哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称：机器学习

课程类型：必修

实验题目：PCA模型实验

学号：1190200523

姓名：石翔宇

1 实验目的

实现一个 PCA 模型，能够对给定数据进行降维（即找到其中的主成分）。

2 实验要求及实验环境

2.1 实验要求

测试

1. 人工生成一些数据（如三维数据），让它们主要分布在低维空间中，如首先让某个维度的方差远小于其它唯独，然后对这些数据旋转。生成这些数据后，用你的 PCA 方法进行主成分提取；
2. 找一个人脸数据（小点样本量），用你实现 PCA 方法对该数据降维，找出一些主成分，然后用这些主成分对每一副人脸图像进行重建，比较一些它们与原图像有多大差别（用信噪比衡量）。

2.2 实验环境

Windows 11 + Python 3.7.8

3 设计思想

主成分分析（PCA）是一种常用的无监督学习方法，利用正交变换把线性相关变量表示的观测数据转换为少数几个由线性无关变量表示的数据，线性无关的变量称为主成分。

假设 $X=(x_1,x_2,\dots,x_m)^T$ 是 m 维随机变量，其均值向量是

$$\mu = E(x) = (\mu_1,\mu_2,\dots,\mu_m)^T \tag{1}$$

协方差矩阵是

$$\Sigma = cov(X,X) = E[(X-\mu)(x-\mu)^T] \tag{2}$$

考虑由 m 维随机变量 X 到 m 维随机变量 $Y=(y_1,y_2,\dots,y_m)^T$ 的线性变换

$$y_i = \alpha_i^T X = \alpha_{1i}x_1 + \alpha_{2i}x_2 + \cdots + \alpha_{mi}x_m \tag{3}$$

其中 $\alpha_i^T = (\alpha_{1i},\alpha_{2i},\dots,\alpha_{mi}), i=1,2,\dots,m$。

由随机变量的性质可知

$$E(y_i) = \alpha_i^T\mu, \quad i=1,2,\dots,m \tag{4}$$

$$var(y_i) = \alpha_i^T\Sigma\alpha_i, \quad i=1,2,\dots,m \tag{5}$$

$$cov(y_i,y_j) = \alpha_i^T\Sigma\alpha_j, \quad i=1,2,\dots,m \tag{6}$$

$$\tag{7}$$

若如式3所示的线性变换满足下列条件

1. 系数变量 α_i^T 是单位向量，即 $\alpha_i^T\alpha_i = 1, i=1,2,\dots,m$；

2. 变量 y_i 与 y_j 互不相关，即 $cov(y_i, y_j) = 0 (i \neq j)$；
3. y_i 是与 $y_1, y_2, \ldots, y_{i-1}$ $(i = 1, 2, \ldots, m)$ 都不相关的 X 的所有线性变换中方差最大的，这时分别称 $y_1, y_2, \ldots, y_m$ 为 X 的第一主成分、第二主成分、⋯、第 m 主成分。

我们将采用拉格朗日乘子法求出主成分。

首先求 X 的第一主成分 $y_1 = \alpha_1^T X$，即求系数向量 α_1。由主成分需要满足的条件可知，第一主成分的 α_1 是在 $\alpha_1^T \alpha_1 = 1$ 的条件下，X 的所有线性变换中使方差

$$var(\alpha_1^T X) = \alpha_1^T \Sigma \alpha_1 \tag{8}$$

达到最大的。

求解第一主成分就是求解优化问题

$$\max_{\alpha_1} \ \alpha_1^T \Sigma \alpha_1 s.t. \ \alpha_1^T \alpha_1 = 1 \tag{9}$$

定义拉格朗日函数

$$L_1 = \alpha_1^T \Sigma \alpha_1 - \lambda(\alpha_1^T \alpha_1 - 1) \tag{10}$$

其中 λ 是拉格朗日乘子。将 L_1 对 α_1 求导，并令导数为 0，得

$$\frac{\partial L_1}{\partial \alpha_1} = 2\Sigma\alpha_1 - 2\lambda\alpha_1 = 0 \tag{11}$$

因此，λ 是 Σ 的特征值，α_1 是对应的单位特征向量，

$$L_1 = \alpha_1^T \Sigma \alpha_1 - \lambda(\alpha_1^T \alpha_1 - 1) = \alpha_1^T \Sigma \alpha_1 - \lambda \alpha_1^T \alpha_1 + \lambda = \lambda \tag{12}$$

假设 Σ 的最大特征值 λ_1 对应单位特征向量 α_1，则 λ_1 和 α_1 是最优化问题的解

$$var(\alpha_1^T X) = \alpha_1^T \Sigma \alpha_1 = \lambda_1 \tag{13}$$

求解第二主成分需要求解约束最优化问题

$$\max_{\alpha_2} \ \alpha_2^T \Sigma \alpha_2 s.t. \ \alpha_2^T \alpha_2 = 1, \quad \alpha_1^T \Sigma \alpha_2 = 0, \quad \alpha_2^T \Sigma \alpha_1 = 0 \tag{14}$$

注意到

$$\alpha_2^T \Sigma \alpha_1 = \alpha_1^T \Sigma \alpha_2 = \lambda_1 \alpha_2^T \alpha_1 = \lambda_1 \alpha_1^T \alpha_2 = 0 \tag{15}$$

则定义拉格朗日函数

$$L_2 = \alpha_2^T \Sigma \alpha_2 - \lambda(\alpha_2^T \alpha_2 - 1) - \phi \alpha_2^T \alpha_1 \tag{16}$$

其中，λ 和 ϕ 是拉格朗日乘子。将 L_2 对 α_2 求导，并令导数为 0，得

$$\begin{aligned} \frac{\partial L_2}{\partial \alpha_1} = 2\Sigma\alpha_2 - 2\lambda\alpha_2 - \phi\alpha_1 &= 0 \\ 2\alpha_1^T \Sigma \alpha_2 - 2\alpha_1^T \lambda \alpha_2 - \alpha_1^T \phi \alpha_1 &= 0 \\ \phi \alpha_1^T \alpha_1 &= 0 \\ \phi &= 0 \end{aligned} \tag{17}$$

$$\Sigma\alpha_2 - \lambda\alpha_2 = 0$$

由此，λ 是 Σ 的特征值，α_2 是对应的单位特征向量，

$$\begin{aligned} L_2 =& \alpha_2^T \Sigma \alpha_2 - \lambda(\alpha_2^T \alpha_2 - 1) - \phi \alpha_2^T \alpha_1 \\ =& \alpha_2^T \Sigma \alpha_2 - \lambda \alpha_2^T \alpha_2 + \lambda \\ =& \lambda \end{aligned} \tag{18}$$

假设 Σ 的第二大特征值 λ_2 对应单位特征向量 α_2，则 λ_2 和 α_2 是最优化问题的解

$$var(\alpha_2^T X) = \alpha_2^T \Sigma \alpha_2 = \lambda_2 \tag{19}$$

按照上述放大求得第一、第二、直到第 m 主成分，其系数向量 $\alpha_1, \alpha_2, \ldots, \alpha_m$ 分别是 Σ 的第一个、第二个、直到第 m 个单位特征向量，$\lambda_1, \lambda_2, \ldots, \lambda_m$ 分别是对应的特征值。

4 实验结果分析

4.1 人工数据测试

我们生成了“瑞士卷”数据。“瑞士卷”数据是分布在三维空间的瑞士卷结构，正投影为漩涡状。我们设定数据量为 2000，噪声系数为 0.1，生成的数据如图 1所示。

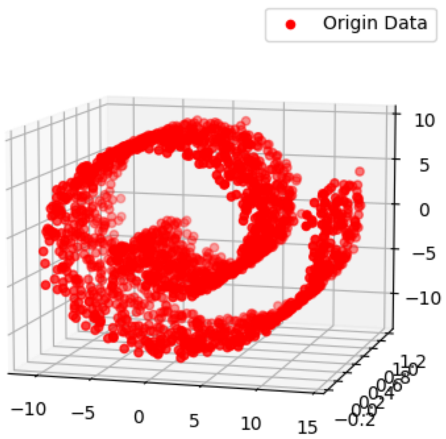

图 1

我们使用 PCA 算法将三维的“瑞士卷”结构降维到二维空间中，结果如图 2所示。

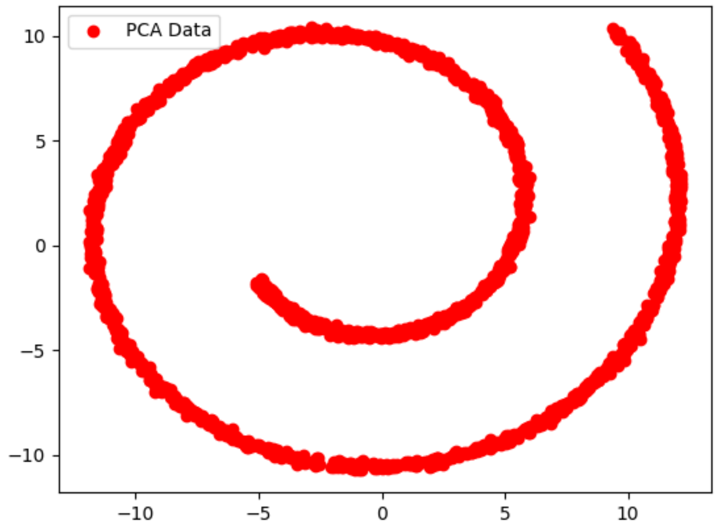

图 2

从上图我们可以看出，正投影后得到的形状是信息量最大的方向，于是我们得到了漩涡状的二维图形。

4.2 人脸数据测试

我们选取了三幅图片进行测试，分别将降维后的维度设为 $(30, 20, 10, 5, 4, 3, 2, 1)$，结果如图图 **3**、图 **4**、图 **5**所示。

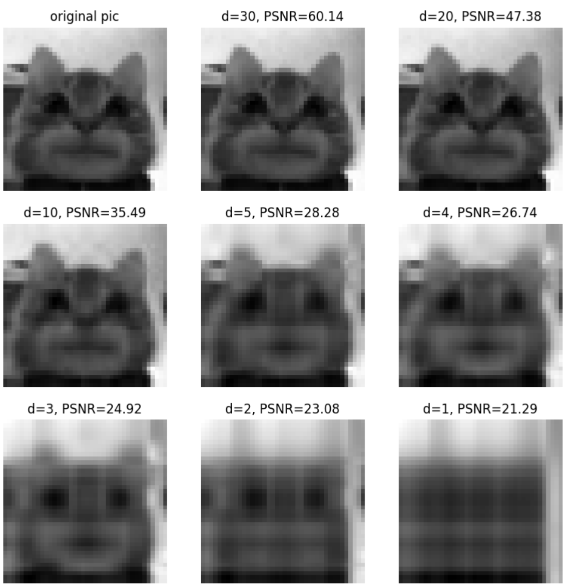

图 3

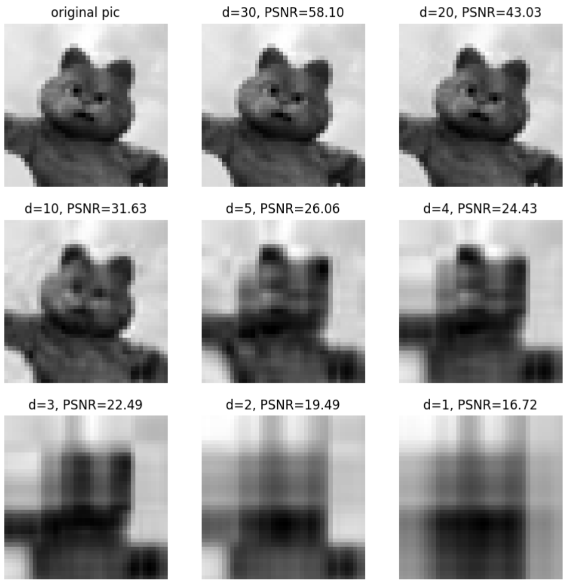

图 4

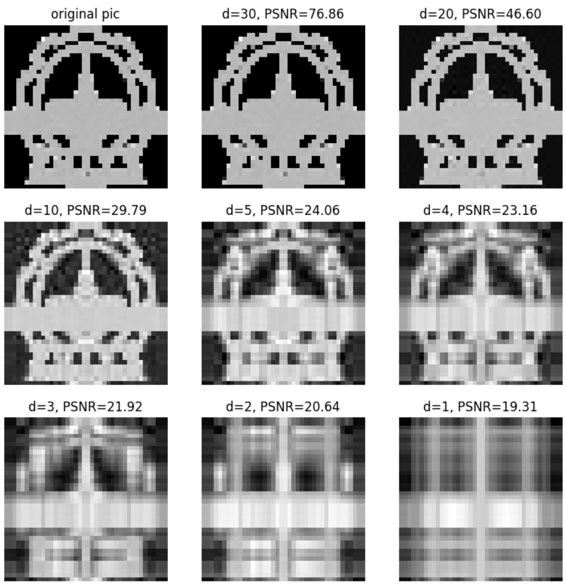

图 5

可以看出，在目标维度不小于 20 时，能够较好地保留图片特征，人眼几乎无法分辨损失；目标维度为 5 时，已经能够看出图片明显地损失；当目标维度降到 2 甚至 1 时，已经无法分辨出原来的图像。同时，信噪比随着目标维度的下降而下降。

5 结论

PCA 在降低数据的维度的同时，保留了主要信息，能够有效地应用于数据压缩、数据降维和图片特征提取等领域。但目标维度的选择对于 PCA 来说至关重要，需要仔细考虑，太高会使得压缩效果不显著，太低则会导致信息被过度压缩，丢失重要信息。

A 源代码（带注释）

A.1 生成数据

Listing 1: data.py

```python
import numpy as np

def generate_swiss_roll(n_sample=2000, noise=0.1, theta=70 * np.pi / 180):
    t = 1.5 * np.pi * (1 + 2 * np.random.rand(1, n_sample))
    x = t * np.cos(t)
    y = np.random.rand(1, n_sample)
    z = t * np.sin(t)
    X = np.concatenate((x, y, z))
    X += noise * np.random.randn(3, n_sample)
    rotate = np.array([[np.cos(theta), 0, np.sin(theta)], [0, 1, 0], [-np.sin(theta), 0, np.cos
        (theta)]])
    return np.dot(rotate, X).T
```

A.2 PCA

Listing 2: pca.py

```python
import numpy as np
import os
from PIL import Image

from data import *
from utils import *


def pca(x, k, rebuild=False):
    mu = np.mean(x, axis=0)
    cov = np.cov(x, rowvar=False)
    values, vectors = np.linalg.eig(cov)
    index = np.argsort(values)[: -(k + 1): -1]
    vectors = vectors[:, index]
    x_pca = (x - mu).dot(vectors)
    if rebuild:
        x_pca = x_pca.dot(vectors.T) + mu
```

```
    return x_pca


def show(x, x_pca):
    show_3d(x)
    show_2d(x_pca)


def pca_my_data():
    x = generate_swiss_roll()
    x_pca = pca(x, 2)
    show(x, x_pca)


def read_face(file_path, figsize=(40, 40)):
    file_list = os.listdir(file_path)
    data = []
    for i in range(len(file_list)):
        path = os.path.join(file_path, file_list[i])
        L = Image.open(path).resize(figsize).convert('L')
        data.append(np.array(L))
    return np.array(data)

def pca_face(file_path, figsize=(40, 40)):
    x = read_face(file_path, figsize=figsize)
    print(x.shape)
    n_samples = x.shape[0]
    d_decreased = [30, 20, 10, 5, 4, 3, 2, 1]
    for i in range(n_samples):
        plt.subplot(3, 3, 1), plt.title('original pic'), plt.axis('off')
        plt.imshow(x[i], cmap='gray')
        for j in range(len(d_decreased)):
            plt.subplot(3, 3, j + 2), plt.axis('off')
            x_pca = pca(x[i], d_decreased[j], rebuild=True)
            plt.title('d={}, PSNR={:.2f}'.format(d_decreased[j], psnr(x[i], x_pca)))
            plt.imshow(np.real(x_pca.reshape(figsize)), cmap='gray')
        plt.show()

if __name__ == '__main__':
    # pca_my_data()
    file_path = "./face"
    pca_face(file_path)
```

A.3 辅助代码

Listing 3: utils.py

```
from matplotlib import pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import numpy as np

def show_3d(x):
    ax = plt.gca(projection='3d')
    ax.scatter(x[:, 0], x[:, 1], x[:, 2], color="r", label='Origin Data')
    plt.legend()
    plt.show()

def show_2d(x):
    plt.scatter(x[:, 0], x[:, 1], facecolor='r', label='PCA Data')
    plt.legend()
    plt.show()

def psnr(source, target):
    rmse = np.sqrt(np.mean((source - target) ** 2))
    return 20 * np.log10(255.0 / rmse)
```