Detectarea si recunoasterea faciala a personajelor din serialul The Flintstones

Ghețoiu Gheorghe-Laurențiu

Concepte si Aplicatii in Vederea Artificiala - Tema 2
Jan 18 2023

1 Problema si etapele de rezolvare

Scopul temei este implementarea unui program care identifica fete in imagini din serialul animat "The Flintstones" alaturi de recunoasterea unor personaje principale din serial.

1. Generarea exemplelor pentru antrenare
2. Descriptori HOG
3. Retele convolutionale
4. Task 2
5. Modelul YOLO
6. Bibliografie

2 Rezolvarea

Generearea exemplelor pentru antrenare

Crearea exemplelor pozitive si negative este prima etapa necesara pentru aplicarea oricarui model. Dimensiunea aleasa pentru aceste exemple este de 64x64. Alegerea dimensiunii a fost realizata in urma unei analize manuale a dimensiunilor unor anotari alese la intamplare astfel incat sa fie o dimensiune care se apropie de dimensiunea reala a esantionului de fete alese din imagini. Exemplele de antrenare sunt extrase conform adnotarilor primite si redimensionate la 64x64.

Pentru exemplele negative am folosit aceeasi dimensiune ca cea pentru datele de antrenare si anume 64x64. In prima faza am blurat fetele din adnotari, alegand patch-uri random din imaginile cu fetele blurate. Asadar, dupa extragerea

fetelor le-am blurat in imaginile originale din care ulterior sunt extrase patch-urile negative.

```
img_fata_croped = img[y_min:y_max, x_min:x_max]
img_fata_croped = cv.resize(img_fata_croped, (64, 64))
cv.imwrite(os.path.join("pozitive_total", str(nr).zfill(5) +
    ".jpg"), img_fata_croped)
img[y_min:y_max, x_min:x_max] =
    cv.GaussianBlur(img[y_min:y_max, x_min:x_max], (43,43), 50)
...
cv.imwrite(os.path.join("negative_total", str(nr).zfill(5) +
    ".jpg"), img)
```

Pentru urmatorul pas am ales sa modific si alegerea exemplelor negative. Astfel, pe langa blurarea fetelor considerate pentru exemplele pozitive, exemplele negative sunt alese ca patch-uri care au mai putin de un treshold x ca intersection over union. In acest sens am modificat doar functia *get_negative_descriptors*. Patch-urile negative dupa modificari arata astfel:

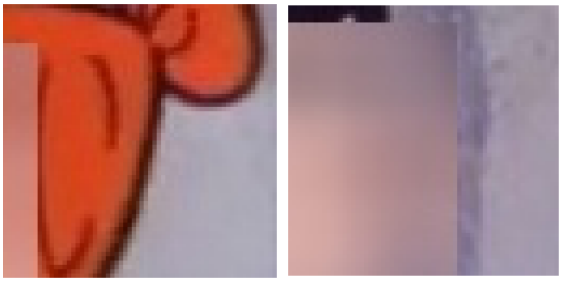

Figure 1: Patch-uri negative

Descriptori HOG

Pentru detectia fetelor am folosit un model antrenat pe baza descriptorilor HOG. Am modificat codul din laborator in mai multe moduri pentru a imbunatatii rezultatele pe setul de date.

Average precision-ul obtinut pentru HOG cu aceste imagini de antrenare si cu modificari ale parametrilor a fost 0.40. Am folosit un numar similar de exemple negative cu cel de exemple pozitive(10 000 cu flip), dar acest model a obtinut un AP de sub 0.30. Apoi am dublat numarul de exemple negative (20 000) obtinand un AP de peste 0.4 pentru diferite dimensiuni ale ferestrelor de descripori.

De asemenea, am modificat modelul pentru predictie intr-o **Regresie logistica** care perfoma mai bine pe setul meu de date(patch-urile generate).

O observatie realizata in urma incercarilor a fost ca modelul, pe langa identificarea aproximativ corecta a fetelor, identifica si multe alte non-fete din imag-

ine. Asadar, exista multe Fals Pozitive. In urma acestei observatii am ales sa cresc progresiv numarul de exemple negative. Astfel, am ajuns la utilizarea a intre 50 000 si 70 000 de patch-uri ca exemple negative

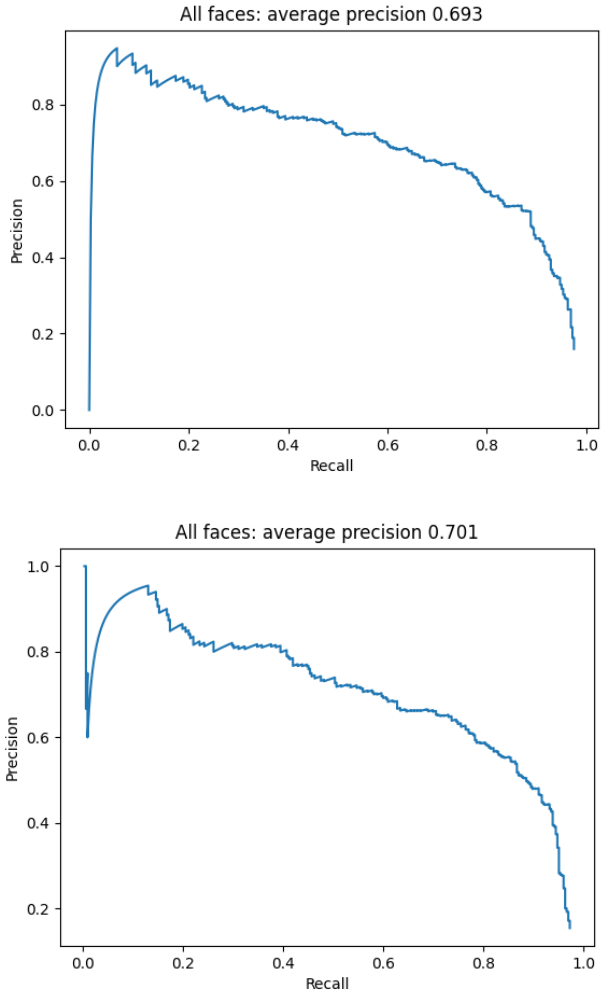

Figure 2: AP HOG

Aceasta modificare a adus imbunatatiri semnificative modelului.Pe langa aceasta modificare am adaugat modelului HOG si parametrul *orientations*, contribuind si el la imbunatatirea modelului. Pentru aceasta stare am obtinut AP aproximativ 0.70.

Un alt pas care a dus la o imbunatatire importanta a fost gasirea unor parametrii de redimensionare potriviti pentru setul de date. Astfel, facand grid

search pentru acesti parametrii am gasit ca o scalare la dimensiuni prea mari aduce un efect negativ, prin adaugarea de numeroase fals pozitive. In schimb, o scalare la dimensiuni foarte mici nu afecteaza modelul in mod sesizabil(scalare 0.2 - 1.3 cu pas 0.1). Cele mai bune rezultate sunt obtinute pentru o dimensiune a celulei de 8 si parametrul *orientations* = 64. In final, pentru modelele bazate pe descriptori HOG problema este reprezentata in continuare de numarul ridicat de fals pozitive, mai ales pe imagini care contin forme circulare sau au luminozitatea scazuta.

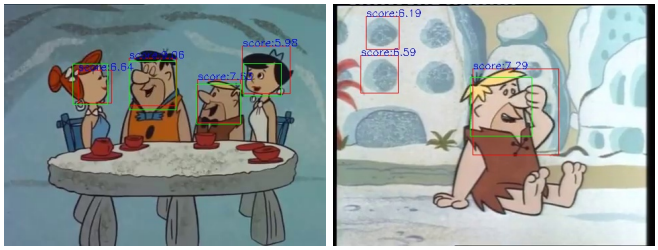

Figure 3: Exemplu bun(stang) Exemplu slab(dreapta)

Retele convolutionale

Un model incercat, dar care nu este folosit in solutia finala este utilizarea unei retele convolutionale pentru extragerea trasaturilor. Pentru configuratia modelului si setul de date dat rezultatele obtinute sunt mai bune decat cele obtinute cu descripori HOG(0.8 AP), dupa cum se observa in Figure 4. In schimb, timpul creste considerabil atat la antrenare cat si la inferenta, pe un GPU normal fiind de aproximativ 100-120 minute, sau 5-6 ore pentru ambele task-uri.

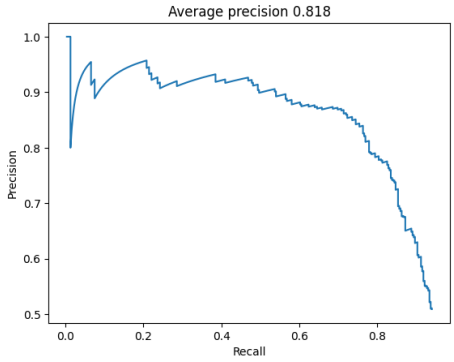

Figure 4: AP pentru CNN

Task 2

Pentru task-ul doi am folosit aceiasi configuratie pentru descriptori ca pentru primul, schimbarile care au intervenit fiind la generarea datelor. Astfel, pentru patch-urile negative sunt blurate doar fetele personajului care trebuie identificat pentru a exista o diferentiere intre personaje. De asemenea este modificat si treshold-ul pentru care o imagine este considerata fata(0.25). AP-ul obtinut pentru acest task se poate observa in grafice. Setul de antrenare cu exemple pozitive pentru acest task este alcatuit din imagini de dimensiune 64x64 reprezentand fetele personajului respectiv. Exemplele negative sunt reprezentate de patch-uri care contin obiecte din fundal, corpuri ale personajelor si fete ale personajelor, inafara celui pentru care antrenam modelul.

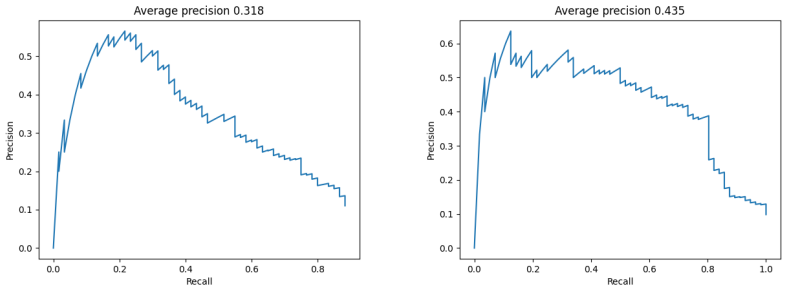

Figure 5: AP Wilma(stanga) si Betty(dreapta)

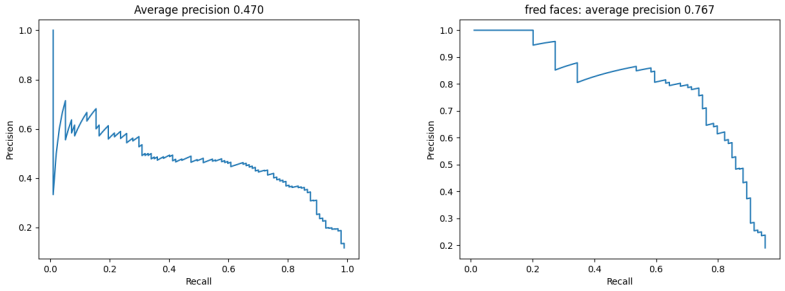

Figure 6: AP Barney

Modelul YOLO

YOLO(You Only Look Once) este un algoritm de deep learning folosit pentru diverse task-uri de vedere artificiala printre care detectarea obiectelor si recunoasterea faciala. Acesta functioneaza impartind imaginea intr-un grid si trecand o singura data prin fiecare imagine din grid. De aici provine atat numele, cat si rapiditatea algoritmului.

Figure 7: Yolo Format - Preluat din documentatia ulralytics YOLOv8 (Sursa)

In cazul task-ului nostru, ce am avut de facut a fost adaptarea datelor la modul in care YOLO le poate folosi. Astfel a fost necesara adaptarea imaginilor in formatul xywh. Acesta presupune ca pornind din centrul ferestrei noastre sa furnizam inaltimea si latimea raportate la o imagine intr-un sistem de coordonate cu originea in $(0, 0)$ reprezentand coltul stanga sus al imaginii si cu latime si inaltimea maxima 1.

```
x_center = int(x_min) + (int(x_max) - int(x_min)) / 2
y_center = int(y_min) + (int(y_max) - int(y_min)) / 2
width = int(x_max) - int(x_min)
height = int(y_max) - int(y_min)
x_center = x_center / im_width
y_center = y_center / im_height
width = width / im_width
height = height / im_height
last_file_name = file_name
```

Bibliografie

1. Cod laborator
2. Documentatie ultralytics, YOLOv8
3. Documentatie pytorch