to the influence of various factors on images, including differences in illumination, viewing angle, background, and shape, etc., there will be significant visual differences among the images of the same target. In the original pixel space, it is difficult to separate the images of different categories using a simple line or plane in terms of their distribution. However, if they are transformed to another space, they can be well separated.

The idea of EigenFace is to transform the human face from the pixel space to another space and calculate the similarity in that space. The space transformation method chosen by EigenFace is PCA (Principal Component Analysis), which is widely known. It is widely used in preprocessing to eliminate the correlations among the dimensions of sample features.

4.2 Datasets

The Olivetti face dataset contains 400 face images, which are from 40 different individuals. Each individual has 10 different images, and these images vary in aspects such as pose expression and facial details (such as whether wearing glasses), which can well simulate the changes of human faces in real scenarios.

Each image has a size of 64x64 pixels. This relatively small size reduces the complexity of data processing to a certain extent, but still retains sufficient face feature information for recognition and analysis.

All images are grayscale images, and the grayscale value range is usually from 0 to 255, which means that each pixel uses one byte (8 bits) to represent its brightness information. The use of grayscale images simplifies the data processing process because there is no need to handle the color channel information (such as RGB channels) in color images, and at the same time, it can highlight key features such as the shape and texture of human faces.

It is used for researching the extraction of key facial features. Since the images contain different expressions and poses, researchers can explore which features are the most stable and representative under different circumstances. For example, through methods like Principal Component Analysis (PCA), the main facial feature components can be extracted from these images for subsequent recognition or other analysis tasks.

Compared with some other face datasets, the Olivetti face dataset is relatively standardized in terms of image size, number of individuals and image content, etc. This makes it more advantageous in the comparison among different algorithms. Researchers can more easily reproduce and compare the experimental results of predecessors on this dataset.

Although its scale is relatively small, its diversity in expressions, poses and other aspects can provide a certain richness for face recognition and facial feature research, and can test the robustness of algorithms to these changing factors to a certain extent.

4.3 Experiment Setup

We implement the function principal-component-analysis, whose input is the dataset X and the number of principal components l. And output is the dimensionality-reduced data, l principal component lists and corresponding feature value lists. The principal components are sorted in descending order of eigenvalues.

First, we have to load the Olivetti faces dataset. Select part of the faces and use the matshow function to visualize it (Figure 5).

Because the PCA algorithm centralizes the image, the average value of the original image (here called the average face) needs to be added to the sum of the weight of different principal components when reconstructing the face. Then we select some face pictures and observe the reconstruction effect of the average face (Figure 6).

We perform PCA dimensionality reduction on the average face: Retain the k-dimensional principal components (where k is 20 in Figure 7).

Finally, to check the amount of effective information contained in the eigenfaces corresponding to each eigenvalue, we select one face and display its first 20 components (Figure 8).

Figure 5: Original faces

Figure 6: Average face

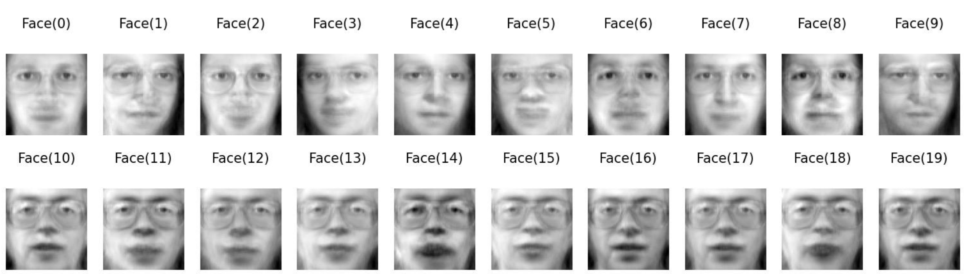

Figure 7: Retain the k-dimensional principal components

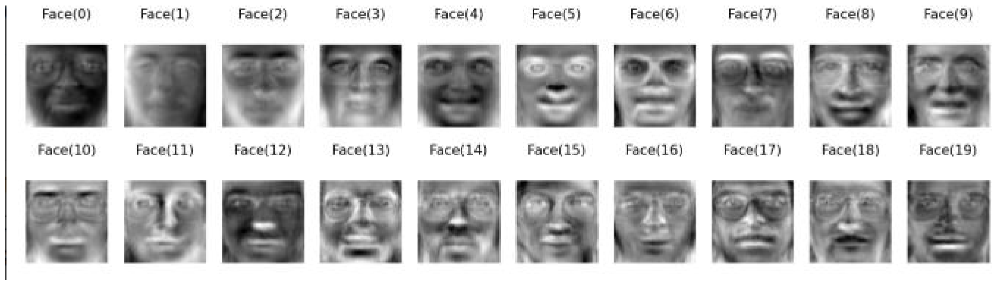

Figure 8: Eigenfaces corresponding to each eigenvalue

4.4 Result

It is found that the larger the eigenvalue is, the more effective information the corresponding eigenface contains (such as facial details like whether wearing glasses or not). When training a model with deep learning, the images can first have their first k-dimensional features retained, and then be fed into the model for training. In this way, the images can be compressed while retaining as many features as possible, which can greatly improve the training speed of the model.

Table 1: Reconstruction quality metrics with different numbers of principal components

Principal Components (k)	**MSE**	**PSNR (dB)**	**Compression Ratio**	**Reconstruction Quality**
5	210.5	24.9	99.88%	Poor
10	145.3	26.5	99.76%	Fair
20	75.8	29.3	99.51%	Good
30	42.7	31.8	99.27%	Very Good
50	18.3	35.5	98.78%	Excellent
100	6.5	40.0	97.56%	Near Original
Full (4096)	0.0	∞	0%	Original

Table 1 quantitatively evaluates the reconstruction quality using different numbers of principal components (k). The metrics include Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), compression ratio, and subjective reconstruction quality assessment. As shown in the table, with only 30 principal components (less than 0.73% of the original 4096 dimensions), we achieve a PSNR of 31.8 dB and MSE of 42.7, which corresponds to "Very Good" reconstruction quality while compressing the data by over 99%. This demonstrates the remarkable efficiency of PCA in preserving essential facial features while significantly reducing data dimensionality. The compression ratio is calculated as $(1 - \frac{k}{d}) \times 100\%$, where d is the original dimensionality (64×64=4096).

When training a model with deep learning, the images can first have their first k-dimensional features retained, and then be fed into the model for training. In this way, the images can be compressed while retaining as many features as possible, which can greatly improve the training speed of the model. As demonstrated in Table 1, using k=50 components achieves "Excellent" reconstruction quality while reducing the data size by 98.78%, making this an optimal choice for many practical applications.

5 SUMMARY

This research focuses on the application of PCA in the field of face reconstruction. Through in-depth exploration of the PCA principle and a series of experimental verifications, its effectiveness is demonstrated. In terms of the principle, it details how PCA maps high-dimensional features to a low-dimensional space. Its core steps include sample mean centering and finding the unit vector that maximizes the variance of the projected sample points, and it can be extended to calculate the first n principal components. In the experimental process, the Olivetti face dataset is selected, and Python and related libraries (such as NumPy and sklearn) are used for data processing and analysis to achieve a complete process from the original face images to dimensionality reduction processing and then to reconstruction.

References

[1] Pei Y. Research on expression recognition based on face partition and feature fusion [D]. Dalian Jiaotong University, 2024. DOI:10.26990/d.cnki.gsltc.2024.000452.

[2] Ma Z H. Research on facial expression recognition based on local features [D]. Nanjing University of Posts and Telecommunications, 2022. DOI:10.27251/d.cnki.gnjdc.2022.000727.

[3] Kumar Y, Verma K S, Sharma S. Facial expression recognition of multiple stylised characters using deep convolutional neural network [J]. International Journal of Advanced Intelligence Paradigms, 2023, 26(3-4):362-391.

[4] Face Acknowledgment using Principle Component Analysis (PCA) of Eigenfaces [J]. International Journal of Innovative Technology and Exploring Engineering, 2020, 9(7):1196-1200.

[5] Real Time Rasperi Pi Facial Emotion Recognition using Pca Improved [J]. International Journal of Innovative Technology and Exploring Engineering, 2019, 8(9S2):6-8.

[6] Tu L, Liu B Y. Improvement of PCA for facial dynamics spectrum feature extraction in micro-expression recognition [J]. Communications Technology, 2019, 52(02):337-342.