Ionescu Andrei
gr.333

Documentație proiect:
Detectare și recunoaștere facială

1. Scurtă descriere:

Ținta acestui proiect a fost construirea unui sistem de recunoaștere facială capabil să identifice personajele din seria de desene animate „Scooby-Doo” în afară de Scooby-Doo în sine care este un cățel și nu are o față umană. Implementarea se bazează pe crearea de histograme de gradienți orientați (HOG descriptori) și un clasificator SVM liniar peste care am implementat o funcție de glisare a unei ferestre asupra imaginii și una de eliminare a non-maximelor (pentru detecții care se suprapun).

2. Metodologie:

2.0 Citirea Imaginilor

- Ca date de plecare am un set de imagini pentru fiecare personaj și o listă de fișiere text care conținea informații care ne-au permis extragerea fețelor (numele imaginii, coordonate și numele personajului).

- Având în vedere proporțiile similare ale fețelor personajelor din serial, am decis, ca mod de abordare, folosirea unei singure mărimi de imagine la care voi redimensiona fețele extrase. Am calculat media înălțime/lungime pentru toate fețele din setul de date și am ajuns la numărul 1,3123… Pentru a păstra niște numere rezonabile cu divizori comuni am ajuns la dimensiunea de 36x48 pixeli cu media înălțime/lungime 1,(3).

- Pentru că nu am fost încredințat un set de date cu imagini negative (fundal, obiecte, animale, etc.) am extras din imaginile care conțineau personaje exemplele negative de care aveam nevoie. (am evitat intersecția cu fețele folosindu-mă de coordonatele menționate mai devreme).

2.1 Extragerea de caracteristici (HOG)

Am folosit HOG (Histograma de Gradienți Orientați) pentru a reprezenta conținutul vizual al patch-urilor menționate mai devreme.

- Marimea ferestrei: 36x48 pixeli

- Marimea celulelor: 6x6 pixeli
- Marimea blocurilor: 2x2 celule

Am ales HOG deoarece este un model bun pentru reprezentarea fețelor: ignoră eficient schimbări în lumină și umbre și capturează bine muchii precum gura, sprâncenele și pleoapele ochilor unui personaj reținând orientarea acestora.

2.2 Antrenarea Clasificatorilor (SVM Liniar)

Am antrenat patru clasificatori diferiți pentru fețele fiecărui personaj în parte plus unul general pentru fețele tuturor personajelor (inclusiv alte personaje din poze în afară de cei patru). La această etapă am avut grijă să am o rată pozitive - negative în jur de 1 la 5 pentru a balansa setul de date.

- Pentru cele patru personaje am folosit ca exemple pozitive imaginile clasificate în etapa de citire/tăiere a fețelor doar cu personajul respectiv și aceleași imagini doar că oglindite orizontal (2000 - 3000 exemple / personaj), iar, ca exemple negative, toate celelalte fețe plus oglindirea lor și 4000 de exemple cu fundal.
- Pentru clasificatorul general, am folosit ca exemple pozitive toate fețele plus oglindirea lor (total aproximativ 13.000), iar, ca exemple negative, toate exemplele de fundal (40.000).

Am ales algoritmul „Linear Support Vector Machine” (Linear SVM) pentru că merge excepțional de bine cu caracteristici de tip HOG și are un trade off bun calitate timp față de alte SVM-uri.

2.3 Detecția

Procesul de detecție pentru o imagine test funcționează în felul următor:

1. Funcția de glisare a unei ferestre pe o imagine la dimensiuni diferite: Deoarece fețele pot apărea la variate distanțe în imagini, nu putem să ne bazăm că fața va fi detectată din prima.
    a. Dimensiunea imaginii este modificată de 6 ori (100%, 90%,... 40%)
    b. Pentru fiecare dimensiune o fereastra de 36x48 pixeli glisează de-a lungul imaginii
    c. Caracteristici HOG sunt calculate pentru fiecare poziție diferită a ferestrei. Cu aceste caracteristici sunt trecute prin SVM si oferite un scor
    d. Coordonatele sunt salvate (având grijă să corespundă imaginii originale și nu celei redimensionate)
2. Folosim un prag mic (0) în etapa de detecție pentru a maximiza Recall-ul (găsirea tuturor fețelor). Detecțiile cu un scor de încredere mai mare vor influența precizia medie a modelului mult mai mult.
3. Din cauza faptului că fereastra glisantă verifică în intervale de 6 pixeli, se întâmplă ca aceeași față să apară de mai multe ori ca detecție. În această situație am implementat funcția de eliminare a non-maximelor care, dintre

două ferestre suprapuse (30% sau mai mult), o păstrează pe aceea cu scorul cel mai mare.

3. Dificultăți Întâmpinate:

3.1 Lipsa Imaginilor Negative Pure

Absența unui set de imagini exclusiv cu fundaluri a îngreunat procesul de "Hard Negative Mining". Deși am luat în considerare implementarea minării pe baza intersecției cu adnotările, am decis să ne bazăm pe un set inițial robust de negative randomizate pentru a evita riscul de a introduce fețe reale în setul de negative (contaminarea datelor).

3.2 Problema Calibrării Scorurilor

În faza de optimizare pentru Task-ul 2, am încercat o abordare de tip "Conflict Resolution": dacă doi clasificatori (ex: Fred și Shaggy) revendicau aceeași regiune, păstram doar detecția cu scorul mai mare.

- Această metodă a eșuat, scăzând performanța drastic.
- Cauza: Deoarece modelele au fost antrenate independent, scorurile lor nu sunt calibrate între ele. Un model "zgomotos" (ex: Shaggy) tinde să ofere scoruri medii mai mari (ex: 3,5) decât un model conservator (ex: Fred, scor 1,5). Astfel, modelul cu scoruri mai mari a suprimat incorect detecțiile valide ale celorlalte modele.
- Soluția Finală: Am renunțat la filtrarea inter-clase și am permis raportarea tuturor detecțiilor independente, lăsând metrica de mAP să gestioneze confuziile.

4. Biblioteci Utilizate

- *OpenCV (cv2):* Procesare imagini, redimensionare, citire/scriere.
- *Scikit-image (skimage.feature.hog):* Calculul descriptorilor HOG.
- *Scikit-learn (LinearSVC):* Implementarea și antrenarea clasificatorului SVM.
- *Numpy:* Operații matriciale și manipularea coordonatelor.
- Pickle: Serializarea și deserializarea modelelor antrenate (salvarea clasificatorilor în format .pickle pentru utilizare ulterioară).
- OS/Time/Datetime: Pachete standard Python utilizate pentru parcurgerea sistemului de fișiere, monitorizarea performanței și estimarea timpului rămas de execuție