Image search engine

Autor: Giuvelca Horia-Ioan, 331AA

Cuprins:

1. Introducere, prezentare a temei, prezentare a obietivelor

2. Prezentare pe scurt a suportului tehnic

3. Prezentare tehnica a etapei de realizare/implementare

4. Prezentare mod de utilizare, interactiune cu utilizatorul

5. Concluzii in care se vor urmari aspecte legate de: obiective, utilitate, performanta

6. Referinte bibliografice

1.Introducere

Toata lumea stie ce este un “text-based search engine” si toata lumea le foloseste. Exista multe exemple, printre care Google Chrome, Bing, Mozilla etc. Modul de functionare este simplu: introducm cateva cuvinte cheie legate de lucrul pe care il dorim, iar rezultatul ne este returnat. In acest fel este usor de intuit ce inseamna un “image search engine”: un motor de cautare care nu este bazat pe cuvinte cheie, ci pe o imagine. In teorie este destul de clar, dar cum functioneaza de fapt un astfel de motor de cautare?

O prima abordare a acestei probleme este sortarea dupa meta-date. Meta-datele unei poze se refera la data, ora si locatia unde poza a fost facuta, tag-urile pozei, care sunt adaugate manual de persoana care face/uploadeaza poza. Toate aceste meta-date sunt de fapt tot informatii de tip text despre poza. Cautarea folosind meta-date rareori sau chiar niciodata nu tine cont de ce este reprezentat in poza, “continutul” propriu-zis.

Cand o persoana incearca sa faca o cautare dupa meta-date, de fapt face o cerere similara cu un sistem clasic pe baza de text, iar imaginile care au meta-date similare sunt returnate.

Un exemplu de aplicatie care foloseste o sortare a imaginilor dupa meta-date este Instagram. Cand un utilizator posteaza o fotografie, el poate adauga o locatie, o descriere, care poate contine “hashtag-uri” si poate da si tag altor persoane in fotografie. Astfel, Instagram face aceasta sortare pentru a afisa utilizatorului alte poze ale altor utilizatori care au folosit acelasi “hashtag”, alte poze ale persoanelor care au primit tag in poza originala si alte poze care au fost postate din aceeasi locatie. Astfel, Instagram considera ca acele poze ar putea fi relevante pentru utilizatorul care a postat poza, chiar daca imaginile propriu-zise nu au legatura una cu alta.

1.Captura de ecran facuta in aplicatia Instagram, pe baza unor descrieri legate de sport

O alta abordare, mai complexa si mai precisa, din punctul de vedere al relevantei pentru utilizator este ceea ce se numeste “Search by example”. In contrast cu cautarea dupa meta-date, cautarea dupa exemplu se bazeaza numai pe continutul imaginii, fara a avea nevoie de cuvinte cheie sau metadate. Imaginea este analizata, cuantificata si depozitata in asa fel incat alte imagini similare vor fi afisate in timpul unei cautari.

Motoarele de cautare dupa imagini care cuantifica continutul imaginilor se numesc, in termeni specializati **Content-Based Image Retrieval** (CBIR) systems. Un astfel de motor de cautare este TinEye. Pentru a utiliza TinEye, utilizatorul trebuie sa introduca o imagine, iar TinEye returneaza imaginile cu un grad ridicat de similaritate si chiar paginile web unde a fost folosita imaginea originala.

2.Imaginea introdusa in aplicatia TinEye

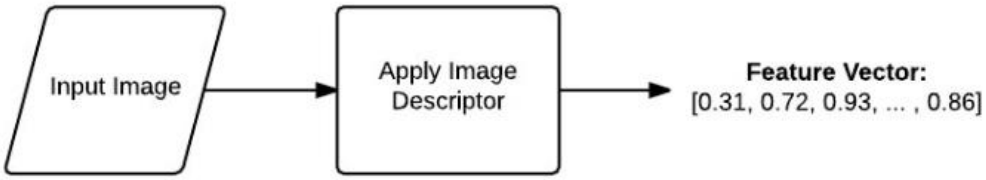
3.Imagini similare returnate de TinEye

Si atunci cum functioneaza o aplicatie ca TinEye? Exista o echipa de oameni care clasifica fiecare poza in parte? Ar fi un proces destul de indelungat si costisitor. In loc de acest lucru, se foloseste un fel de algoritm care extrage “proprietati” din imagine (un sir de numere care poate sa cuantifice si sa reprezinte imaginea in mod abstract). Apoi, cand un utilizator efectueaza cautarea pe baza unei imagini, proprietatile acelei imagini sunt comparate cu cele existente in baza de date si se returneaza imaginile similare.

2.Notiuni utile

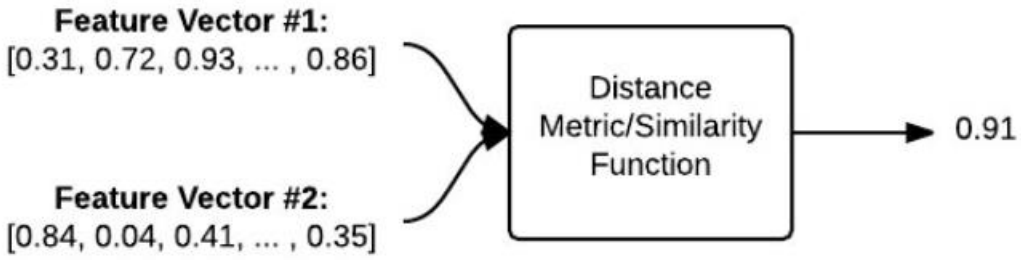
Primul lucru care ar trebui facut, ar fi o baza de date cu poze, cuantificate si clasificate, procedeu numit **indexare**. Pentru acest procedeu, va fi folosit un **descriptor de imagine** pentru a extrage **prorpietatile** fotografiei. Asadar, un descriptor de imagine defineste algoritmul utilizat pentru a descrie imaginea. De tinut minte este ca descriptorul de imagine defineste cum este cuantificata imaginea.

Pe de alta parte, proprietatile sunt rezultatul aplicarii descriptorului de imagine. Cand se aplica descriptorul de imagine asupra fotografiei, rezultatul va fi lista de proprietati. Cu alte cuvinte, proprietatile sunt doar numere folosite pentru a reprezenta intr-un mod abstract si pentru a cuantifica imaginea.



4.Aplicarea descriptorului de imagine asupra imaginii

Aceaste liste de proprietati pot fi apoi comparate intre ele folosindu-se o **functie de similaritate**. O astfel de functie accepta doua liste de proprietati (cu alte cuvine doua imagini) si returneaza un numar care reprezinta cat de similare sunt cele doua imagini.



5.Aplicarea functiei de similaritate

3.Etapa de realizare

Pentru implementarea aplicatiei, am urmat 4 pasi:

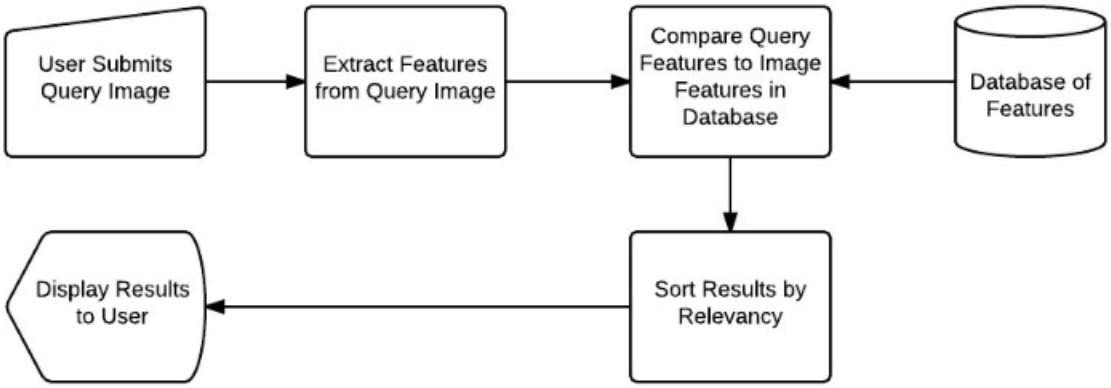
1) Definirea descriptorului de imagine: Am ales utilizarea unui simple, dar eficient descriptor de imagine - histograma de culoare.

2) Indexarea datelor: Dupa implementarea descriptorului de imagine, acesta trebuie aplicat asupra tuturor imaginilor din baza de date si trebuie extrase si stocate proprietatile acestora pentru a putea fi ulterior comparate pentru similaritati.

3) Definirea unui mod de a calcula similaritatile: Printre cele mai cunoscute tehnici se numara distanta Euclidiana, distanta cosinusoidala si “chi-squared distance”.

4) Cautarea propriu-zisa: Utilizatorul va efectua o cautare, introducand o imagine, iar aplicatia va returna toate imaginile similare cu cea introdusa.

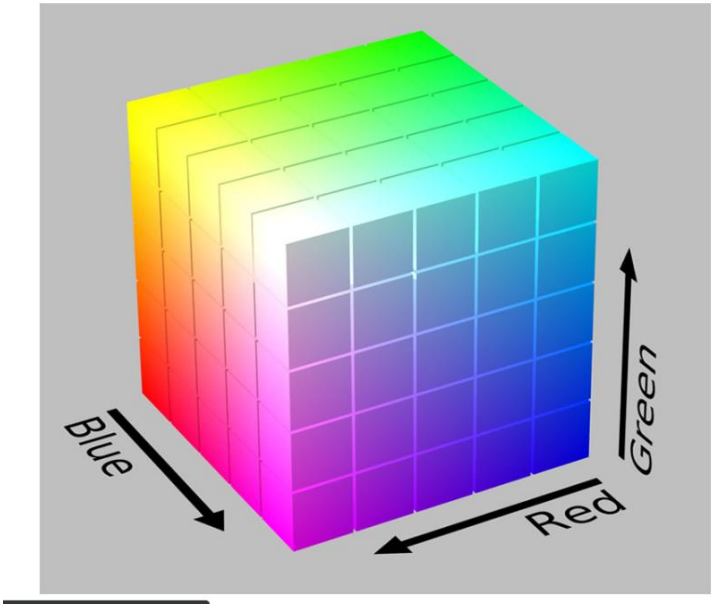
6.Vizualizare a pasilor 1 si 2



7.Vizualizare a pasilor 3 si 4

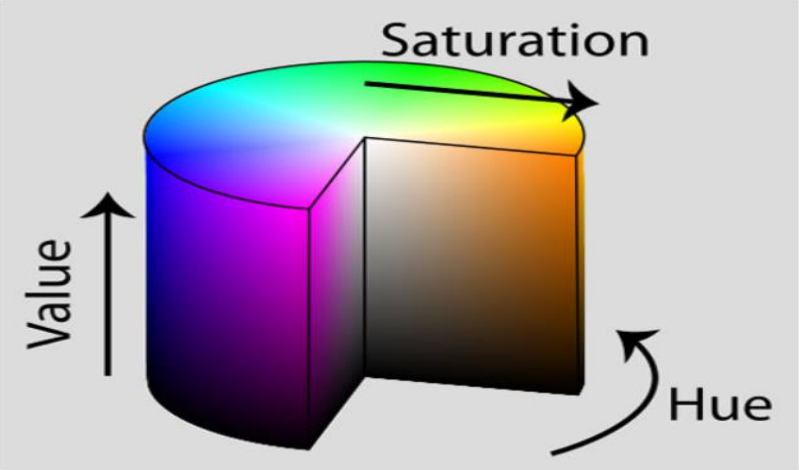
Pasul 1: definirea descriptorului de imagini

In mod normal, imaginile sunt reprezentate in spatiul de culoare “RGB”, cum se vede mai jos:



8.Cubul RGB

Cu toate ca spatiul de culoare RGB este simplu de inteles, acesta nu reuseste sa emuleze felul in care oamenii percep culorile. Astfel, voi folosi spatiul de culoare HSV, care mapeaza intensitatile pixelilor intr-un cilindru.

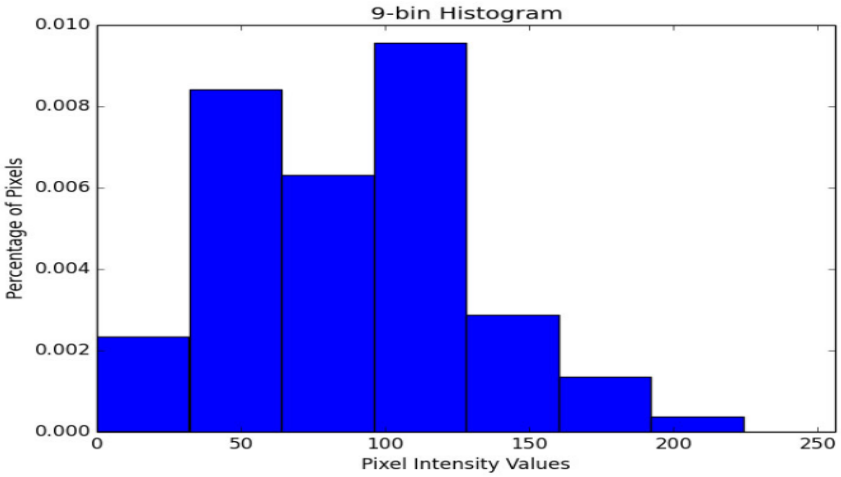


9.Cilindrul HSV

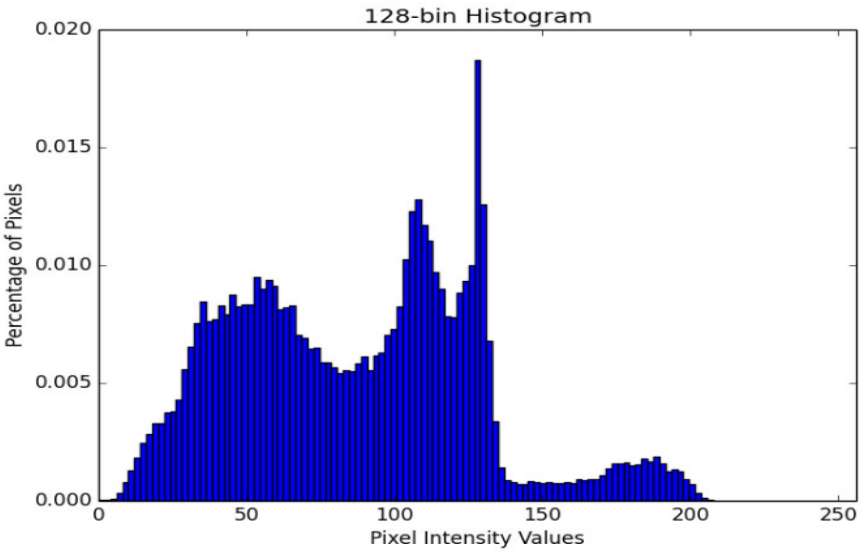
Exista si alte spatii de culoare care emuleaza si mai bine realitatea, cum ar fi spatiile CIE L*a*b si CIE XYZ, dar pentru aplicatia curenta, spatiul HSV este suficient.

Dupa alegerea spatiului de culoare, trebuie ales numarul de **intervale** pentru histograma. Histogramele arata o aproximare a densitatii intensitatii pixelilor dintr-o imagine. Deci, histograma ne va spune probabilitatea P ca o culoare C sa se afle in imaginea I.

Alegerea numarului de intervale este una importanta, pentru ca exista un compromis care trebuie facut. Daca sunt alese prea putine intervale, histograma va avea putine componente si nu va face diferenta atat de precis intre imagini care nu sunt atat de similare. Daca sunt alese prea multe intervale, histograma va avea multe componente si imaginile cu continuturi similare pot fi considerate prea diferite.



10.Histograma cu prea putine intervale



11.Histograma cu prea multe intervale

Deci, descriptorul de imagine HSV va stabili cati pixeli ai unei imagini au valoarea “Hue” care se incadreaza in intervalul 1 si cati pixeli au valoarea “Saturation” care se incadreaza in intervalul 2 si cati pixel au valoarea “Value” care se incadreaza in intervalul 3.

Pentru ca nu ar produce rezultate atat de precise, histograma pentru toata imaginea a fost inlocuita cu histograma pentru diferite regiuni ale imaginii. Folosind histograme regionale in locul celor globale, putem stabili densitatea unei culori intr-o anumita zona a imaginii.

12.Imaginea aleasa

In aceasta imagine se observa cerul albastru in partea de sus a imaginii si o plaja in partea de jos. Folosind o histograma globala, nu s-ar putea determina unde anume in imagine este cerul

albastru si unde este nisipul maro. In schimb, am sti doar ca exista un procent de albastru si un procent de maro. Pentru a remedia aceasta problema, imaginea este impartita in 5:

13.Imaginea impartita in 5 regiuni

In acest fel, putem avea o idee despre locatia si densitatea culorilor in imagine.

Pasul 2: indexarea

Dupa ce descriptorul de imagini a fost definit, acesta este aplicat fiecarei poze din colectie pentru a le indexa in functie de proprietati.

Pasul 3: Calculul similaritatilor

Pentru a calcula daca similaritatile dintre doua imagini, am folosit algoritmul “chi-squared distance”.

Pasul 4: Cautarea

Tot ce a ramas este legarea celor trei pasi de mai sus si se obtine rezultatul dorit.

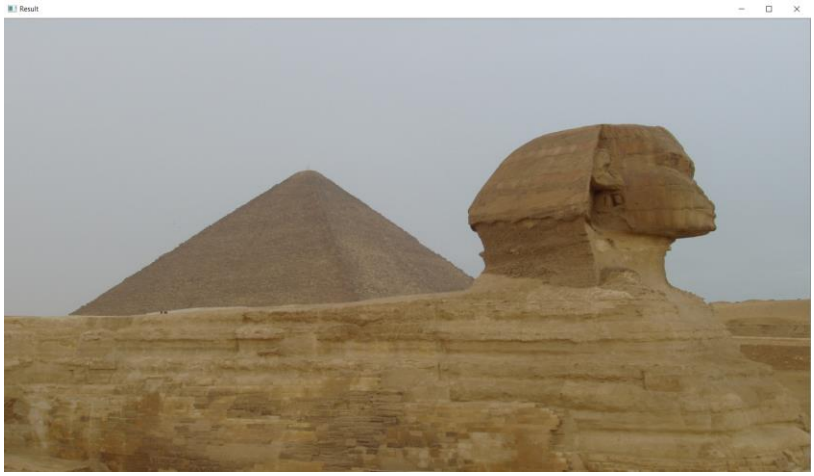
4.Utilizarea aplicatiei

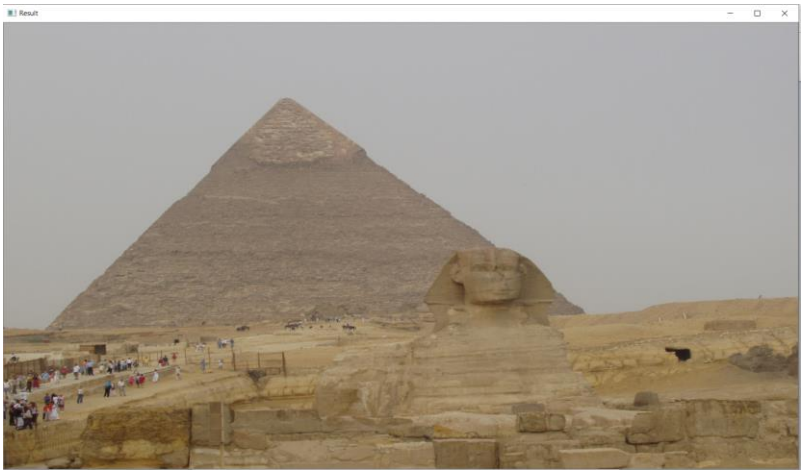
Mai intai, trebuie aleasa o colectie de poze pentru a crea baza de date cu proprietati. Pentru acest lucru, am folosit colectia gasita aici: http://lear.inrialpes.fr/people/jegou/data.php.

Apoi acestea trebuie indexate. Acest lucru se realizeaza dintr-un terminal cu ajutorul comenzii "**python index.py –dataset 'Path catre directorul cu poze' –index 'numele fisierului baza_de_date'**".

Pentru a efectua o cautare utilizatorul trebuie sa scrie intr-un terminal urmatoarea comanda: "**python search.py –index 'numele fisierului baza_de_date' –query 'Path-ul catre poza care se doreste a fi comparata' –result-path 'Path-ul catre directorul cu poze'**".

14.Exemplu de poza aleasa de utilizator





15.Imaginile returnate de aplicatie

5.Concluzii

Chiar daca aplicatia prezentata se bazeaza doar pe histograme de culoare, exista si alte moduri de a extrage proprietatile unei imagini, cum ar fi: texturile si formele. Compararea texturilor presupune observarea anumitor modele vizuale in imagini si aranjamentul lor in spatiu. Formele din imagini pot fi de multe ori determinate folosind algoritmi de segmentare sau “edge detection”. Descriptorii de forma folositi trebuie de asemenea sa tina cont si de rotatie, translatie sau scalare. O combinatie a acestor metode poate crea o aplicatie foarte precisa si puternica in ceea ce inseamna cautarea dupa imagini.

O astfel de aplicatie are multe utilizari: de la uzul personal, pana la domenii precum arhitectura, ingineria, medicina (pentru diverse diagnostice), industria textila, recunoastere faciala, armata, chiar si filtre de detectare a nuditatii etc.

6.Referinte bibliografice

[1] https://en.wikipedia.org/wiki/Content-based_image_retrieval#cite_note-22

[2] https://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/

[3] Chi-squared distance: https://www.hindawi.com/journals/mpe/2015/352849/

[4] https://docs.python.org/3/tutorial/

[5] https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html