UNIVERSITATEA BUCUREȘTI

FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ

Lucrare de licență

Pocket Library – scanner Android pentru recunoașterea cărților

Coordonator științific

Conf. Dr. Alexe Bogdan

Absolvent

Olariu Ciprian

București, iunie 2017

Abstract

In prezent, inteligenta și vederea artificiala joaca un rol important în viețile noastre, facilitând activități de zi cu zi. Modul în care aceste avansuri tehnologice au intrat în cotidian a fost unul subtil în multe cazuri, oamenii adaptându-se foarte ușor la noile tendințe.

Tema principala a acestei lucrări este propunerea unei soluții pentru oamenii aflați în căutare de cărți noi, ce frecventează librăriile și bibliotecile. Aceștia au nevoie de o sursa de informare în privința cărților pe care le vizualizează, care să fie mereu la îndemâna lor, cat mai ușor de folosit în astfel de situații.

Astfel, scopul prezentei lucrări este să propună un set de algoritmi pentru detectarea și identificarea automata a cărților pe baza fotografiilor făcute coperților lor. Mai mult de atât, lucrarea prezinta și o aplicație mobila dezvoltata în scopul demonstrării acestor tehnici.

Detectarea automata a unei coperți de carte a fost abordata utilizând un set de operații de preprocesare, precum eliminarea fundalului din imagine, detectarea muchiilor și a contururilor și ajustări de perspectiva. Rezultatele furnizate de acestea sunt în general bune, reușindu-se detectarea colturilor coperții din imagine.

Modalitățile propuse pentru identificarea cărții se bazează atât pe trăsăturile imaginii, cat și pe textul prezent pe coperta cărții. Caracteristicile unei poze au fost extrase cu ajutorul unor instrumente foarte populare în prezent în domeniul vederii artificiale, anume rețelele convoluționale. Mai multe astfel de rețele au fost comparate ca performanta în cadrul acestei lucrări și trăsăturile returnate de acestea au fost comparate cu cele existente într-o baza de date cu cărți preprocesate.

O a doua metoda de identificare a fost bazata de utilizarea unui motor de recunoaștere optica a caracterelor. Textul de coperta identificat astfel este mai apoi folosit într-o interogare online pentru căutarea informațiilor privind cartea în cauza.

Experimentele efectuate în timpul perioadei de cercetare au ajuns la concluzia ca soluția oferita de aceasta lucrare este un bun punct de plecare în rezolvarea problemei propuse, existând doar anumite neajunsuri și limitări tehnice ce pot fi optimizate în aplicația mobila în viitor. Pe măsura ce tehnologia din domeniul vederii artificiale va fi îmbunătățita și mai mult, aplicații de identificare automata a cărților vor putea fi dezvoltate cu ușurință.

Abstract

In the present, artificial intelligence and computer vision are playing an important role in our lives, making our daily activities easier. The way in which these technological improvements have become ordinary was very subtle in many cases, people adapting very easily to the new trends.

The main theme of this study is to propose a solution for those people searching new books, who are visiting libraries and bookshops often. Those people need an information source regarding books they are seeing, which should be always accessible to them and as easy to use as possible.

Therefore, the objective of this project is to propose a set of algorithms for the automatic book detection and recognition based on pictures of their covers. More than that, this study is also presenting a mobile application, developed with the purpose of demonstrating these techniques.

. The automatic detection of a book cover was approached using a set of preprocessing operations, such as background removal, edge and contour detection and perspective transformation. The results offered by these are good in general, the corners of the book cover being successfully detected in the picture.

The methods proposed for the book identification are based both on the picture's features and on the text present on the book cover. The image's features were extracted with the help of some very popular tools from the computer vision domain, namely convolutional neural networks. A number of such neural networks were compared looking at the performance in this current study and the features returned by them were matched with those existing in a preprocessed book database.

A second method of book recognition was based on using an optical character recognition engine. The text from the cover identified in this way is then used in an online query for searching the information related to the respective book.

The experiments made in the research period have reached the conclusion that the solution offered by this study is a good starting point in solving the proposed problem, while there are only some technological limitations to be optimised in the mobile application in the future. As the technology in the domain of computer vision will evolve even further, applications for automatic book recognition could be developed easily.

Cuprins

Lista de figuri

1. Introducere

1.1. Motivație

Problema care sta la baza acestei lucrări este una a iubitorilor de cărți. Mai exact este vorba despre persoanele care frecventează librarii și biblioteci în căutare de cărți noi. Se poate întâmpla des ca o persoana să vadă pe raftul unei librarii o carte despre care nu știe foarte multe și pe care ar intenționa să o cumpere, însă nu este sigura și ar vrea să afle mai multe informații online. De asemenea mai exista situația în care un student ar vedea la facultate o carte la un coleg sau la un profesor și ar dori cumva să își salveze acea carte în memorie pentru a se interesa de ea mai târziu acasă.

Pentru aceste situații exista posibilitatea de a te mulțumi cu informațiile de pe coperta, de a memora numele cărții pana acasă sau de căuta cartea pe internet direct de pe telefon tastând numele cărții și autorii. Însă fiecare din aceste metode poate avea unele deficiente în funcție de persoana sau de situație și de aceea noi metode care vin în ajutorul acestor persoane sunt binevenite.

1.2. Obiectiv

Aceasta lucrare își propune să prezinte o alta metoda de informare pentru cititorii de cărți aflați în situațiile descrise mai sus. Ca alternativa la memorarea unor nume de cărți și autori sau la tastarea pe telefon a acestora, prezenta lucrare propune folosirea camerei foto a telefoanelor pentru a face o poza coperții cărții în cauza.

Obiectivul este ca dintr-o imagine a unei coperți de carte, o serie de algoritmi de procesare și căutare să reușească să determine corect cartea pozata și sa-i ofere utilizatorului o serie de informații despre aceasta de pe internet în mod automat, scutindu-l de efortul căutării manuale.

1.3. Istoric

Exista și au existat un număr de experimente și aplicații care si-au propus un scop asemănător cu cel al recunoașterii cărților.

Un prim astfel de exemplu este proiectul SnapTell, un start-up pornit la ETH Zurich, care s-a focusat pe recunoașterea obiectelor din pozele capturate cu telefoanele mobile. SnapTell a lansat aplicații atât pentru dispozitivele cu sistem de operare Android, cat și pentru iPhone. Aplicația lor permitea utilizatorilor să fotografieze o coperta de CD, DVD, carte sau joc video pentru ca apoi să detecteze automat despre ce produs era vorba, utilizatorul primind informații despre el, cum ar fi

recenzii și comparații de preț pentru mai bine de 5 milioane de produse. Aplicațiile SnapTell au avut un foarte mare succes, start-up-ul fiind achiziționat ulterior de Amazon.

Un alt start-up, Recognize.im, de data aceasta polonez, a lansat în 2010 o tehnologie numita iTraff Technology pentru detecția de diverse produse comerciale în poze. Aceștia au pus la dispoziția tuturor dezvoltatorilor de aplicații mobile un API pentru recunoașterea produselor din comerț, cu scopul de a îmbunătăți experiența de cumpărături în supermarket-uri. Scopul lor a fost ca orice client să poată fotografia într-un magazin un produs cu telefonul, iar mai apoi, prin intermediul unei aplicații ce folosește API-ul lor, să poată primi informații relevante despre acesta.

Tot în 2010, gigantul american Google a lansat aplicația mobila Google Goggles, care a lansat oportunitatea de a căuta obiecte pe bine cunoscutul sau motor de căutare prin intermediul pozelor făcute cu telefoanele mobile. Prin aceasta aplicație, Google a oferit multe alte funcționalități în plus, precum detecția pe baza codurilor de bare sau detecția pe baza textului printat folosind un motor OCR. O versiune mult îmbunătățită a acestei aplicații la capitolele procesare de imagini și inteligenta artificiala a fost prezentata la conferința Google I/O 2017, sub denumirea de Google Lens.

După cum se poate concluziona, exista în acest domeniu un număr de aplicații de succes, dintre care unele nu se rezuma la detecția de cărți, ci oferă o gama mult mai larga de produse ce pot fi detectate. Fiind aplicații comerciale, multe dintre aceste exemple nu pun la dispoziția publicului o baza teoretica pentru munca lor. Astfel, aceasta lucrare nu își propune o competiție directa cu proiectele menționate anterior, ci găsirea și publicarea unor experimente ce conduc la o buna detecție a cărților folosind coperta lor.

1.4. Structura lucrării

Conținutul efectiv al prezentei lucrări este structurat în patru capitole ulterioare acestei introduceri.

Capitolul 2 își propune să introducă cititorilor acestei lucrări noțiunile teoretice folosite în implementarea aplicației. Se va porni de la noțiuni de bază de formare și procesare a imaginilor digitale și se vor parcurge în continuare inclusiv informații mai avansate despre inteligență artificială și vedere artificială.

Capitolul 3 se ocupă de descrierea aspectelor tehnice, mai exact se vor prezenta detalii despre îmbinarea noțiunilor teoretice descrise anterior într-o structură practică. În secțiunile acestui capitol se vor regăsi de asemenea și detalii despre tehnologiile auxiliare folosite la implementarea aplicației prototip, precum și o prezentare detaliată a interfeței acestei aplicații și a modului ei de utilizare.

Capitolul 4 este destinat detalierii experimentelor ce au fost conduse de-a lungul dezvoltării aplicației. Aceste experimente au vizat aspecte precum bază de date cu cărți folosită, seria de algoritmi utilizați, parametrizarea acestora, teste de acuratețe și teste de performanță.

Capitolul 5 încheie această lucrare cu concluzii privind soluția propusă, precum și privind utilitatea și performanță aplicației prototip dezvoltate.

1.5. Aplicație practică

Pentru a demonstra în mod practic conceptele enunțate în această lucrare, a fost realizată o aplicație mobilă pentru sistemul de operare Android, în care utilizatorul poate parcurge cu o imagine proprie toți pașii de procesare și detecție.

Deși aplicația implementează toată seria de algoritmi, ea nu își propune a fi o aplicație optimizată pentru a fi comercială, ci doar un prototip demonstrativ pentru conceptele teoretice enunțate în lucrare.

2. Fundamente teoretice

2.1. Procesarea de imagini digitale

O imagine poate fi definita ca o funcție bidimensionala $f: \mathbb{R}^2 \to \mathbb{R}$, unde $f(x, y)$ reprezinta intensitatea nivelului de gri la coordonatele (x, y) din imagine. Atunci când x, y și $f(x, y)$ sunt toate finite și reprezintă cantități discrete, spunem ca imaginea este una digitala. Spunem ca imaginea digitala este color daca avem $f: \mathbb{R}^2 \to \mathbb{R}^3$. O imagine digitala este așadar compusa dintr-un număr finit de elemente, fiecare cu o locație particulara. Aceste elemente se numesc pixeli.

Procesul de captura al unei imagini digitale, exemplificat în **Figura 2-1**, este alcătuit din doua procese operate asupra reflecției luminii de pe obiect pe senzorul aparatului foto. Aceste procese sunt eșantionarea, adică discretizarea spațiului în pixeli în funcție de rezoluția senzorului, respectiv cuantizarea, ce reprezintă discretizarea luminozității în mulțimea numerelor întregi din intervalul $[0, 255]$.

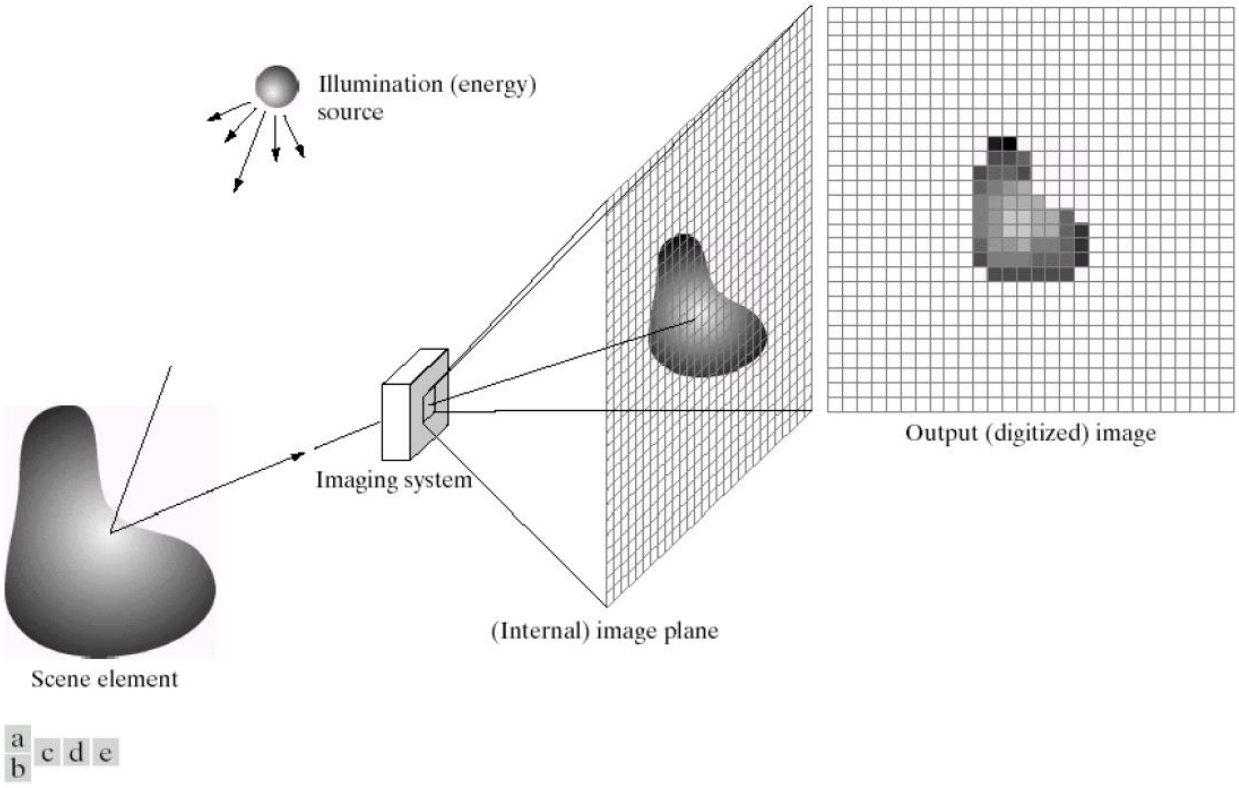


Figura 2-1 Procesul de captura al unei imagini digitale *– a) sursa de lumina; b) un element al scenei c) aparatul foto; d) proiecția luminii reflectate de pe obiect pe senzorul aparatului foto; e) imaginea digitalizata [1]*

Procesarea de imagini digitale se poate defini ca un set de operații de manipulare a pixelilor unei imagini cu ajutorul calculatorului.

2.2. Filtre și zgomot

Principalul exemplu de astfel de operație de manipulare este aplicarea de filtre. Un filtru, denumit și kernel sau masca, poate fi privit ca o matrice de dimensiuni impare ce conține ponderile asociate acelui filtru. Un exemplu este arătat în **Figura 2-2**.

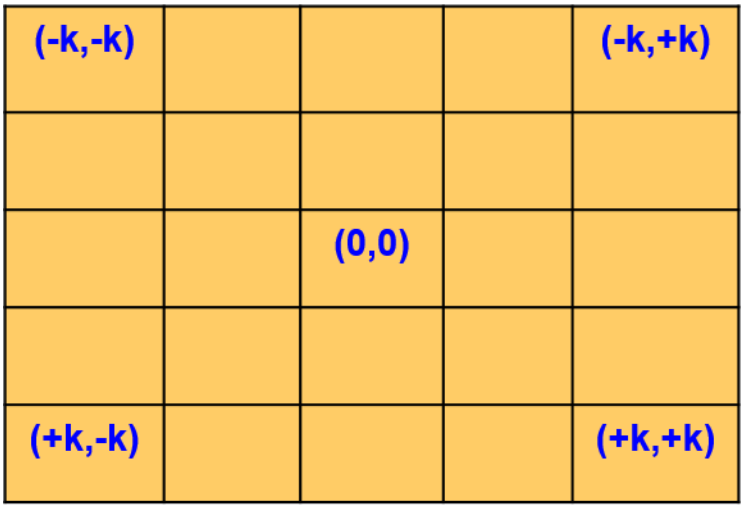


Figura 2-2 Exemplu de filtru – *Filtru F de dimensiuni* $(2k+1)*(2k+1)$ *centrat în* $(0,0)$

Aplicarea unui filtru se refera la înlocuirea intensității fiecărui pixel din imagine cu o combinație liniara a intensităților pixelilor din vecinătatea sa. Matematic, aceasta operație se numește convoluție și se definește ca mai jos:

$$O(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u,v) * I(i+u, j+v) \qquad (2.1)$$

unde:

- O este imaginea rezultata după aplicarea filtrului
- F este filtrul aplicat, de dimensiune $(2k+1)*(2k+1)$, centrat în $(0,0)$
- I este imaginea originala
- (i,j) sunt coordonatele pixelului din imagine care se procesează în acest moment

In cazul imaginilor color, operația de aplicare a unui filtru se face separat pe fiecare canal de culoare.

Un exemplu de filtru este cel de medie, în care ponderile fiecărui pixel din vecinătate sunt egale, cu suma 1. Rezultatul aplicării lui este așadar înlocuirea valorii unui pixel cu media aritmetica a valorilor pixelilor din vecinătatea lui.

Un bun exemplu de caz de utilizare a filtrelor este reducerea zgomotului din imagine. Zgomotul reprezintă o valoare cu care este alterata intensitatea unui anume pixel:

$$O(i,j) = I(i,j) + \eta(i,j) \qquad (2.2)$$

unde:

- O este imaginea obținuta
- I este imaginea ideala
- η este zgomutul

Acest defect este datorat în principal imperfecțiunilor tehnologice ale senzorilor cu care sunt captate imaginile. Exemple de zgomot sunt:

- Zgomotul normal, distribuit Gaussian de medie 0 și deviație standard σ
- Zgomotul „salt and pepper", în care unii pixeli au valoarea 0 sau 255
- Zgomotul de tip „impuls", în care unii pixeli au valoarea 255

Prin natura lor de a combina valorile intensităților pixelilor vecini în valoarea pixelului curent, filtrele reprezintă un mod bun de reducere a zgomotului, întrucât ne bazam pe prezumția ca acei pixeli afectați puternic de zgomot se afla la depărtare mare unul de celălalt în cadrul imaginii.

2.3. Gradienți și muchii în imagini

Într-o imagine, o muchie este locul în care se produce o schimbare bruscă a funcției de intensitate. Din punct de vedere matematic, muchiile corespund punctelor de extrem ale derivatei funcției de intensitate, fapt exemplificat în **Figura 2-3** pentru cazul unidimensional.

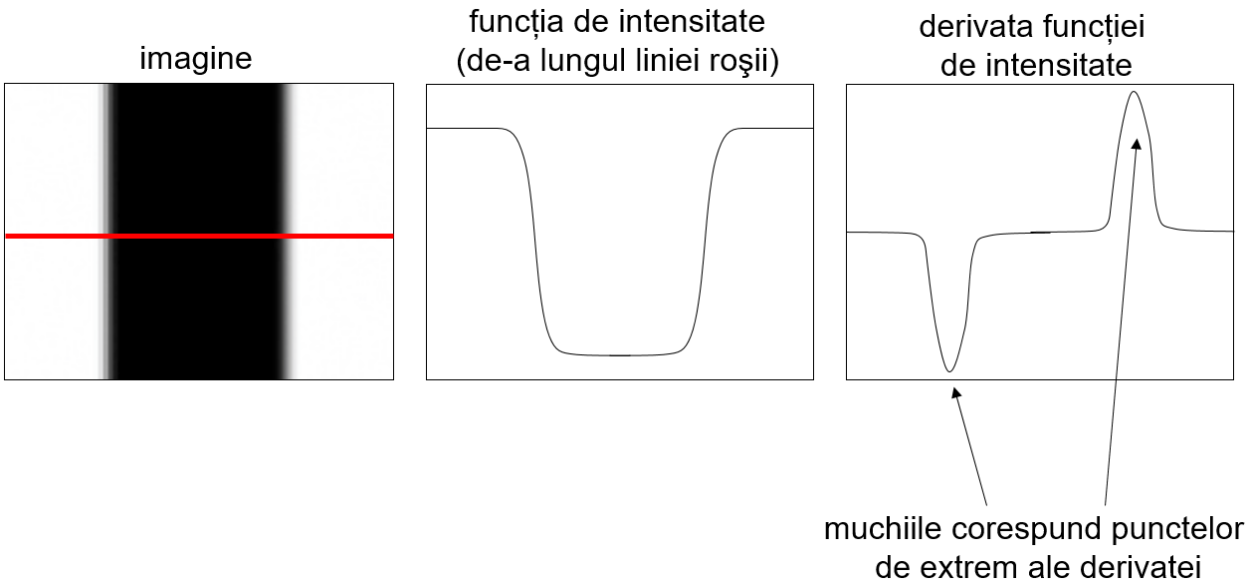


Figura 2-3 Exemplificare matematica a unei muchii în 1D

In cazul bidimensional al imaginilor, punctele de extrem ale derivatelor parțiale în raport cu x și cu y corespund muchiilor verticale, respectiv orizontale din imagine.

Pentru o funcție $f(x, y)$ derivata partiala în raport cu x este:

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon, y) - f(x,y)}{\varepsilon} \qquad (2.3)$$

O problema care apare la aceasta formula în cazul procesării imaginilor digitale este ca nu o putem calcula întocmai, fiind limitați de condițiile de discretizare ale imaginii. Astfel apare nevoia aproximării acestei derivate parțiale cu o formula de tipul următor:

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y)-f(x,y)}{1} \quad (2.4)$$

Aceasta aproximare a derivatei parțiale în raport cu x, respectiv cea corespunzatoare derivatei partiale în raport cu y se pot implementa ca operații de aplicare a filtrelor următoare:

$$F_{dx} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} F_{dy} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (2.5)$$

Cu toate acestea, exista în literatura de specialitate filtre ce aproximează mult mai bine derivatele parțiale pentru funcția de intensitate a unei imagini, cum ar fi filtrele de tip Prewitt, Sobel și Roberts.

De interes pentru aceasta lucrare sunt filtrele Sobel, utilizate în majoritatea algoritmilor de detecție a muchiilor dintr-o imagine. Acestea au fost introduse pentru prima data de către Irwin Sobel și Gary Feldman în prezentarea „Isotropic 3x3 Image Gradient Operator", susținută în cadrul Laboratorului de Inteligenta Artificiala de la Stanford în 1968 [2].

Filtrele Sobel calculează derivatele parțiale luând în considerare vecinătăți mai mari. Acestea sunt:

$$Sobel_{dx} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} Sobel_{dy} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (2.6)$$

Rezultatele aplicării acestor filtre se pot observa în **Figura 2-4**.

O alta noțiune matematica ce ne oferă informații în plus despre muchii este gradientul. Acesta se definește astfel:

$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} & \frac{\partial f(x,y)}{\partial y} \end{bmatrix} \quad (2.7)$$

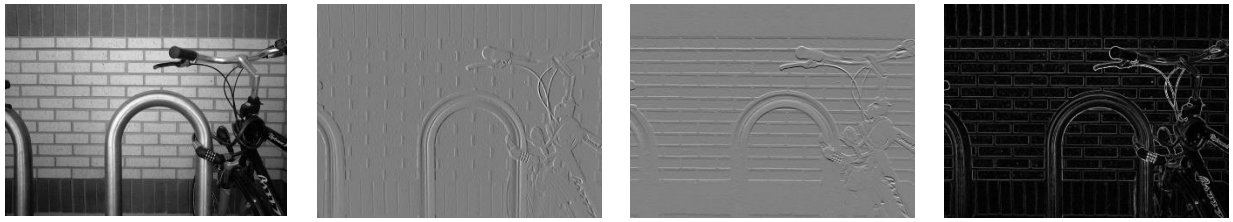
Gradientul arata direcția celei mai rapide schimbări de intensitate, întrucât direcția muchiei este perpendiculara pe direcția gradientului, ce se poate determina ca în formula de mai jos. Aceasta corelație intre direcția muchiei și a gradientului este exemplificata în **Figura 2-5**.

$$\theta = \tan^{-1}\left(\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}}\right) \quad (2.8)$$

De asemenea, o alta caracteristica a unei muchii este conferita de către magnitudinea gradientului, ce se poate calcula cu diferite formule precum:

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \; sau \; \|\nabla f\| = \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right| \qquad (2.9)$$

Așa cum se poate observa în **Figura 2-4**, afișarea magnitudinii gradientului unei imagini ne oferă o imagine sugestiva a muchiilor din imaginea originala.



a) b) c) d)

Figura 2-4 Rezultate ale operatorului Sobel *– a) imaginea originala; b) muchii verticale obținute cu filtrul* $Sobel_{dx}$*; c) muchii orizontale obținute cu filtrul* $Sobel_{dy}$*; d) muchii obținute prin afișarea magnitudinii gradientului*

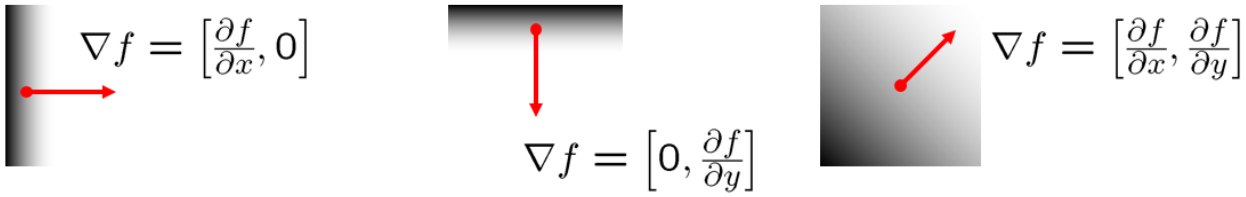


Figura 2-5 Direcția muchiei corelata cu direcția gradientului

2.4. Detectorul de muchii Canny

Acum ca avem o definiție clara a unei muchii, a intensității și orientării ei, putem prezenta și un algoritm de detecție a muchiilor dintr-o imagine.

Ce ne dorim să obținem cu un detector de muchii comparativ cu rezultatul obținut în **Figura 2-4** prin afișarea magnitudinii gradientului este o imagine alb-negru ce conține doar muchiile cele mai puternice din punct de vedere vizual, având o grosime cat mai mica în număr de pixeli.

Un foarte cunoscut astfel de algoritm, detectorul Canny, a fost publicat în 1986 de către John Canny în articolul „A Computational Approach to Edge Detection” [3].

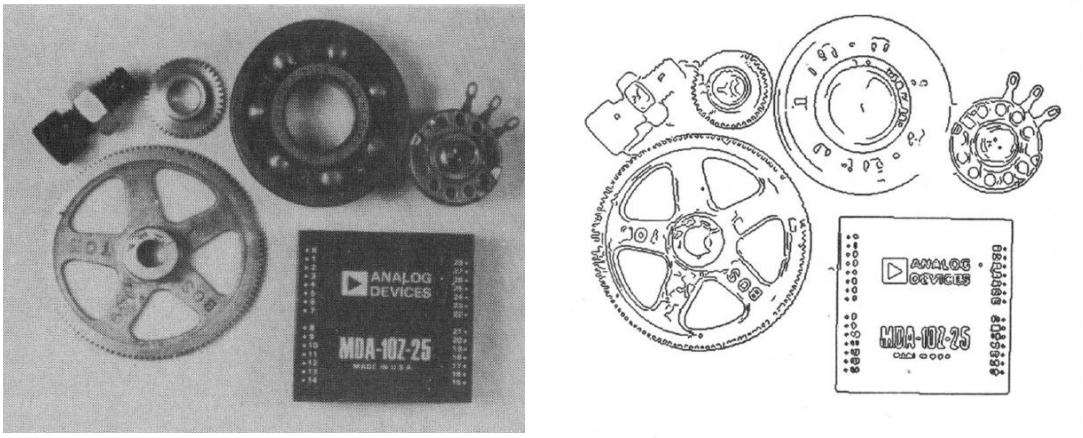
Acest algoritm de detectie a muchiilor se foloseste de operatorul Sobel definit anterior și are cinci pasi principali:

i. Se aplica un filtru Gaussian pentru netezirea imaginii, cu scopul eliminării zgomotului
ii. Se calculează gradientul pentru fiecare pixel împreună cu magnitudinea sa, utilizând filtrele Sobel

iii. Se suprima gradientul pentru acei pixeli ce nu au magnitudinea gradientului maxima comparativ cu pixelii vecini cu aceeași orientare a gradientului, cu scopul subțierii muchiilor

iv. Se stabilesc doua praguri $0 \leq L < H \leq 255$. Pixelii cu magnitudinea gradientului mai mica decat L se suprima, fiind sigur ca nu pot face parte dintr-o muchie suficient de vizibila. Cei cu magnitudinea mai mare decât H sunt marcați ca sigur făcând parte din muchii suficient de vizibile

v. Pixelii ramași nemarcați la pasul anterior, cu magnitudinea gradientului intre L și H, se păstrează doar daca sunt conectați de-a lungul unei muchii ce conține deja pixeli cu magnitudinea mai mare decât H



a) b)

Figura 2-6 Rezultatul detectorului de muchii Canny *– a) imaginea originala; b) muchiile detectate [3]*

2.5. Detectorul de contururi Suzuki

Deși prin detectorul de muchii Canny putem obține muchiile dintr-o imagine, ele nu sunt deloc corelate intre ele. Nu știm să spunem cate contururi distincte avem în imagine sau cărui contur ii aparține un anume pixel.

Aceasta problema se rezolva folosind un detector de contururi, care ne poate furniza o lista de contururi dintr-o imagine, precum și ierarhia lor.

Un exemplu faimos de astfel de detector este cel publicat în anul 1985 de către Satoshi Suzuki în articolul „Topological structural analysis of digitized binary images by border following” [4].

Algoritmul propus de Suzuki lucrează doar cu imagini binare, cum ar fi imaginea muchiilor rezultata după aplicarea detectorului de muchii Canny. Având în imagine doar pixeli albi sau negri, algoritmul efectuează o scanare de sus în jos a pixelilor. La întâlnirea unui nou pixel alb nemarcat se decide din ce fel de contur face parte, interior sau exterior, precum și în interiorul cărui contur deja determinat se afla, iar apoi se marchează toți pixelii albi conectați cu cel întâlnit.

2.6. Eliminarea fundalului unei imagini cu GrabCut

Un alt algoritm cheie pentru aceasta lucrare este eliminarea fundalului dintr-o imagine pentru separarea obiectelor de interes.

Un astfel de algoritm este GrabCut [5], dezvoltat într-un laborator de cercetare al Microsoft pentru produsele sale din suita Office. Acesta este un algoritm iterativ și interactiv, ce necesita un anumit număr de intervenții din partea utilizatorului.

Pașii principali ai acestui algoritm sunt:

- Utilizatorul specifica la început un dreptunghi din imagine în care se afla cu siguranță încadrat complet obiectul de interes
- Se marchează pixelii din afara dreptunghiului ca fiind cu siguranță fundal. Tot ce e în dreptunghi rămâne nemarcat, fiind incert daca e fundal sau obiect
- Un model de mixtura Gaussian este folosit pentru a modela distribuția pixelilor din zona marcată ca fiind fundal
- Urmând statistica de culoare rezultată pentru modelul de fundal, pixelii rămași nemarcați sunt etichetați ca fiind probabil fundal sau probabil obiect.
- Se construiește un graf cu costuri pe muchii în care pixelii imaginii sunt noduri, iar doua noduri adiționale sunt adăugate : nodul sursa și nodul destinație. Nodurile pixelilor sunt unite cu nodul sursa prin muchii cu costul egal cu probabilitatea pixelilor de a face parte din obiect. Analog se conectează pixelii la nodul destinație cu probabilitățile de a face parte din fundal. Se mai pun și muchii intre pixelii vecini, având un cost cu atât mai mic cu cat pixelii sunt mai diferiți ca și culoare
- Se executa un algoritm de tăietura minima în graful cu costuri, pentru a separa graful în doua subgrafuri, cel sursa și cel destinație. Nodurile ramase conectate cu sursa se marchează ca făcând parte din obiect, iar cele ramase conectate cu destinația se marchează ca făcând parte din fundal
- Algoritmul se repeta pana când clasificarea pixelilor converge sau pana când un număr maxim de iterații stabilit este atins

Pe lângă singura intervenție inițială a utilizatorului de a seta un dreptunghi de start pentru modelul statistic al fundalului, acesta mai poate interveni la final pentru a trasa linii de marcaj pe zonele din imagine care fac parte sigur din obiect sau din fundal, în cazul în care nu este mulțumit de rezultatul obținut, algoritmul reluând clasificarea pixelilor.

Un exemplu de demers al interacțiunii cu GrabCut este ilustrat în **Figura 2-7**.

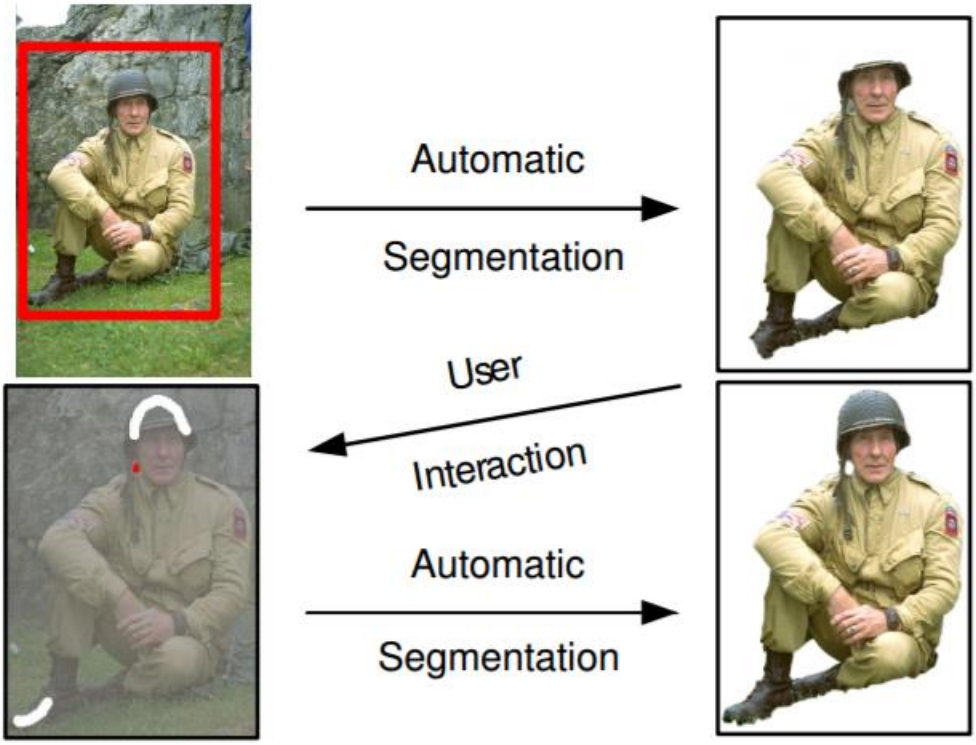


Figura 2-7 Exemplu de interacțiune cu GrabCut *– 1) utilizatorul marchează dreptunghiul ce conține soldatul; 2) GrabCut returnează soldatul aproape complet decupat; 3) Utilizatorul marchează zonele omise din obiect sau din fundal; 4) GrabCut returnează soldatul complet decupat [5]*

2.7. Rețele convoluționale

Pornind de la noțiunea de convoluție explicata într-o secțiune anterioara, un grup de cercetători condus de Yann LeCun a dezvoltat un nou concept în domeniul inteligentei artificiale și al învățării supervizate, sub denumirea de rețele convoluționale.

Conform articolului [6] publicat de Yann LeCun, ceea ce își dorește să obțină arhitectura unei astfel de rețele comparativ cu o rețea neuronala tradițională este invarianta la translatări, scalari și distorsiuni în procesarea automata a imaginilor.

Motivul pentru care în cazul acestor rețele imaginea de intrare este păstrată sub forma matriceala este acela ca pixelii au o anumita însemnătate într-o imagine când sunt păstrați împreună într-o vecinătate, creând un șablon. Daca am liniariza imaginea într-un vector, toate șabloanele din imagine s-ar pierde la procesarea din cadrul rețelei, ignorând-se astfel foarte multa informație.

In acest articol, autorul a propus și un prim model concret de rețea convolutionala, dezvoltata pentru recunoașterea automata a caracterelor ASCII, numit LeNet-5 [6], ilustrat în **Figura 2-8**.

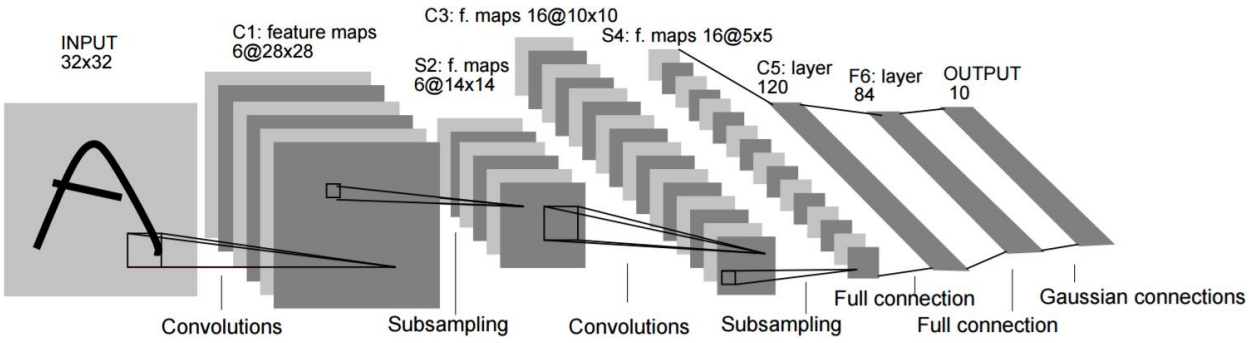


Figura 2-8 Arhitectura rețelei LeNet-5 *– o prima rețea convoluțională pentru recunoașterea caracterelor [6]*

Arhitectura generala a unei rețele convoluționale conține următoarele elemente:

- O imagine de intrare, sub forma unei matrici tridimensionale
- Straturi de convoluție, ce conțin unul sau mai multe filtre, fiecare filtru căutând în imaginea de intrare un anumit șablon. Fiecare filtru din stratul de convoluție se aplica pe imaginea de intrare în fiecare sub-matrice de dimensiunea sa, rezultând un set de imagini de ieșire de o dimensiune redusa, conținând atâtea imagini cate filtre au fost aplicate.
- După fiecare strat de convoluție urmează de obicei un strat cu o funcție de activare, care analizează rezultatele filtrelor aplicate anterior și decide cate din șabloanele căutate de acestea s-au descoperit cu adevărat în imaginea analizata.
- Întrucât după aplicarea multor filtre în straturile de convoluție rezulta matrici tridimensionale de imagini de ieșire foarte mari din punct de vedere al memoriei, este necesar să avem uneori și straturi de eșantionare. Aceste straturi selectează din numărul mare de rezultate doar acele șabloane relevante care au fost găsite pana în acel moment, reducând dimensiunea setului de imagini procesat.
- Spre finalul rețelei convoluționale vom avea câteva straturi de perceptori tradiționale, în care toți perceptorii vor fi conectați intre ei. La fel ca în rețelele neuronale tradiționale, aceste straturi au scopul de a procesa datele de intrare în scopul clasificării. Rezultatul obținut după aceste straturi este un vector de caracteristici.
- Vectorul de caracteristici obținut anterior este trecut printr-o funcție de activare ce va calcula scoruri, ce reprezintă probabilitatea ca imaginea de intrare să facă parte dintr-o anume clasa.

Deși s-a dovedit ca aceste rețele au rezultate cu mult mai bune fata de cele tradiționale, totuși ele au și un mare dezavantaj. Pentru antrenarea lor este necesar un volum mare de imagini, în scopul ajustării numărului foarte mare de ponderi din straturile componente.

In scopul ajutării comunității de cercetători din domeniul vederii artificiale, s-a născut proiectul ImageNet [7], cea mai mare baza de date cu imagini etichetate în vederea procesării automate, conținând în prezent peste 14 milioane de imagini adnotate conform standardului ierarhiei WordNet.

Odată cu crearea acestei baze de date, a fost inițiată și o competiție anuala [8] în care cele mai bune rețele neuronale concurează pentru clasificarea imaginilor în 1000 de categorii de obiecte. Acestora li se testează rata de acuratețe raportată la două rezultate: primul rezultat să fie cel corect sau acesta să se regăsească în primele 5.

Cu prilejul acestei competiție și cu ajutorul bazei de date ImageNet, rețelele convoluţionale au ajuns din nou în centrul atenției comunității de vedere artificiala.

Rețeaua convolutionala AlexNet a fost prezentata în 2012 cu un uriaș succes în cadrul competiției ILSVRC [8], sub denumirea de SuperVision, obținând o eroare top-5 de doar 16.4%. Deși în anii următori noi rețele și mai performante au fost dezvoltate, articolul de debut al acestei rețele [9] este și astăzi faimos, însumând peste 7000 de citări în literatura de specialitate.

Arhitectura acestei rețele, prezentata în **Figura 2-9**, urmează standardul propus în 1998 de LeCun [6], având 5 straturi de convoluție, intercalate cu straturi de activare și eșantionare, plus 3 straturi complet conectate la final ce reduc imaginea de intrare la un vector cu 1000 de scoruri corespunzătoare celor 1000 de clase de obiecte din cadrul competiției mai sus menționate. Având peste 650 de mii de noduri și peste 60 de milioane de ponderi reglabile, antrenarea acestei rețele a necesitat 5-6 zile de procesare pe doua placi video.

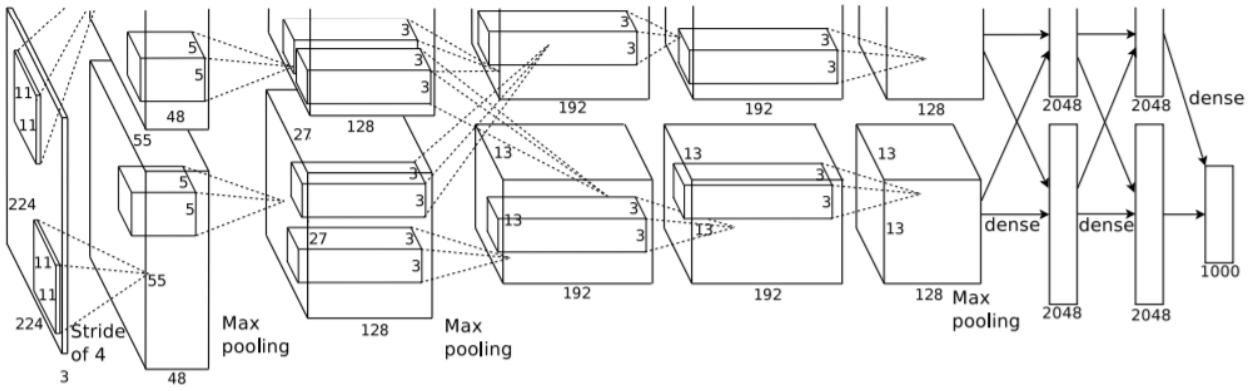


Figura 2-9 Arhitectura rețelei AlexNet *– prima rețea convolutionala de succes din cadrul competiției ILSVRC [9]*

In anii următori, rețelele participante la aceasta competiție au ajuns la un nivel de acuratețe în clasificarea imaginilor de 4.94% eroare, ce a depășit capabilitățile omului de 5.1% eroare [10].

3. Implementarea aplicației

3.1. Tehnologii folosite

Pentru a ilustra aplicabilitatea sistemului de algoritmi de procesare și detecție propus de aceasta lucrare cu obiectivul detecției automate de cărți, a fost creata o aplicație mobila ce va fi detaliata în acest capitol.

Aplicația mobila, intitulata „Pocket Library”, este dedicata rulării pe sistemul de operare Android. Motivul alegerii unei aplicații mobile în detrimentul unui script pe calculator este ușurință de a testa aplicația pretutindeni în cazuri de zi cu zi cu ajutorul camerei foto de pe orice telefon modern. Platforma Android a fost aleasa pentru dezvoltare în detrimentul altor sisteme de operare mobile, precum iOS, datorita procentajului mare de dispozitive ce rulează Android, de o diversitate mare ca număr de producători. In cele mai recente sondaje, Android rulează pe mai mult de 85% din telefoanele inteligente aflate în uz.

Pentru dezvoltarea aplicației, s-a utilizat suita nativa recomandata de limbaje de programare, având baza în clase Java și fișiere de interfață în format XML, mediul de programare și testare folosit fiind Android Studio.

De o importanta foarte mare în implementarea acestei aplicații a fost librăria open-source de vedere artificiala OpenCV [11]. Aceasta conține un număr mare de algoritmi de vedere artificiala și procesare de imagini și videoclipuri, optimizați din punct de vedere computațional. Un alt mare avantaj al librăriei este acela ca oferă interfațare cu limbaje de programare precum C++, Java și Python, suportând dezvoltarea pe multiple platforme și sisteme de operare.

O alta tehnologie cheie folosita în cadrul experimentelor și în aplicație este implementarea de rețele convoluționale furnizata de cercetătorii de la BAIR, laboratorul de inteligenta artificiala al Universității Berkeley, proiect lansat sub denumirea de Caffe [12]. Printre avantajele notabile ale lui Caffe se numără viteza de procesare de peste 60 de milioane de imagini pe zi utilizând o singura placa video, ușurință de configurare a rețelelor și comunitatea mare de dezvoltatori contributorii. Acesta oferă interfațare cu Python și MATLAB, însă comunitatea a dezvoltat și suport pentru alte platforme, precum Android.

In mod evident, dezvoltarea acestei aplicații ar fi fost imposibila fără o baza de date cu imagini pentru coperțile de cărți. Căutarea unei astfel de baze de date poate fi anevoioasa și tocmai de aceea cei de la Internet Archive si-au propus realizarea unui astfel de proiect, accesibil tuturor. Scopul proiectului intitulat OpenLibrary [13] este ca fiecare carte să aibă o pagina web cu cat mai multe informații despre ea. In decursul mai multor ani, aceștia au strâns informații despre peste 20 de milioane de cărți, aproape 7 milioane având și o imagine a coperții. Un avantaj foarte mare al acestei platforme publice este ca toate informațiile despre cărți sunt puse la dispoziția dezvoltatorilor printr-un API gratuit. O alta baza de date cu cărți ce oferă un API util acestei aplicații este cea oferita de Google în cadrul platformei Google Books [14].

In perioada de dezvoltare s-au efectuat și experimente în care s-a folosit un motor de recunoaștere optica a caracterelor pe imaginea coperții. Alegerea în acest domeniu s-a îndreptat către Tesseract [15], un motor OCR open-source popular, a cărui acuratețe este considerata ca fiind

printre cele mai bune de pe piață. Acesta a fost dezvoltat încă din 1985 de către HP, iar din 2006 suportul pentru dezvoltare a fost asigurat de către Google. La fel ca în cazul celor menționate anterior, comunitatea de dezvoltatori a ajutat la portarea lui pe diverse platforme și la interfațarea cu multe limbaje de programare.

3.2. Interfața și modul de utilizare

In aceasta secțiune se va prezenta modul de utilizare al aplicației. Interfața ei este una simplista, iar interacțiunea utilizatorului cu aplicația este una intuitiva, trecându-se algoritmic prin fiecare pas de procesare și detecție. In scop demonstrativ și educativ, rezultatul fiecărui pas de procesare este afișat pe ecran, spre deosebire de o aplicație comerciala unde ar fi fost afișat direct rezultatul final dorit.

Primul ecran din aplicație îl întâmpină pe utilizator cu meniul de opțiuni, prezentat în **Figura 3-1**. Acesta are opțiunea de a-si alege o poza din galeria de imagini, de a face pe loc o poza utilizând camera foto a telefonului sau de a merge într-un ecran informativ al aplicației.

Cum se poate observa în **Figura 3-2**, indiferent de sursa imaginii selectate, aceasta va fi vizualizata într-un ecran ulterior, ce reprezintă punctul de start în procesarea ei.

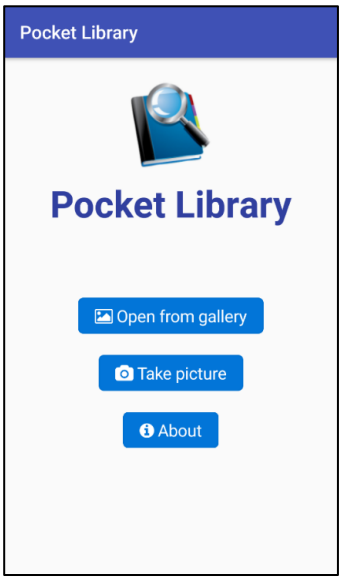


Figura 3-1 Meniul principal

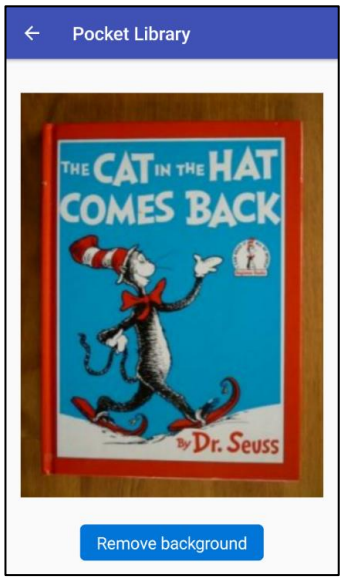


Figura 3-2 Afișarea imaginii selectate

Primul pas efectuat este eliminarea automata a fundalului, acțiune al cărei rezultat este vizualizat în ecranul următor, prezentat în **Figura 3-3**. Pe imaginea rezultata astfel se vor identifica contururile și se vor vizualiza într-o imagine alb-negru în ecranul următor, capturat în **Figura 3-4**.

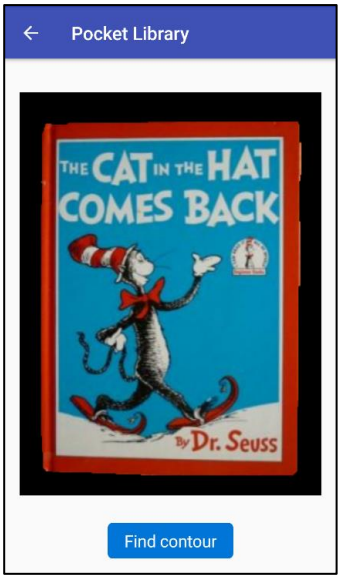


Figura 3-3 Eliminarea fundalului

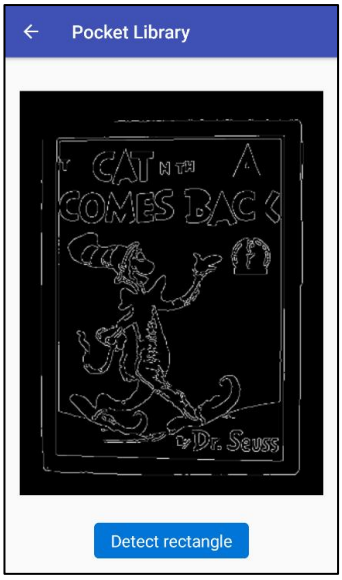


Figura 3-4 Vizualizarea contururilor

Următorul pas este detectarea automata a patrulaterului ce reprezintă coperta cărții în sine. Așa cum se poate observa în **Figura 3-5**, în funcție de cat de bine au fost identificate contururile din imagine, este posibil ca patrulaterul de încadrare găsit să nu fie cel dorit, iar imaginea decupata din acesta în ecranul următor, din **Figura 3-6**, poate fi incompleta.

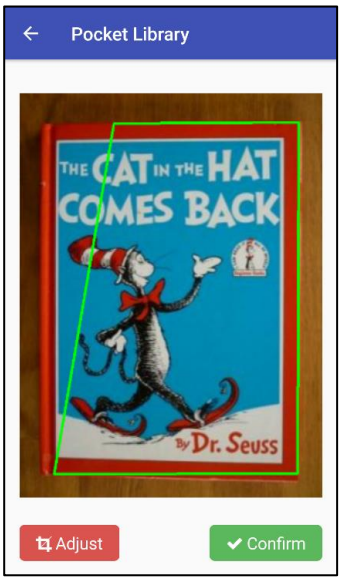


Figura 3-5 Detectarea coperții

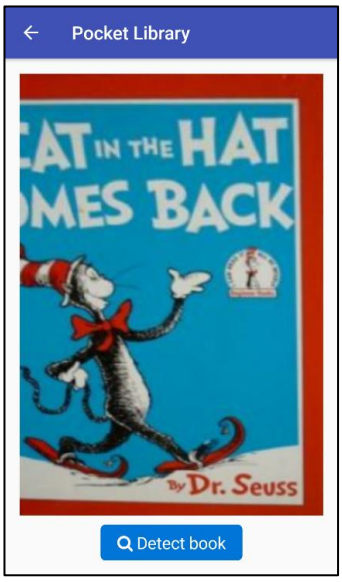


Figura 3-6 Decuparea coperții

Cu acest pas de decupare a coperții din imagine se finalizează seria de algoritmi de procesare a imaginii și urmează cei ce au ca scop identificarea cărții propriu-zise.

In acest sens, utilizatorul trebuie să opteze în ecranul următor pentru metoda de detecție dorită: pe baza caracteristicilor imaginii coperții sau pe baza textului din aceasta.

După selectarea opțiunii ca în **Figura 3-7**, un ecran de încărcare va face tranziția către lista rezultatelor obținute, exemplificata în **Figura 3-8**. Aceasta lista derulabila pe verticala conține cele

mai relevante rezultate pentru căutarea făcută, oferindu-i utilizatorului opțiunea de a identifica intre primele rezultate cartea căutata.

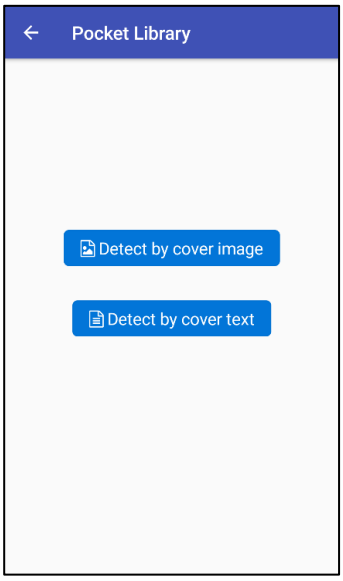


Figura 3-7 Opțiunile de detecție

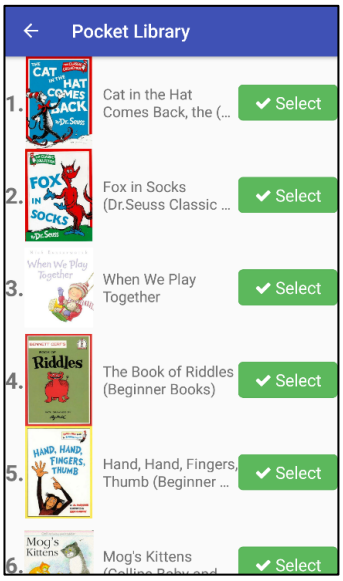


Figura 3-8 Lista rezultatelor

După cum se poate observa în **Figura 3-8**, în cazul căutării folosind imaginea coperții, cel mai relevant rezultat din lista este unul corect, deși coperta fusese doar parțial detectata.

La apăsarea butonului din dreptul cărții identificate drept rezultatul dorit, utilizatorul va ajunge la ultimul ecran al aplicației, cel care ii prezinta un set de informații relevante despre cartea sa.

Aceste informații conțin imaginea coperții, titlul, autorii, ISBN-ul, editura, data publicării, numărul de pagini, precum și adresa unei pagini web ce poate oferi mai multe detalii. Acest ultim ecran poate fi vizualizat mai jos, în **Figura 3-9**.

Figura 3-9 Informații despre carte

In **Figura 3-5**, unde observasem ca detectarea coperții a eșuat parțial, utilizatorul are opțiunea de a ajusta patrulaterul de încadrare înainte de a continua. Ajustarea se realizează într-un ecran separat, prezent în **Figura 3-10**, cu ajutorul a patru colturi ajustabile.

Utilizatorul se întoarce apoi la ecranul anterior, actualizat cu noul patrulater, ca în **Figura 3-11**. Astfel se obține o coperta decupata perfect, ilustrata în **Figura 3-12**.

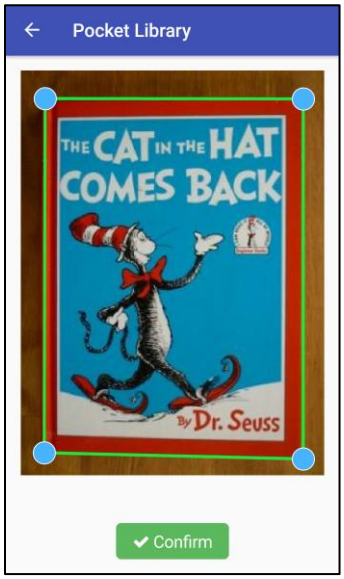


Figura 3-10 Ajustarea patrulaterului de încadrare

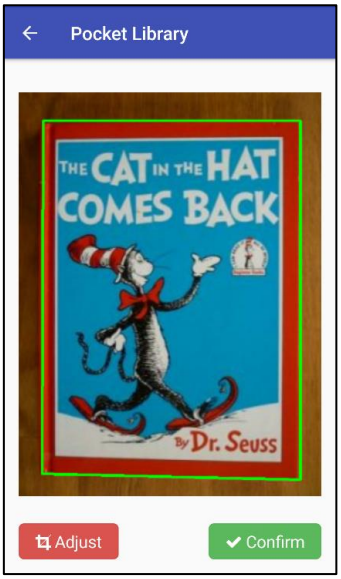


Figura 3-11 Confirmarea ajustării

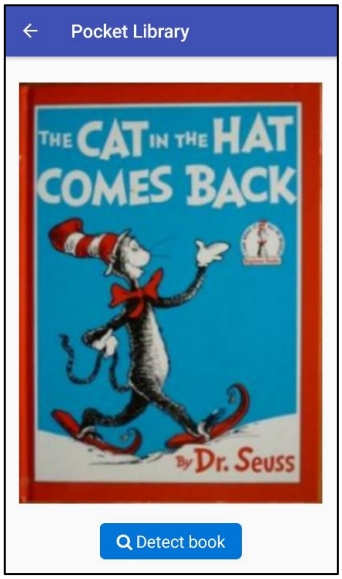


Figura 3-12 Coperta decupata complet

3.3. Detalii de implementare

In secțiunea anterioara, am descris interfața și modul de utilizare cu toți pașii prin care trece o imagine pana la detectarea cărții corespunzătoare. In aceasta secțiune vor fi descrise exact metodele utilizate în implementarea acestor pași.

In primul rând, pe imaginea originala se va aplica algoritmul GrabCut [5] pentru a separa cartea de fundal, în vederea unei mai bune procesări ulterioare. Se presupune ca utilizatorul acestei aplicații va fi bine intenționat în a fotografia o carte pe o anumita suprafață, cartea fiind aproximativ paralela cu axele de coordonate ale imaginii și ocupând în medie peste 75% din continutul acesteia. Având în vedere aceste presupuneri, putem stabili ușor un dreptunghi de inițializare pentru algoritmul GrabCut fără a necesita intervenția utilizatorului. Acesta specifica faptul ca aproximativ 5% din fiecare margine a imaginii conține doar fundal.

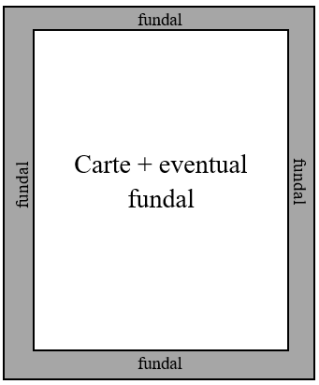


Figura 3-13 Dreptunghi de inițializare pentru GrabCut

După ce se obține imaginea cu fundalul eliminat de către GrabCut, se va aplica algoritmul de detecție a muchiilor al lui Canny, care ne va furniza o imagine alb-negru a muchiilor din imagine.

Motivul pentru care dorim să obținem aceasta imagine a muchiilor este acela ca e mult mai ușor să vizualizam și deci și să detectam un patrulater corespunzător coperții în aceasta. Astfel, pe imaginea muchiilor se va rula algoritmul de detecție a contururilor al lui Suzuki, rezultând o lista de contururi pe care le vom sorta după aria lor. In mod evident, se caută un contur care are aria maxima dintre cele ce pot fi aproximate la un poligon cu patru laturi, iar acesta va fi afișat drept patrulaterul căutat.

Mai departe, folosind transformări geometrice de baza, vom decupa patrulaterul și ii vom ajusta perspectiva pentru a-l transforma într-un dreptunghi. Imaginea rezultata astfel va fi cea folosita în cadrul procesului efectiv de identificare a cărții.

O prima metoda utilizata în detecție este bazat pe extragerea de trăsături din imaginea coperții. Pentru acest lucru s-a utilizat o rețea convolutionala de tipul celor din concursurile ILSVRC [8]. Așa cum a fost menționat într-un capitol anterior, aceste rețele sunt antrenate să extragă trăsături din imagini pe care apoi le folosesc la clasificarea lor. In cazul nostru, vom folosi rețeaua AlexNet [9] doar pentru a extrage un vector de 4096 de trăsături din imaginea data.

Anterior, aceeași rețea a fost folosita pe o baza de date cu imagini de coperți de cărți furnizata de OpenLibrary [13]. Astfel vom putea compara trăsăturile obținute din imaginea curenta cu ce avem în baza de date și utilizând o funcție de distanta vom putea returna lista celor mai apropiate cărți din acest punct de vedere.

O alta metoda de identificare a cărții se bazează pe ideea rulării unui motor OCR pe imaginea coperții. In acest sens, aplicația utilizează Tesseract [15] pentru a identifica textul de pe coperta, ce reprezintă în general titlul și autorii cărții. Mai departe, utilizând spre exemplu Google Books API [14] putem returna o lista de cărți pe baza unei interogări online cu textul identificat de Tesseract.

In continuarea ambelor metode de detecție, mai este necesar doar să afișam detalii despre cărțile identificate în lista de rezultate. Aceste informații sunt obținute utilizând API-ul de la OpenLibrary [13] pentru prima metoda și al celor de la Google Books [14] pentru a doua.

4. Experimente

4.1. Parametrii aleși în preprocesarea imaginii

O buna parte din algoritmii descriși mai sus ca fiind utilizați în cadrul aplicației necesita anumiți parametri pentru a performa bine în scopul aplicației curente.

Un prim astfel de algoritm este GrabCut [5], unde deja s-a oferit și o motivație pentru care experimental s-a ales un dreptunghi de inițializare pentru modelul de fundal având 5% din fiecare margine a imaginii, precum în **Figura 3-13**.

In seria de algoritmi urmează detectorul de muchii Canny [3], care necesita în general ca pas de preprocesare eliminarea zgomotului. Aceasta eliminare a fost făcută cu un filtru Gaussian cu o masca de convoluție de dimensiune 5. Asa cum a fost descris într-un capitol anterior, algoritmul lui Canny necesita doua praguri de confidenta a intensității muchiilor, acestea fiind alese în aplicație egale cu 75, respectiv 200.

Am menționat în capitolul anterior ca următorul pas este să căutam printre contururile returnate de algoritmul lui Suzuki [4] cel mai mare contur ca și arie, care se poate aproxima la un poligon cu patru laturi. Experimental s-au găsit cazuri în care contururile găsite de combinația detectorilor Canny și Suzuki nu au fost deloc cele dorite, din cauza ca patrulaterul mare ce se dorea identificat avea porțiuni ale laturilor care nu se evidențiau bine.

Soluția propusa pentru acest neajuns este bazata pe comparația patrulaterului maxim găsit cu dreptunghiul ce încadrează toate muchiile detectate ale coperții. Experimental s-a decis ca daca raportul ariilor acestor doua patrulatere este mai mic de 0.8 să se aleaga dreptunghiul de incadrare. Rationamentul pentru aceasta alternativa este bazat pe buna intentie a utilizatorului care ar incerca să fotografieze cartea astfel încât să ocupe cat mai mult din imagine și să fie cat mai aliniata la axele de coordonate ale imaginii, asemeni dreptunghiului ales de noi în cazul acesta.

4.2. Metode de extragere a trăsăturilor

Așa cum s-a menționat în secțiunile anterioare, aceasta lucrare si-a propus extragerea trăsăturilor unei imagini folosind rețele convoluţionale dintre cele antrenate pentru concursurile ILSVRC [8].

Fiecare astfel de rețea produce un vector de 4096 de caracteristici după un anumit număr de straturi, înainte de a realiza operațiile necesare clasificării pentru care au fost antrenate.

Un dezavantaj foarte mare al acestei metode de căutare pe baza trăsăturilor imaginii este limitarea impusa de memoria ocupata de baza de date. Astfel pentru a se alinia la un nivel maxim de memorie pentru o aplicație mobila, baza de date nu ar putea conține mai mult de câteva zeci de mii de cărți.

Cu ajutorul unui script MATLAB au fost efectuate mai multe experimente privind relevanta acestor trăsături extrase.

Toate experimentele menționate mai jos au fost efectuate pe o mulțime de testare de 50 de cărți alese dintr-un subset de 10000 al bazei de date pusa la dispoziție de OpenLibrary [13]. Pentru

fiecare dintre cele 50 de cărți s-a ales o imagine a coperții de pe internet, care să nu fie perfect identica cu cea aflata în baza de date. Unele poze alese au conținut și variații mai mici sau mai mari ale elementelor aflate pe coperta, în funcție de ediția aleasa a cărții în cauza.

In **Figura 4-1** putem observa rezultate pentru un prim experiment, anume acuratețea rețelelor testate în a identifica cartea în vârful listei sau în primele 5 rezultate.

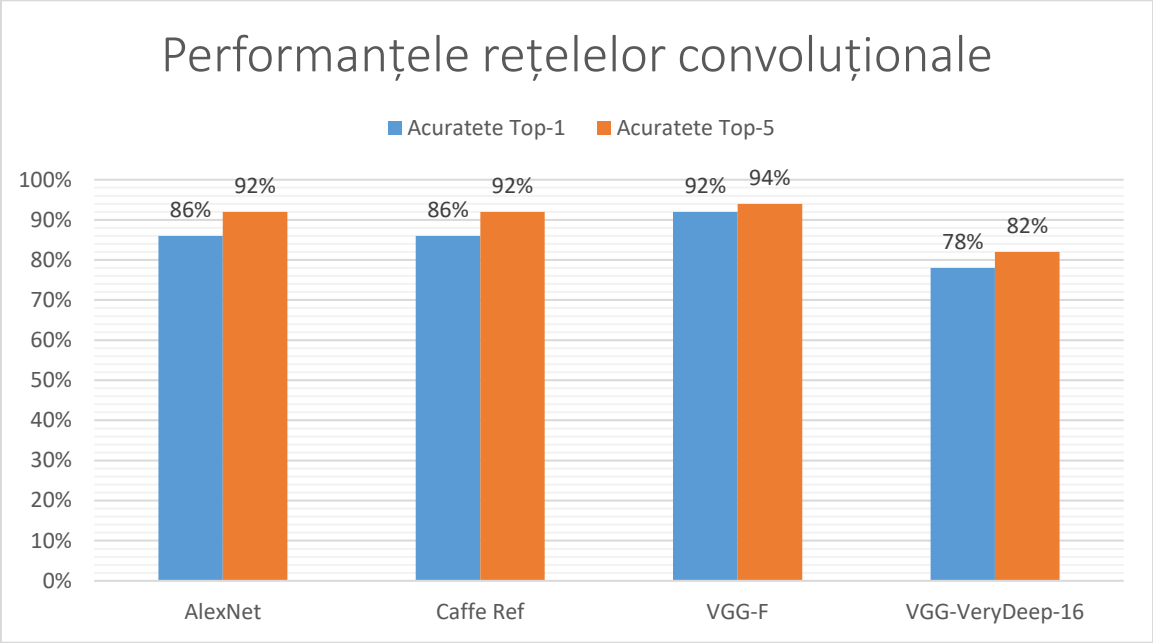
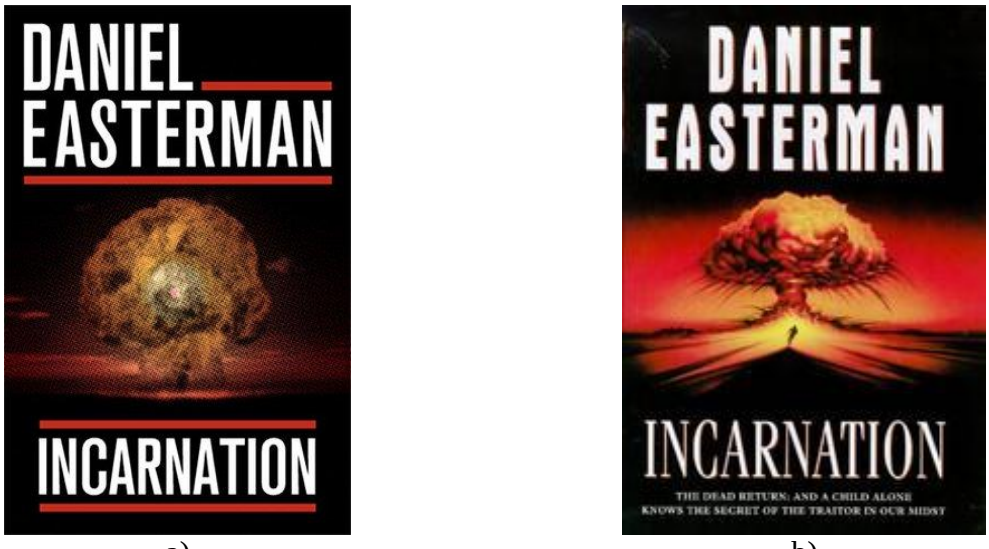


Figura 4-1 Performantele rețelelor convoluționale

Privind aceste rezultate se poate nota faptul o flexibilitate buna la variații ale coperții, cum ar fi poziționări și culori diferite ale elementelor sale. Un exemplu care descrie bine aceasta situație este prezentat în **Figura 4-2**, ce corespunde unei căutări cu succes.



a) b)

Figura 4-2 Exemplu de flexibilitate la variații ale coperții *– a) Imaginea căutată; b) Imaginea returnata din baza de date ca fiind cea mai similara*

Pentru a compara vectorul de caracteristici extrase din imaginea căutată cu vectorii din baza de date, s-au experimentat mai multe funcții de distanță. Conform rezultatelor din **Figura 4-3**, s-a ajuns la concluzia ca cele mai comune doua funcții de distanta utilizate produc rezultate asemănătoare, neinfluențând rezultatele căutării decisiv. Aceste experimente s-au făcut pe aceeași mulțime de testare ca mai sus, utilizând rețeaua convolutionala AlexNet [9] pentru extragerea trăsăturilor.

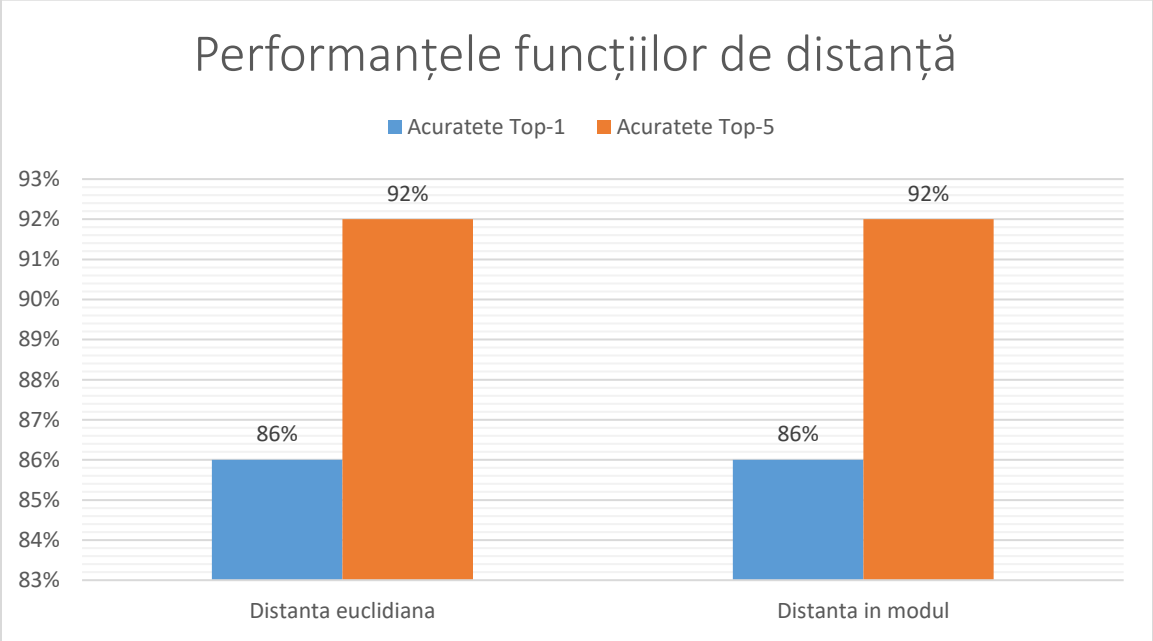


Figura 4-3 Performantele funcțiilor de distanță

4.3. Recunoașterea caracterelor

In cadrul acestei metode de detectare a cărții s-au efectuat mai multe experimente manuale pe coperți de cărți de diverse complexități.

Deși motorul OCR Tesseract [15] este destul de evoluat, totuși domeniul recunoașterii optice a caracterelor rămâne unul deschis cercetătorilor pentru îmbunătățiri.

In cazul coperților de carte, ce conțin destul de puțin text la o dimensiune a fontului relativ mare, Tesseract are rezultate foarte variate.

Pentru coperțile de o complexitate scăzută, ce conțin doar un fundal monocolor și text mare boldat, motorul OCR nu are deloc dificultăți în a întoarce un rezultat perfect. Însă în alte cazuri, textul detectat poate să nu conțină nici măcar un cuvânt dintre cele dorite.

Aceasta instabilitate mare în rezultate se datorează în special diversității de fonturi de text folosite pe coperțile cărților, precum și elementelor grafice variate ce intervin pe fundalul acestora.

După seria de experimente manuale ale acestei funcționalități, s-a ajuns la un procedeu de validare ulterioara a textului rezultat pentru îmbunătățirea acurateței de căutare online. Acest procedeu se folosește de funcționalitățile Tesseract și consta în validarea textului detectat la nivel

de cuvânt pe baza gradului de încredere întors de motorul OCR, care ar trebui să fie de măcar 30%. Textul rezultat din combinarea cuvintelor cu încredere mare, intercalate prin caractere spațiu, are șanse dovedite mult mai mari de a furniza cartea dorita la o căutare online.

5. Concluzii

In capitolele acestei lucrări am putut analiza așadar o serie de algoritmi ce își atinge cele doua scopuri propuse la început.

In primul rând un set de preprocesări bazat pe eliminarea fundalului [5], detecția muchiilor [3] și a contururilor [4], plus decupare și transformare de perspectiva reușește în majoritatea cazurilor să returneze o imagine conținând doar coperta cărții pe care utilizatorul a fotografiat-o.

In continuare, cele doua metode complet diferite de identificare a cărții își ating scopul suficient de bine cu ajutorul unor baze de date cu cărți și a motoarelor de căutare furnizate de acestea [13] [14].

Deși combinația de algoritmi propusa de lucrarea curenta își atinge scopurile demonstrative de rezolvare a problematicii prezentate în motivația ei, exista totuși neajunsuri în aplicabilitatea ei practica.

Metoda bazata pe extragerea trăsăturilor are marea problema de a consuma în primul rând multă memorie pentru a menține o bază de date cu cărți mare. In al doilea rând ar necesita și un timp foarte îndelungat de interogare a unei astfel de baze de date, având în vedere cat de costisitoare sunt operațiile efectuate în timpul comparațiilor dintre vectori de caracteristici. Aceste neajunsuri de memorie și timp sunt foarte importante având în vedere intenția de a utiliza o astfel de aplicație pe un dispozitiv mobil cu resurse limitate. O posibila soluție pe viitor ar fi stocarea bazei de date în cloud și căutarea distribuită pe mai multe mașini server.

Pe de alta parte, metoda bazata pe recunoașterea optica a caracterelor suferă atât din cauza performantelor slabe în acest domeniu la momentul redactării acestei lucrări, cat și din cauza particularităților unor coperți de carte ce cresc complexitatea căutării la un nivel mult prea mare. Pe măsura ce noi metode performante de îmbunătățire a detecției OCR vor fi propuse în viitor, se vor putea rezolva și cazurile cu performantă scăzută ale aplicației curente.

In concluzie, lucrarea aceasta reprezintă un punct de plecare important în rezolvarea problemei detecției automate a cărților pe baza fotografiilor, oferind o suita de algoritmi și observații în acest sens, precum și o aplicație mobila realizata în scop pur academic și demonstrativ, ce ar putea fi îmbunătățită pe viitor pentru a putea fi folosita la scara larga de către public.

Bibliografie

[1] R. Gonzalez și R. Woods, Digital Image Processing, Pearson, 2007.

[2] I. Sobel, *History and Definition of the Sobel Operator,* 2014.

[3] J. Canny, „A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 8, nr. 6, pp. 679-698, 1986.

[4] S. Suzuki, „Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics and Image Processing,* vol. 30, nr. 1, pp. 32-46, 1985.

[5] C. Rother, V. Kolmogorov și A. Blake, „GrabCut - Interactive Foreground Extraction using Iterated Graph Cuts," *ACM Transactions on Graphics,* vol. 23, nr. 3, pp. 309-314, 2004.

[6] Y. LeCun, L. Bottou, Y. Bengio și P. Haffner, „Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, nr. 11, pp. 2278-2324, 1998.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li și L. Fei-Fei, „ImageNet: A Large-Scale Hierarchical Image Database," *IEEE Computer Vision and Pattern Recognition,* pp. 248-255, 2009.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg și L. Fei-Fei, „ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision,* vol. 115, nr. 3, p. 211–252, 2015.

[9] A. Krizhevsky, I. Sutskever și G. Hinton, „ImageNet Classification with Deep Convolutional Neural Networks," în *Advances in Neural Information Processing Systems 25*, Lake Tahoe, 2012.

[10] K. He, X. Zhang, S. Ren și J. Sun, „Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," în *IEEE International Conference on Computer Vision,* Santiago, 2015.

[11] „OpenCV," Intel, [Interactiv]. Available: http://opencv.org/.

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama și T. Darrell, „Caffe: Convolutional architecture for fast feature embedding," în *Proceedings of the 22nd ACM international conference on Multimedia*, Orlando, 2014.

[13] „Open Library," Internet Archive, [Interactiv]. Available: https://openlibrary.org/.

[14] „Google Books API," Google, [Interactiv]. Available: https://developers.google.com/books/.

[15] R. Smith, „An Overview of the Tesseract OCR Engine," în *Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR)*, Parana, 2007.