Tema 2
Matei Rares-Andrei
315CC

Timp impletentare: 10 ore
Durata rulare checker pe masina personala ~ 27 sec

Rezolvare:

Task 1

Initializez variabila new_x si dau cast fotografiei la tipul de date double. Folosesc functia svd din octave pentru a obtine matricea diagonala cu valorile singulare (S) si matricile ortogonale U si V a caror coloane reprezinta vectorii singulari stanga, respectiv dreapta ai matricei photo. Apoi retin primele k coloane din U si V si retin submatricea formata din primele k linii si coloane din matricea S. Calculez apoi aproximarea imaginii folosind matricile obtinute si retin rezultatul in new_X, caruia ii dau cast la uint8.

Task 2

Initializez variabila new_x si dau cast fotografiei la tipul de date double. Parcurg matricea fotografiei linie cu linie si calculez media pentru fiecare folosind sum, urmand apoi sa scad din fiecare element al liniei media obtinuta. Construiesc matricea Z folosind formula data in cerinta si aplic svd acesteia. Retin apoi in matricea W primele pcs coloane din V, obtinand astfel spatiul componentelor principale. Calculez apoi matricea Y folosind formula $Y=W^T$ * photo, Y fiind proiectia pozei in spatiul obtinut anterior. In final, calculez aproximarea imaginii in variabila new_X folosind formula din cerinta si dau cast variabilei la uint8.

Task 3

Urmez pasii de la task 2 pana la calculul matricei de covarianta Z pe care o obtin cu formula $Z = photo * photo^T / (n - 1)$, unde n reprezinta numarul de coloane ale matricii imagine. Apoi calculez folosind functia eig vectorii proprii (retinuti in matricea v) si valorile proprii (retinute in matricea diagonala S). Folosesc vectorul s ce reprezinta diagonala lui S pentru a sorta descrescator valorile proprii si retin in vectorul poz pozitia din vectorul nesortat a fiecarei valoare proprie. Folosind poz sortez matricea v astfel incat prima coloana sa fie vectorul propriu corespunzator celei mai mari valori proprii si asa mai departe, rezultatul este pus in V. Apoi urmez pasii 5, 6, 7 de la task 2.

Task 4

Functia prepare_data primeste ca argumente un tabel si numarul de imagini pe care vrem sa le extragem si folosind functia load incarca datele intr-o variabila din care salvez in train_mat primele no_train_images imagini si in train_val etichetele acestora.

Functia visualise_image primeste ca argumente matricea cu imagini si un numar si afiseaza imaginea cu indicele number din matrice. Folosesc functia reshape pentru a transforma imaginea dintr-un vector de lungime 784 intr-o matrice de 28 x 28.

Functia magic_with_pca calculeaza matricea de covariatie si spatiul k-dimensional al componentelor principale identic ca la task-ul 3. Apoi calculeaza proiectia imaginii in spatiul obtinut si in final calculeaza aproximatia imaginii.

Functia prepare_photo inverseaza imaginea, fiecare pixel devenind diferenta in modul dintre valoarea acestuia si 255. Apoi matricea de dimensiune 28 x 28 este transformata intr-un vector linie de lungime 784.

Functia KNN calculeaza distanta euclidiana dintre fiecare linie a matricei Y (imaginile vectorizate) si vectorul de test. Apoi sortez crescator distantele obtinute si retin pozitiile din vectorul initial pentru fiecare valoare si extrag etichetele pentru primele k (= 5) imagini. Folosind functia median pe vectorul de etichete calculat obtin predictia.

Functia classify_image pune cap la cap functiile precedente din task 4. Apeleaza magic_with_pca pe setul de antrenamente, scade din imagine (sub forma de vector) media fiecarei coloane din train_mat (miu obtinut la magic_with_pca), schimba baza imaginii (spatiul obtinut la magic_with_pca) si aplica KNN pentru valorile obtinute (train_val = etichetele imaginilor, Y = proiectia imaginilor de antrenament in spatiul componentelor principale, test = imaginea in noua baza si k = 5 (numarul de imagini pe care le cautam)), obtinand in final predictia.