METODE NUMERICE: Tema #1
Identificarea obiectelor în imagini

Termen de predare: 3 aprilie 2016

Titulari curs: *Florin Pop, George Popescu*

Responsabili: Darius Neațu, Denisa Sandu, Radu Stochițoiu, Alexandru Țifrea, Radu Vișan

Obiectivele Temei

În urma parcurgerii acestei teme studentul va fi capabil să:

- utilizeze în mod eficient funcționalitățile limbajului GNU Octave
- calculeze inverse de matrice sau să rezolve sisteme de ecuații lineare folosind GNU Octave și să eficientizeze calculul acestora.

Descrierea problemei

Dorel stă toată ziua pe Internet. De dimineața până seara stă și se uită la pozele pe care prietenii lui le încarcă pe rețelele de socializare. A observat un tipar care se repetă: multe din ele sunt imagini cu pisici. Dorel iubește pisicile așa că s-a decis să selecteze și să să salveze toate pozele cu pisici pe care prietenii săi le postează. Pentru aceasta are nevoie de un program care să identifice acele poze care conțin pisici. Evident, la fel cum se întâmplă de obicei în enunțurile temelor, voi trebuie să-l ajutați pe Dorel.

Aveți la dispoziție un set de imagini, dintre care o parte sunt cu pisici. Plecând de la acest set de date, va trebui să implementați un algoritm de clasificare simplu care să funcționeze pe imagini noi, care nu fac parte din data set-ul pe care îl aveți la dispoziție.

Task 1. Histogramă RGB

Imaginile pot fi reprezentate în spațiul RGB al culorilor, în care fiecare pixel este definit de un triplet (r, g, b), care reprezintă intensitatea roșului, a verdelui și, respectiv, a albastrului. Un mod prin care poate fi reprezentată distribuția culorilor într-o imagine este prin analizarea unei histograme, precum cea din Figura 1. O histogramă măsoară frecvența de apariție a pixelilor care au anumite valori pentru r, g sau b. Se observă că există câte o histogramă separată pentru fiecare dimensiune a spațiului culorilor (respectiv pentru roșu, verde și albastru).
Un model folosit frecvent este cel în care valorile lui r, g și b sunt întregi din intervalul $[0, 255]$. O histogramă pentru R, de exemplu, poate avea 256 de valori pe axa orizontală, caz în care vom obține, pe a i-a verticală, numărul de pixeli care au valoarea i a componentei R. Dacă pe axa orizontală există *count_bins* valori atunci pe a i-a verticală se va regăsi numărul de pixeli care are valoarea componentei R în intervalul $\left[i \cdot \frac{256}{count\_bins}, i \cdot \frac{256}{count\_bins} + \frac{256}{count\_bins}\right)$. În mod analog se procedează pentru componenetele G și B ale imaginii.

Cerință: Construiți cele trei histograme RGB pentru o imagine dată.

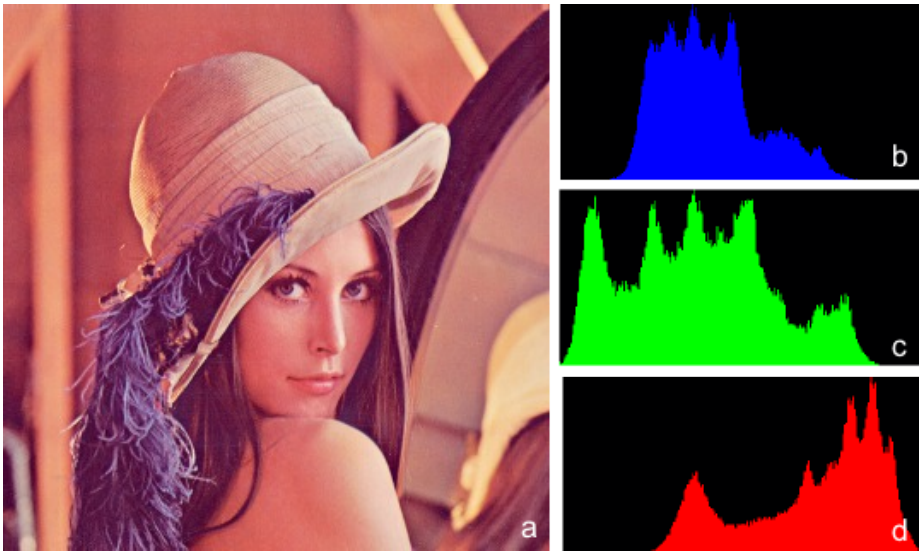

Figura 1. Histograma RGB a unei imagini.
(a) Imaginea 250x250 (b) Canalul albastru (c) Canalul verde (d) Canalul roșu.
Sursa: http://opencv-code.com/tutorials/drawing-histograms-in-opencv/

Hint: Puteți folosi funcția *imread* și orice valoare mai mare sau egală cu 16 pentru *count_bins*.

Input: Funcția voastră va primi ca parametru calea către o imagine și numărul de intervale de interes *count_bins*.

Output: Funcția voastră se va numi rgbHistogram și va returna o listă cu $3 \cdot$ *count_bins* elemente care va reprezenta concatenarea histogramelor imaginii.

Task 2. Histogramă HSV

Un spațiu al culorilor mai puțin familiar decât RGB, este HSV (Hue, Saturation, Value). La fel ca la RGB, fiecare pixel poate fi reprezentat ca un triplet (h, s, v) unde fiecare valoare din triplet îi corespunde unei dimensiuni din spațiul de reprezentare HSV. Pentru detalii, puteți consulta referințele de la finalul enunțului. Există metode prin care se poate face conversia între RGB și HSV. Cum imaginile pe care le aveți la dispoziție sunt reprezentate în formatul RGB, ne intereseaza conversia RGB2HSV. Puteți să folosiți algoritmul de mai jos pentru a realiza conversia unui singur pixel din spațiul RGB în spațiul HSV.

Țineți cont de faptul că, spre deosebire de RGB, unde fiecare componentă lua valori între 0 și 255, la HSV domeniile dimensiunilor sunt diferite: H ia valori între 0 și 360, în timp ce S și V iau valori de la 0 la 100. De aceea, pentru ușurință, e util să normăm valorile lui H, pentru a le aduce în intervalul $[0, 1]$ q(pasul 11 din algoritm).

Cerință: Construiți cele trei histograme HSV pentru o imagine dată.

Input: Funcția voastră va primi ca parametru calea către o imagine și numărul de intervale de interes *count_bins*.

Output: Funcția voastră se va numi hsvHistogram și va returna o listă cu $3 \cdot$ *count_bins* elemente care va reprezenta concatenarea histogramelor imaginii.

Algorithm 1 RGB2HSV

procedure RGB2HSV
  $R' \leftarrow R/255.$
  $G' \leftarrow G/255.$
  $B' \leftarrow B/255.$
  $Cmax \leftarrow \max(R', G', B').$
  $Cmin \leftarrow \min(R', G', B').$
  $\Delta \leftarrow Cmax - Cmin.$
  if $\Delta = 0$ **then** $H = 0$
  else
    if $Cmax = R'$ **then** $H \leftarrow 60° \cdot \left(\frac{G'-B'}{\Delta} \mod 6\right)$
    if $Cmax = G'$ **then** $H \leftarrow 60° \cdot \left(\frac{B'-R'}{\Delta} + 2\right)$
    if $Cmax = B'$ **then** $H \leftarrow 60° \cdot \left(\frac{R'-G'}{\Delta} + 4\right)$
  $H \leftarrow \frac{H}{360}.$
  if $Cmax = 0$ **then** $S = 0$
  else$S = \frac{\Delta}{Cmax}$
  $V \leftarrow Cmax.$

Task 3. Clasificare pisici

Pentru a rezolva problema de clasificare, ne vom folosi de histogramele (RGB sau HSV) imaginilor din setul de date. Vom folosi un model matematic pentru a surprinde distribuția datelor și pentru a determina suprafața care delimiteaza clasa pozelor cu pisici. Pentru a folosi acest model, e nevoie ca fiecare imagine sa fie reprezentată printr-un vector de caracteristici (*feature vector*). În cazul nostru, vom considera că acest vector este chiar rezultatul intors de funcțiile care construiesc histogramele. Astfel, vom avea $count\_bins \cdot 3$ caracteristici pentru fiecare imagine. Putem construi o matrice X care conține pe fiecare linie vectorul cu caracteristici al unei imagini. Astfel, avem $X \in \mathbb{R}^{N \cdot M}$, unde N este numărul total de imagini din setul de date și M este numărul de feature-uri ale unei imagini (i.e. $M = count\_bins \cdot 3$).

De asemenea, fiecare imagine va avea asociată o *etichetă*, care spune din ce clasă face parte aceasta. În cazul nostru este vorba doar de două clase: poze cu pisici sau poze fără pisici. Considerăm următoarea codificare: dacă imaginea cu numărul de ordine i este o poză cu pisici, atunci $t_i = 1$ și $t_i = -1$ altfel. Astfel, obținem un vector t cu N elemente care va conține label-urile imaginilor din setul de date.

Modelul pe care îl vom folosi este descris de un vector de parametri notat w, care are lungimea egală cu numărul de caracteristici ale fiecărui element din setul de date plus 1. Astfel lungimea lui w va fi egală cu $count\_bins \cdot 3 + 1$. Vectorul w descrie modelul imaginilor în raport cu caracteristicile pe care le-ați ales voi. Plecând de la setul de date pe care îl aveți la dispoziție (numit și set de învățare), trebuie să *învățați* valoarea vectorului w pe care îl veți folosi apoi ca să *ghiciți* clasa în care se află o nouă imagine. Pentru a putea face predicții bune, este necesar ca w să descrie bine nu doar imaginile din setul de învățare, ci trebuie să poată generaliza în așa fel încât împărțirea în clase pe care o modelează să rămână valabilă și pentru imagini noi, care nu au fost luate în considerare la învățarea lui w. După ce w este învățat, produsul scalar $y = w^T \cdot x$ (unde x este feature vector-ul unei imagini) va determina clasa în care se află imaginea dată: daca $y \geq 0$ atunci este o poză cu o pisică, altfel nu.

Pentru învățarea lui w trebuie rezolvat sistemul de ecuații $\widetilde{X}w = t$. $\widetilde{X}$ este matricea cu vectorii cu caracteristici X descrisă mai sus, la care *se adaugă la final o coloană cu 1-uri.* Dar $\widetilde{X}$ nu e neapărat să fie pătratică (de fapt, în practică, foarte des se întâmplă ca $N >> M$). De aceea, soluția acestui sistem va folosi *pseudoinversa Moore-Penrose*:
$w = \widetilde{X}^{\dagger}t = \left(\widetilde{X}^T\widetilde{X}\right)^{-1}\widetilde{X}^T t$, unde am notat cu $\widetilde{X}^{\dagger}$ pseudoinversa Moore-Penrose.

Cerință: Determinați vectorul de parametri w care modelează setul de date de învățare.

Input: Veți implementa două funcții. Funcția *preprocess* va primi ca parametri calea către un director care va conține setul de învățare, un string care selectează tipul histogramei care va fi folosită (i.e. "RGB" sau "HSV") și numărul de feature-uri care se doresc a fi extrase din fiecare imagine (i.e. *count_bins*). Directorul cu setul de date va avea următoarea structură:
/path/to/dataset/cats/
/path/to/dataset/not_cats/
Funcția *learn* va primi ca parametri matricea X și vectorul de etichete t.

Output: Funcția *preprocess* va returna matricea X și vectorul de etichete t, iar funcția *learn* va returna vectorul de parametri w.

Hint: Puteți să folosiți funcțiile de la task-urile anterioare pentru a construi vectorul de caracteristici pentru o imagine. Folosiți-vă de directorul în care se află o imagine (cats vs not_cats) pentru a construi vectorul de etichete t.

Task 4. Evaluare clasificator

Orice model obținut pentru un set de învățare trebuie să fie evaluat pentru date noi, care nu fac parte din setul dat. Pentru aceasta, veți determina predicția modelului vostru, $y = w^T \cdot x$, și în funcție de semnul lui y veți decide dacă imaginea conține sau nu pisici (v. explicațiile de la Task 3). Apoi veți compara predicția făcută folosindu-l pe y cu clasa reală a imaginii respective, care este cunoscută. Concret, aveți la dispoziție în directorul testset/ o serie de imagini de test care conțin pisici și care nu conțin pisici. De asemenea, va trebui să realizați o comparație între clasificarea care folosește histogramele RGB și cea care folosește histogramele HSV. Rezultatele pe care le observați vor trebui notate în README.

Cerință: Va trebui sa determinați care este raportul dintre numărul de imagini care sunt clasificate corect și numărul total de imagini pe care se evaluează modelul.

Input: Funcția voastră va primi ca parametri calea către directorul care conține setul de test, vectorul de parametri w învățat anterior, cât și un string care selectează tipul histogramei care va fi folosită (RGB vs HSV) și numărul de elemente din histogramă (adică *count_bins*).

Output: Funcția va avea numele *evaluate* și va returna procentul (i.e. un număr real între 0 și 100) imaginilor din setul de test care au fost clasificate corect.

Bonus (20 puncte)

Pentru distribuirea punctajului bonus se va realiza un clasament după timpul de execuție al parții de învățare (Task 3). Cele mai rapide 3 teme trimise vor primi bonusul integral. Tema aflată pe locul 4 va primi 15 din cele 20 de puncte, urmând ca temele aflate pe locurile următoare în clasament să primească fiecare cu 1 punct mai puțin decât tema de dinainte. Astfel locul 5 va primi 14 puncte, locul 6, 13 puncte și așa mai departe.

Detalii de implementare și redactare

Tema de casă va implementa funcțiile menționate la fiecare cerință în parte. Pentru implementarea temei puteţi folosi și alte funcţii definite de voi, dar cele menţionate mai sus sunt obligatorii. Trebuie să ţineţi cont de următoarele aspecte:

- codul sursă va conţine comentarii semnificative şi sugestive cu privire la implementarea algoritmilor;

- existenţa unui fişier `readme.txt` care va prezenta detaliile legate de implementarea şi testarea temei; de asemenea, la task-urile care cer explicarea anumitor aspecte, acestea trebuie să fie conținute în fisierul `readme.txt`
- fişierele care compun tema de casa vor fi incluse într-o arhivă `.zip` care respectă specificaţiile din regulamentul cursului;
- tema se va implementa în Octave şi va fi testată în mediul Octave.
- pentru o implementare corectă se vor putea obține maximum 90 de puncte și 10 puncte vor fi acordate pe README.
- pentru a obține punctajul pentru Task 3 va trebui ca modelul descris de parametrii w determinați de voi să aibă acuratețe mai mare decât 80% (acuratețea este măsura de evaluare descrisă la Task 4).

Frequently Asked Questions

Înainte de a posta o întrebare pe forum, parcurgeți secțiunea următoare și celelalte thread-uri de pe forum pentru a vă asigura că întrebarea voastră nu a fost pusă deja de altcineva:

Q: Avem voie să folosim `inv(A)` sau `A^(-1)`?
A: Categoric, nu. Scopul temei este ca voi să implementați metode de inversare a matricelor și nu să le folosiți pe cele existente în GNU Octave.

Q: Putem să folosim MatLab pentru a rezolva tema?
A: Da, cu mențiunea că MatLab pune la dispoziție mult mai multe funcții decât Octave. Cum checker-ul folosește Octave pentru verificare, va trebui să aveți grijă să folosiți doar funcții care sunt implementate și în Octave.

Resurse Web

1. https://en.wikipedia.org/wiki/Color_histogram
2. http://desktoppub.about.com/od/glossary/g/HSV.htm
3. https://en.wikipedia.org/wiki/HSL_and_HSV
4. https://en.wikipedia.org/wiki/Accuracy_and_precision
5. http://imgur.com/r/cats/top/all