BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Prof. Dr.-Ing. Kristian Hildebrand

Visual and Scientific Computing
Datenerforschen und Eigenfaces mit PCA und SVD

WiSe 2019/20

Das Lernziel dieser Übungsaufgabe ist, am Beispiel der Gesichtserkennung, die Wichtigkeit und Allgemeingültigkeit von Principal Component Analyse und Singular Value Decomposition.

Aufgabe 1: Datenerforschung (3 Punkte)

Ziel der Aufgabe ist es eine Intuition für die Ergebnisse und Möglichkeiten der Hauptkomponentenanalyse (PCA) zu erlangen. Dazu ist in der Datei *pca.py* schon eine simple PCA-Implementierung vorgegeben. Diese extrahiert die zwei Hauptkomponenten eines Datensatzes und projiziert alle Datenpunkte in den sich ergebenen 2D-Raum. Der Datensatz besteht aus $(28, 28)$ Pixel großen Bilder von handgeschriebenen Dreien. Diese stammen aus dem sogenannten MNIST-Datatensatz, welcher häufig für Benchmarks benutzt wird. Beispiele aus dem Datensatz sind in Abbildung 1 zu sehen. Ihnen stehen Plotfunktionen und Projektionsfunktionen zur Verfügung. Da wir für die Aufgabe ein weiteres Machine Learning Python Paket verwenden, installieren Sie dieses bitte mit dem folgenden Kommando nach: *conda install scikit-learn*.

Ziel der Aufgabe ist es mithilfe der vorgegebenen Funktionen die Daten sowie die Hauptkomponenten zu erforschen, sodass folgende Fragen beantwortet werden können:

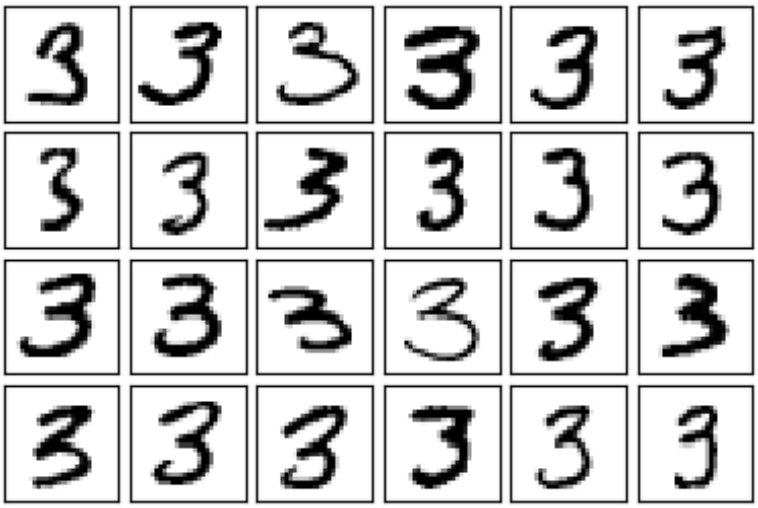

Abbildung 1: Beispiele aus dem MNIST-Datensatz.

1) Der Featurespace:

   a) Wie sieht das Bild der "1" (*one.jpg*) aus, wenn Sie zunächst die Dimensionen des Bildes auf 2 reduzieren und diese dann wieder in den ursprünglichen Raum zurück transformieren? Und wieso? Tipp: Denken Sie daran, dass die PCA nur auf Vektoren operiert, d.h. Sie müssen zunächst das Bild der "1" von einer Matrix in einen Vektor umwandeln.

   b) Wie sieht ein Bild einer "3" aus, wenn Sie zunächst die Dimensionen des Bildes auf 2 reduzieren und diese dann wieder in den ursprünglichen Raum zurück transformieren?

c) Im Bezug auf Klassifikation von Bildern im Allgemeinen: Wozu kann man diese PCA nutzen? Wie würden Sie das zum Beispiel umsetzen?

2) Bedeutung der Hauptkomponenten:

a) Welche Eigenschaft der Zahlen wird mit der ersten Hauptkomponente kodiert?. D.h. Welche Bedeutung hat es, wenn ein Datensatz einen großen negativen bzw. einen positiven Faktor für die erste Hauptkomponente besitzt. Tipp: Zeigen Sie sich die Hauptkomponente. Probieren Sie auch die Funktion *plot_interactive*.

b) Können Sie genauso eine Eigenschaft auch der zweiten Hauptkomponente zuschreiben? Wenn ja, welche wäre das?

c) Im Bezug auf Klassifikation von neuen Bildern einer Drei: Wozu kann man diese PCA nutzen?

Bitte beantworten Sie die Fragen textuell in einer separaten Textdatei. Sie können, müssen aber keinen Code abgeben.

Aufgabe 2: Eigenfaces (7 Punkte)

In der letzten Aufgabe war die PCA-Implementierung schon vorgegeben, in dieser Aufgabe soll die PCA mithilfe der SVD (Singulärwertzerlegung) selbst gefunden werden. Dazu soll die Datei *eigenfaces.py* vervollständigt werden und damit der Eigenface-Algorithmus zur Gesichtserkennung implementiert werden.

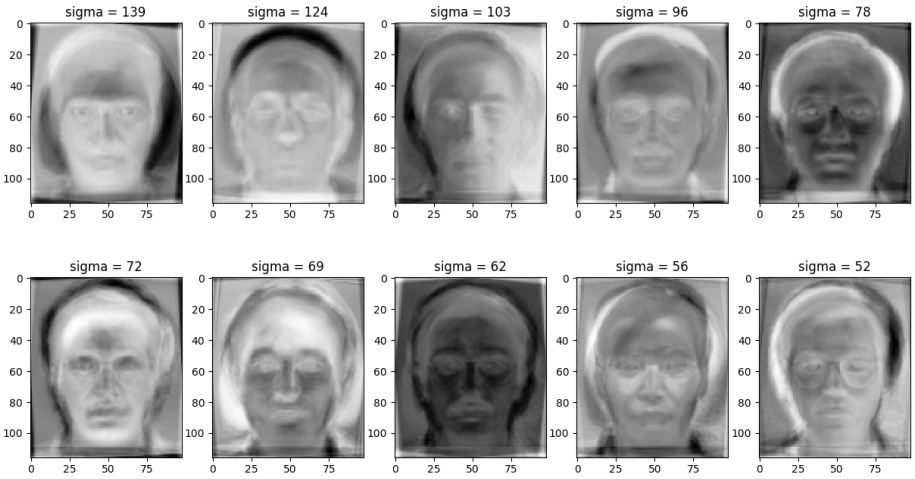

Abbildung 2: Ergebnis der Aufgabe (c)

a) **Laden Sie die Trainingsbilder.** Implementieren Sie die Funktion *load_images()*, welche Bilder aus einem übergebenen Verzeichnis laden soll.

b) **Erstellen Sie für die Liste von geladenen Bildern die dazugehörige Datenmatrix**. Implementieren Sie die Funktion *setup_data_matrix*. Die Bilder sollen dabei die Zeilen der Matrix bilden. Somit müssen die Bilder zunächst in einen Vektor umgewandelt werden.

c) **Berechnen Sie die Hauptkomponenten der Daten.** Implementieren Sie die Funktion *calculate_svd()*. Diese soll mithilfe der Singulärwert-Zerlegung der zuvor erstellten Datenmatrix die Eigenfaces für den Trainingsdatensatz berechnen. Die ersten zehn Eigenfaces sollen mit Hilfe der Funktion *visualize_eigen_faces()*, welche in *lib.py* zur Verfügung gestellt wird, wie in Abbildung 2 dargestellt werden.

d) **Entfernen Sie die Basisvektoren.** Berechnen Sie dazu den Index k, sodass 80% der totalen akkumulierten Magnitude der Singulärwerte in den ersten k Hauptkomponenten enthalten ist. Implementieren Sie dazu *accumulated_energy()*. Stellen Sie das Ergebnis mithilfe der Funktion *plot_singular_values_and_energy()*, welche in lib.py zur Verfügung gestellt wird, graphisch dar (Abb. 3). Entfernen Sie nun alle Basisvektoren, bis auf die ersten k, aus der Basis.

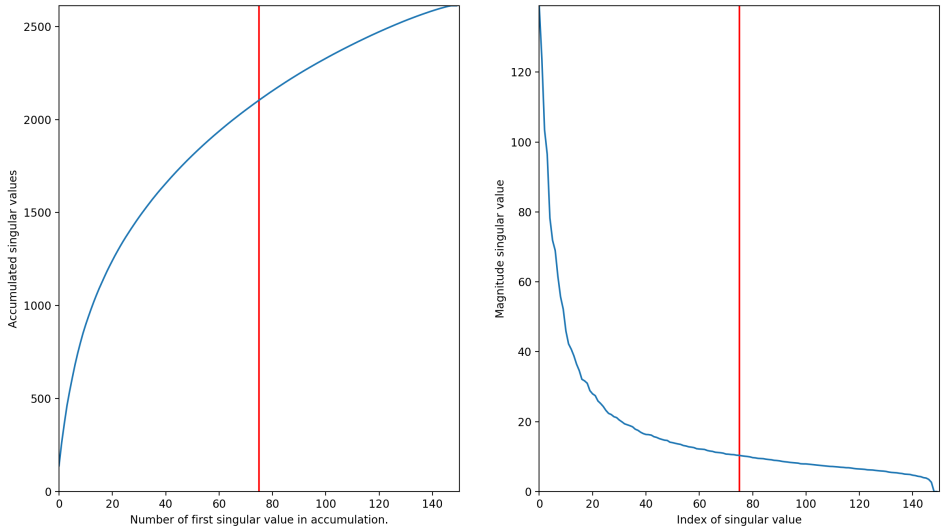

Abbildung 3: Ergebnis der Aufgabe (d)

e) **Projizieren Sie die Trainingsdaten in den gefundenen k-dimensionalen Raum.** Implementieren Sie in der Funktion *project_faces()* die Projektion der übergebenen Bilder in den Raum, welcher durch die ersten k Eigenfaces aufgespannt wird, wobei k der von Ihnen in (d) berechnete Index ist. Die Koeffizienten der Bilder bezüglich der Eigenfaces sollen in Form einer Matrix zurückgegeben werden, wobei die Koeffizienten eines Bildes eine Zeile der Matrix formen und die Zeilen die Koeffizienten für die Hauptkomponenten absteigend entsprechend der Magnitude der Singulärwerte enthalten.

f) **Identifizieren Sie Gesichter aus den Testdaten.**

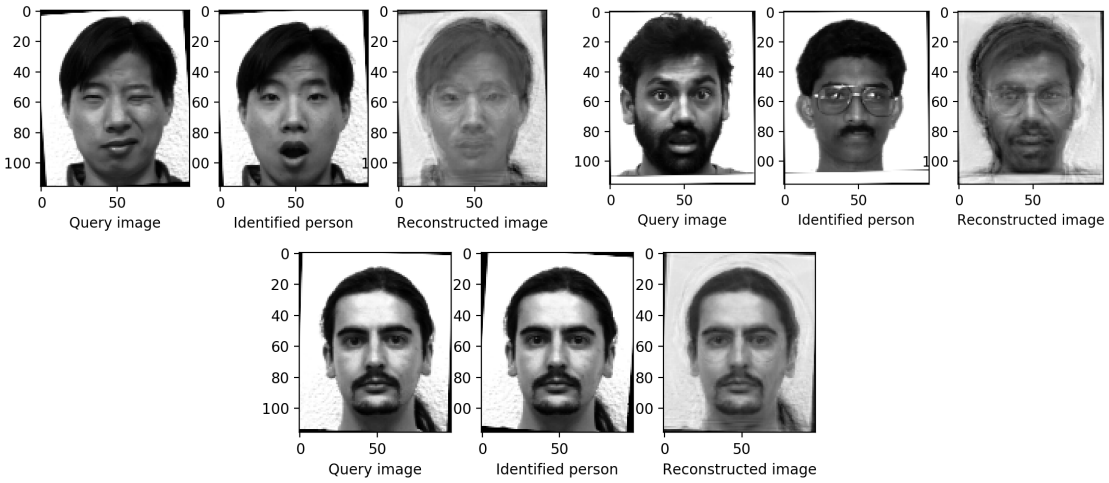

Abbildung 4: Teilergebnis der Aufgabe (f)

i) Laden Sie die Testdaten aus dem Verzeichnis './data/test'und projizieren Sie diese in den

Raum den die ersten k Eigenfaces aufspannen. Verwenden Sie hierfür die von Ihnen bereits implementierten Funktionen.

ii) Bestimmen Sie für jedes der Testbilder die Ähnlichkeit zu den Trainingsdaten. Als Ähnlichkeitsmaß soll der Winkel zwischen den Bildern im Eigenface-Raum dienen. Berechnen Sie dafür für jedes Paar von Trainings- und Testbildern den Winkelabstand. Stellen Sie nun mithilfe der Funktion *plot_identified_faces()*, welche in lib.py zur Verfügung gestellt wird, neben jedem Testbild das ähnlichste Trainingsbild sowie die Rekonstruktion des Bildes dar.

iii) Ohne Wertung: Wie groß ist Ihre Erfolgsquote? Vergleichen Sie die Erfolgsquote, wenn alle und wenn nur die ersten k Eigenfaces verwendet werden.

Abgabe Die Bearbeitungszeit der Teilaufgabe ist für ca. zwei Wochen ausgelegt. Die Abgabe soll via Moodle bis zu dem dort angegebenen Termin erfolgen. Verspätete Abgaben werden mit einem Abschlag von 3 Punkten je angefangener Woche Verspätung belegt. Geben Sie bitte jeweils nur eine einzige .zip-Datei mit den Quellen Ihrer Lösung ab.