인공지능. Project #1

● Features extraction

본 프로젝트에서는 feature extraction을 하는 두 가지 방법을 구현하고 비교한다. 데이터셋은 LFW(Labeled Faces in the World)를 사용한다. 이 데이터셋은 53명의 얼굴을 담은 이미지를 포함한다 이 데이터셋을 이용해 PCA를 이용한 feature extraction을 진행하고, 그렇게 추출한 feature를 이용해 같은 KNN 모델을 이용해 두 feature extraction에 따른 결과 차이를 비교해본다.

● PCA를 이용한 feature extraction

이 실습에서는 LFW(Labeled Faces in the World), 얼굴 데이터셋을 이용한다. 53명의 얼굴을 찍은 이미지를 담고 있다. 얼굴의 주성분들을 시각화 하여 확인해보고, PCA를 적용한 훈련 데이터로 학습시켜서 K-Nearest Neighbor(KNN) 알고리즘을 통해서 정확도를 측정해본다.

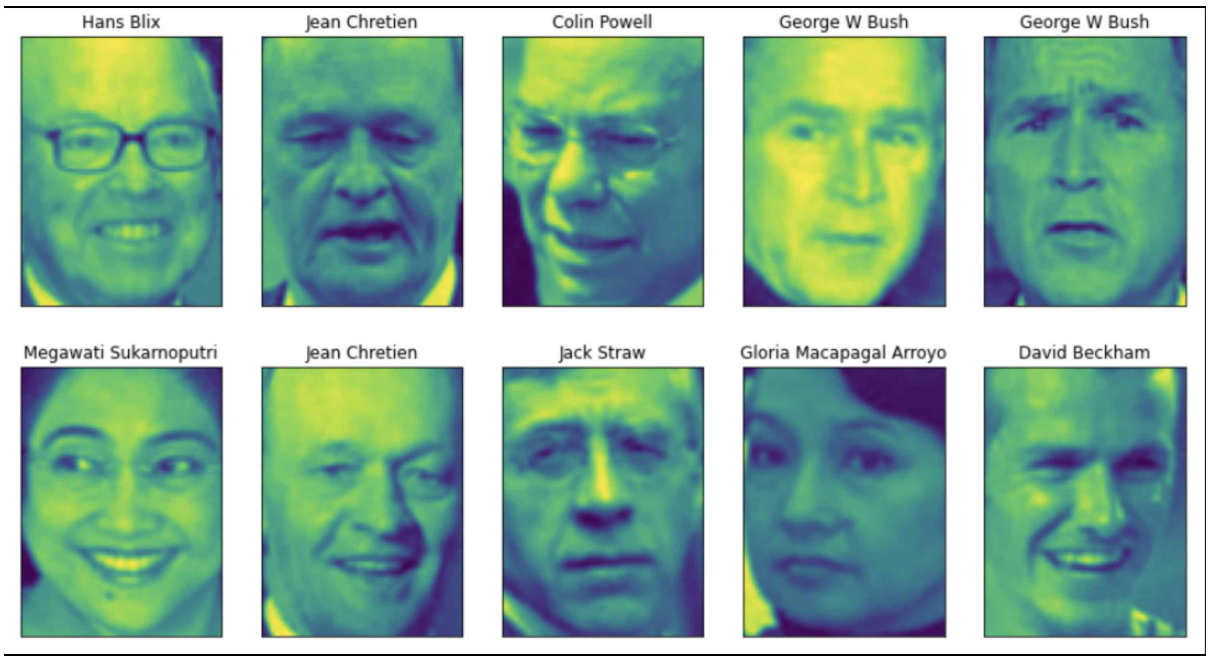

그림 1. 학습 얼굴 영상 집합

1. LFW(Labeled Faces in the World) Dataset

LFW Dataset을 그림 2처럼 불러오면 'people.keys()'로 데이터의 키들을 확인할 수 있다. 여기서 'images'는 위 그림 1에서 볼 수 있는 이미지 형태의 데이터를 확인할 수 있다. PCA를 이용해 데이터 차원을 줄이기 위해서는 이미지를 flatten하는 과정이 필요하다. 하지만 LFW 데이터는 이미 해당 과정을 거친 데이터가 존재한다. 해당 데이터에는 'people.data'처럼 입력해 접근할 수 있다. 'people.target'은 전체 데이터의 라벨에 접근할 수 있는 키이다.

```
people = fetch_lfw_people(min_faces_per_person=100, resize=0.5)
image_shape = people.images[0].shape
print("dataset keys        : ", people.keys())
print("dataset.images shape: ", people.images.shape)
print("dataset.data shape  : ", people.data.shape)
print("dataset.target shape: ", people.target.shape)
```

```
dataset keys        :  dict_keys(['data', 'images', 'target', 'target_names', 'DESCR'])
dataset.images shape:  (1140, 62, 47)
dataset.data shape  :  (1140, 2914)
dataset.target shape:  (1140,)
```

그림 2. Dataset load Example

```
Colin Powell: 236 images
Donald Rumsfeld: 121 images
George W Bush: 530 images
Gerhard Schroeder: 109 images
Tony Blair: 144 images
```

그림 3. 각각의 인물에 대한 이미지의 개수

데이터셋에서 편중된 데이터가 있음을 확인할 수 있다. 사용할 데이터 최소 크기는 100이상으로 하고 행렬 크기에 따라 연산 시간이 크게 증가하므로 resize크기를 0.5 이하로 한다.

2. Naïve PCA

2.1 QR 분해 구현

PCA는 고유값 분해를 기반으로 작동하는데, 이를 구현하는 수학적 알고리즘으로 QR분해가 있다. PCA에서 QR분해의 대상은 공분산행렬 즉, 정사각행렬이므로 계산이 상대적으로 용이하다. 만약 정사각행렬이 아니라면 SVD 등을 사용할 수 있다. QR분해를 컴퓨터에서 구현하기 위한 알고리즘으로는 대표적으로 gram-schmidt process가 있다. 이 프로젝트에서는 QR분해를 위해 gram-schmidt process를 구현하는 것을 목표로 한다.

2.2 Gram-schmidt process와 QR 분해

Gram-schmidt process는 행렬의 직교 기저들을 찾는 알고리즘으로, 보통 첫번째 열벡터를 해당 행렬의 기저 중 하나로 두고, 두번째 열벡터부터 이전 열벡터들에 정사영한 벡터를 그 크기만큼 빼는 과정을 반복한다. 수식으로 살펴보면 아래와 같다. 먼저 벡터 v의 u에 대한 projection을 아래와 같이 정의한다.

$$\mathrm{proj}_u(v) = \frac{<v, u>}{<u, u>} u$$

여기서 <v, u>는 내적을 의미한다. 이제 직교화하고 싶은 행렬 V와 그 벡터들을 v, 직교화된 행렬 U와 그 벡터들을 u라고 할 때 gram-schmidt process를 수식으로 펼쳐보면 아래와 같다.

$$u_1 = v_1$$

$$u_2 = v_2 - \mathrm{proj}_{u_1}(v_2)$$

$$u_3 = v_3 - \mathrm{proj}_{u_1}(v_3) - \mathrm{proj}_{u_2}(v_3)$$

$$u_4 = v_4 - \mathrm{proj}_{u_1}(v_4) - \mathrm{proj}_{u_2}(v_4) - \mathrm{proj}_{u_3}(v_4)$$

...

$$u_k = v_k - \sum_{j=1}^{k-1} \mathrm{proj}_{u_j}(v_k)$$

$$e_k = \frac{u_k}{\|u_k\|}$$

이제 각 u 벡터들을 크기로 나누어 정규화 하면 정규 직교 기저 벡터 e를 얻게 된다. 이제 공분산 행렬을 C, 그 벡터들을 c라고 할 때 아래 수식과 같이 QR분해를 수행할 수 있다.

$$[c_1\ c_2\ c_3] = [e_1\ e_2\ e_3]\begin{bmatrix} c_1^T e_1 & c_2^T e_1 & c_3^T e_1 \\ 0 & c_2^T e_2 & c_3^T e_2 \\ 0 & 0 & c_3^T e_3 \end{bmatrix}$$

2.3 고유값, 고유 벡터 추출

공분산 행렬을 R*Q 로 치환한 후 다시 같은 과정을 공분산 행렬이 대각 행렬에 수렴할 때까지 충분히 반복하면 대각 성분을 공분산 행렬의 고유값으로 해석할 수 있고, Q 행렬을 고유벡터로 해석할 수 있다. 이로써 최종적으로 고유값과 고유 벡터를 구할 수 있다.

2.4 상위 n개의 파라미터 추출

PCA는 고유값의 크기를 해당 파라미터의 중요도로 해석하는 것을 골자로 한다. 지금까지 고유값과 고유 벡터를 구했으니, 고유값을 크기 순으로 나열하고, 해당하는 고유벡터를 따로 얻어 원본 데이터를 선형 변환시킬 행렬을 얻는다. 해당 선형 변환이 완료된 데이터를 출력하면 PCA를 이용한 Feature extraction이 완료된다.

3. PCA with Whitening

픽셀을 비교할 때 얼굴 위치가 한 픽셀만 오른쪽으로 이동해 큰 차이를 만들어 다른 얼굴로 인식하게 된다. 기본적인 PCA 과정은 같지만, 해당 변환을 통해 정확도를 높일 수 있다. Whitening은 PCA를 수행하기 전 데이터 자체를 표준편차로 나누어 적용한다. 또한, 데이터의 평균을 빼서 평균을 0으로 만들어 주는 과정을 포함한다.

$$\mathrm{Whitening}(x) = \frac{x - \mathrm{mean}}{\mathrm{std}}$$

● KNN를 이용한 분류

충분히 잘 설계된 CNN 모델은 그 자체로 분류기 역할을 수행할 수 있기 때문에 별도의 KNN을 이용한 분류가 필요 없을 수 있다. 그런 경우 CNN은 해당 모델로만 결과를 출력하고, KNN을 이용한 분류

```
from sklearn.neighbors import KNeighborsClassifier

from sklearn.model_selection import train_test_split


import numpy as np
```

```
# 전처리한 데이터를 분할

x_train, x_test, y_train, y_test = train_test_split(x_people_scaled, y_people, # 분할할 데이터
                    stratify=y_people, random_state=0) # 그룹화할 데이터, 랜덤상태
```

는 PCA를 거친 데이터에만 적용하고, 결과를 출력해도 무방하다.

그림 4. Dataset split & KNN 라이브러리 로드 예시

```
# 머신 러닝 라이브러리 import

knn = KNeighborsClassifier(n_neighbors=1) # 이웃의 수

knn.fit(x_train, y_train) # 모델 학습
```

그림 5. KNN 예시

● F1 score

F1 score는 분류모델의 평가지표 중 하나이다. 수식은 아래와 같다.

$$\text{F1 score} = 2 * \frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

Recall과 precision은 각각 아래의 수식으로 정의되고, 수식을 구성하는 단어들은 아래 표와 같다.

$$\text{precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

		실제 정답	
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

그림 6. 분류 결과 정리 표

1. Naïve vs Whitening

F1 score를 기준으로 기본 PCA 데이터와 whitening이 적용된 PCA 데이터를 사용한 KNN 분류 결과를 출력한다

□채점 기준

파일	분류	점수	
코드 파일	PCA feature extraction	45	**75**
	Results	23	
	PCA improvements	7	
보고서	Introduction	2	**25**
	Algorithm	6	
	Results	9	
	Improvement	3	
	Consideration	5	

- **Improvement 관련 점수는 추가 점수 성격에 가까움 (총 10점)**
- **코드 파일의 improvements들은 보고서의 내용이 구현되었는지를 확인합니다**

□**주의사항**

✔ **코드 제한사항**

- Numpy 라이브러리의 numpy.linalg.eig, numpy.linalg.qr 등 과제 구현 사항을 **직접적으로 라이브러리를 사용하는 것**, 그 외 다른 라이브러리를 사용해 PCA, 고유값 분해, QR 분해 등을 라이브러리를 이용해 구현하는 것은 **해당 함수를 구현하지 않은 것으로 채점함**
- 사용 가능한 함수들: numpy.array, numpy.linalg.norm, numpy.dot, numpy.outer numpy.transpose, numpy.mean, numpy.sqrt, numpy.zeros, numpy.random.X (random 중 임의의 함수 X) 또는 다른 라이브러리의 예시와 같은 기본적인 함수들 사용 가능

✔ 제출기한

- 제출기한 : **~2024년 10월 09일 23:59:59** 까지 제출
- 추가제출기간 : **2024년 10월 10일 00:00:00 ~ 2024년 10월 16일 23:59:59**

※ 추가제출기간에는 채점 점수에 80% 점수 부여, 추가제출기간 이후에는 과제점수 0점

✓ 제출 방법

- 소스코드와 보고서 파일(pdf)을 함께 압축하여 제출
- 소스코드는 제공된 ipynb 파일을 기본으로 작성
- 모든 그래프, 성능 표 등의 결과물은 ipynb 파일 상에서 한 번 이상 실행하고, **결과물이 파일에 남아있는 상태로 제출**
- KLAS -> 과제 제출 -> 압축 파일 제출

✔ 환경 설치

- 별도의 환경 설치 없이 colab을 사용해 코딩 가능 참조: https://colab.research.google.com/
- 또는 아나콘다 설치, 가상환경 생성, 주피터 노트북 커널 생성 절차를 거쳐 해당 파일 편집 가능

아나콘다 설치:

https://www.codeit.kr/tutorials/76/Anaconda-%EC%84%A4%EC%B9%98%ED%95%98%EA%B8%B0-Windows

가상환경 생성: https://sdc-james.gitbook.io/onebook/2./2.1./2.1.1./2-conda-virtual-environments

주피터 노트북 사용: https://velog.io/@cs2tree/%EC%A3%BC%ED%94%BC%ED%84%B0-%EB%85%B8%ED%8A%B8%EB%B6%81-%EC%83%88%EB%A1%9C%EC%9A%B4-%EC%BB%A4%EB%84%90-%EB%A7%8C%EB%93%A4%EA%B8%B0

코랩 런타임 끊김 해결법: https://sjkoding.tistory.com/79

✔ 제출 형식

- 파일 이름 : 학번_AI_project1.zip

✔ 보고서 작성 형식

■ 보고서 내용은 한글로 작성

■ 보고서에는 소스코드를 포함하지 않음

■ 채점 기준 항목은 보고서 내용에 모두 포함되어야 함

■ 아래 각 항목을 모두 포함하여 작성

- Introduction : 프로젝트 내용에 대한 설명

  프로젝트 목표, 데이터셋 설명 포함

- Algorithm : 프로젝트에서 이용된 Algorithms 의 동작을 설명

  PCA, KNN 알고리즘 설명 포함

- Result : 결과를 그래프 또는 표로 첨부하고 동작을 설명

  KNN의 각 PCA 데이터에 대한 결과 포함

- Improvement : 개선한 부분이 있다면 추가로 작성하고, 결과 그래프 또는 표로 첨부 없다면 작성하지 않아도 무방함

- Consideration : 고찰 작성