Sisteme de Recunoaştere a Formelor – Lab 7

Un clasificator nonparametric pentru recunoaşterea feţelor umane folosind k-Nearest Neigbours

1. Obiective

Obiectivul acestui laborator este demonstrarea construirii unui clasificator k nearest neighbor classifier (k-NN) pentru a face discrimina între imagini care conţin figuri umane şi cele care nu conţin.

2. Fundamente teoretice

Un clasificator este un algoritm care primeşte o mulţime de trăsături ca intrare, şi produce o etichetă a unei clase ca ieşire. Clasificatorii sunt construiţi luând o mulţime de exemple etichetate şi folosindu-le pentru a produce o regulă care va ataşa o etichetă la un exemplu nou. În cazul general, avem un set de date ce conţine exemple, $D = [X_1, X_2, \ldots, X_p]^T$. Fiecare exemplu din set este caraczerizat de o mulţime de trăsături, $X_k = (x_1, x_2, \ldots, x_d)$ şi fiecare exemplu are eticheta unei clase, y_i ce descrie tipul obiectului. Cunoaştem costul relativ al unei clasificări incorecte, şi trebuie să generăm o regulă ce poate ataşa o clasă oricărui exemplu plauzibil *x*.

În această sesiune de laborator vom avea două clase de imagini: feţe (cu eticheta 1) şi non-feţe (eticheta 0).

2.1 Algoritmul k-NN

Metodele vecinului cel mai apropiat clasifică un exemplu necunoscut prin folosirea clasei exemplului cel mai apropiat cu clasă cunoscută din mulţimea de antrenare, sau folosirea mai multor exemple din mulţimea de antrenare şi combinarea rezultatelor prin vot.

Un clasificator de tip vecinul cel mai apropiat *(k, l)* găseşte *k* exemple apropiate de cel pe care vrem să îl clasificăm, şi clasifică acest exemplu cu clasa care are cel mai mare număr de voturi, atâta timp cât această clasă are mai mult de *l* voturi (altfel exemplul se clasifică *necunoscut*).

Un clasificator *(k, 0)* – este de obicei cunoscut ca un clasificator **k-nearest neighbor**, iar un clasificator *(1, 0)*- ca clasificator **nearest neighbor**.

Algoritmul pentru clasificatorul *(k,0)*-este:

- Dându-se o imagine de clasificat, cu un vector de trăsături $X=(x_1, x_2, \ldots, x_d)$

1. Se determină *k* exemple din mulțimea de antrenare care sunt cele mai apropiate, $X_1, \ldots, X_k$; (vezi sectiunea 2.2).
2. Se determină clasa *c* (care poate fi față sau nu) care are cel mai mare număr de reprezentanți *q* în această mulțime; se clasifică *X* ca *c*.

2.2 Metrici de distanță

Pentru a determina cele mai apropiate exemple față de o imagine necunoscută trebuie folosită o metrică de distanță între vectorii de trăsături. O clasă generală de metrici pentru modele d-dimensionale este **metrica Minkowski**, cunoscută şi ca **norma L_k**:

$$L_k(a,b) = \left(\sum_{i=1}^{d} | a_i - b_i |^k \right)^{1/k}$$

Distanța Euclidiană este norma $\mathbf{L_2}$. Norma $\mathbf{L_1}$ este numită şi distanța **Manhattan** sau **city block**.

3. Activitate practică

În această sesiune de laborator se va implementa un detector de fețe simplu. Veți avea 160 imagini cu fețe şi 160 imagini fără fețe, pentru antrenare. Imaginile de antrenare sunt stocate în directorul *lab08_train_imgs*. Se va pune la dispoziție directorul *lab08_test_imgs* care conține pentru testare 8 imagini cu fețe şi 5 imagini fără. Fiecare imagine are dimensiunea 20 pe 20, şi va fi considerată ca un vector de trăsături de dimensiune 20*20=400.

Exemple tip față

Exemple non-față.

Cerințe:

1. Scrieți o funcție **ClassifyKnnManhatan** care clasifică o imagine de test (încărcată în lpSrc) folosind algoritmul k-NN şi distanța Manhattan. Folosiți imaginile de antrenare oferite. Funcția va citi valoarea lui *k* dintr-o căsuță de dialog.

2. Scrieți o funcție **ClassifyKnnEuclidean** ce clasifică o imagine de test (încărcată în) folosind algoritmul k-NN şi distanța Euclidiană.

3. Rulați cele două funcții implementate la (1) şi (2) pentru valori diferite ale lui k şi analizați diferențele de performanță pe măsură ce *k* creşte de la 20 la 100. Analizați diferența de performanță în funcție de metrica folosită.

Bibliografie:

[1] Richard O. Duda, Peter E. Hart and David G. Stork: *Pattern Classification 2nd ed.*

[2] D.A. Forsyth – *Computer Vision a Modern Approach*