Bouruc Petru-Liviu
Grupa 334

Documentație – Tema 2

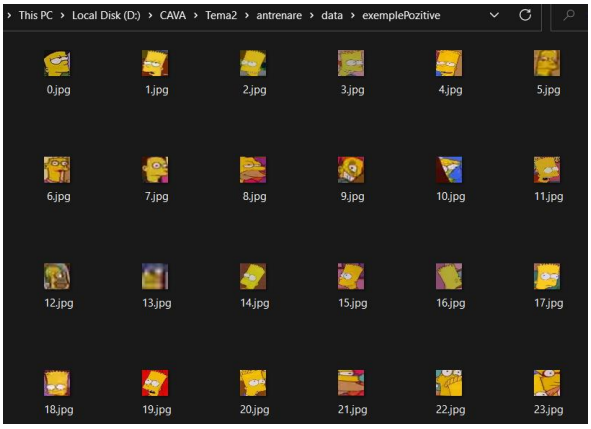
Detectarea și recunoașterea facială a personajelor din serialul de desene animate Familia Simpsons

Taskul 1 – Detectarea fețelor

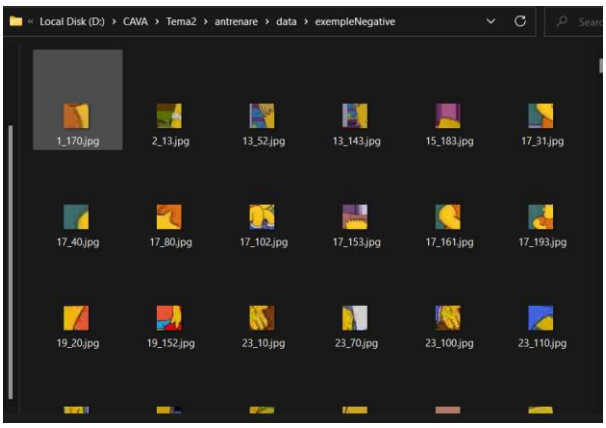
Pentru început, am pornit în realizarea temei de la codul dat în laboratoarele 8, 9, 10 și 11.

Astfel, primul pas a fost sa extrag exemple pozitive și exemple negative din datele de antrenare pentru a le folosi ulterior la extragerea descriptorilor și la antrenarea modelului. Pentru exemplele pozitive am păstrat formatul de la laborator, iar pe lângă aceasta am calculat și diferențele minime și maxime dintre înălțimi și lățimi pentru a putea observa între ce valori variază bounding box-urile.

La exemplele negative, am avut următoarea abordare: pentru început iau patch-uri de dimensiune variabilă, pentru a putea extrage date mai complexe. Astfel, plec inițial cu patch-uri de dimensiuni 36x36, urmând ca apoi să le cresc progresiv dimensiunile de 1.5 ori. Așadar voi avea si exemple negative care se pretează atât pe imagini de dimensiuni mai mici, cât și pe imagini de dimensiuni mai mari, fapt care mă va ajuta atunci când fac sliding window pe dimensiuni variabile. Toate exemplele negative le redimensionez la 36x36 și le salvez. De altfel, atunci când generez puncte random în generarea imaginilor negative, verific dacă acest punct nu este generat în interiorul unui bounding box adnotat corespunzător imaginii respective (putem avea mai multe personaje în aceeași imagine), cu o marja de 18 pixeli în exterior (să depășească puțin bounding box-ul). În plus, aplic un filtru pe galben patch-urilor alese pentru a le salva doar pe cele care au o proporție de galben mai ridicată, deoarece voi folosi filtru pe galben și la sliding window. astfel acolo voi trimite la clasificat doar bucăți galbene, motiv pentru care este bine ca antrenarea modelului să fie cu patch-uri cât mai multe galbene.



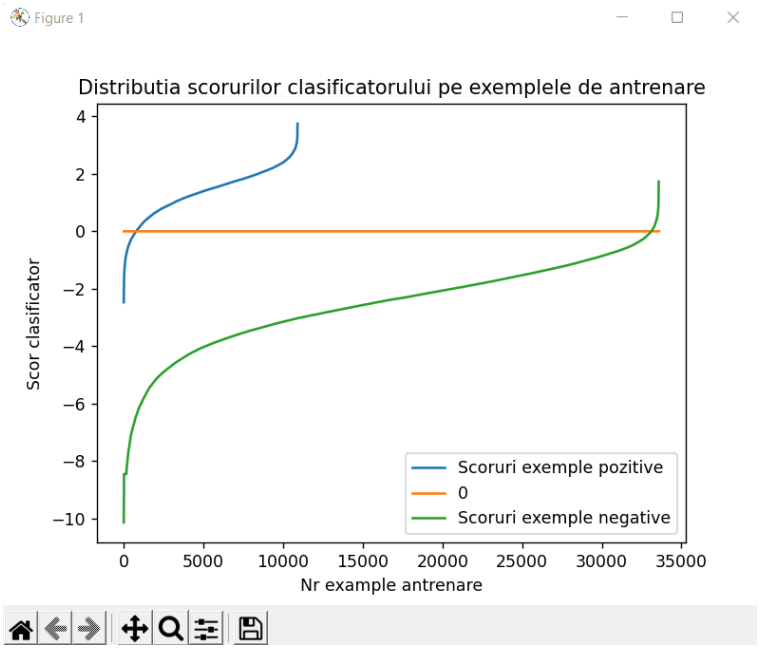
exemple pozitive



exemple negative

Programul principal îmi verifică apoi dacă am descriptorii salvați. Dacă nu, ii generează din nou iar apoi antrenează clasificatorul care îmi spune dacă am sau nu față. Pentru extragerea descriptorilor

pozitivi și negativi am păstrat hog-ul dat, cu *use_flip_images = True*, iar prin trial and error am modificat dimensiunea celulei hog de la 4 la 6. De asemenea, pentru antrenarea clasificatorului am păstrat LinearSVC-ul care se antrenează pe mai multe valori pentru *C*.



Următorul pas este să aplic sliding window peste fiecare imagine iar aici am implementat un sliding window dinamic și un filtru pe galben.

Astfel, sliding window-ul dinamic l-am făcut prin redimensionarea progresivă a imaginii. Am plecat de la o dimensiune mică urmând a incrementa proporțional scala la care fac redimensionarea. Aceasta redimensionare poate depăși valoarea 1, poza fiind astfel practică mărită în loc de micșorată, deoarece, la extragerea feature-urilor pozitive am observat faptul că pot avea bounding box-uri și de dimensiune foarte mică, o mărire a imaginii m-a ajutat în a le putea identifica chiar și pe cele mai mici. De asemenea, tot pentru redimensionare am folosit proporții diferite pentru cele 2 axe, deoarece fețele adnotate pentru bart si homer sunt mai mult dreptunghiulare, astfel pentru axa y voi micșora mai mult imaginea, pentru a o aduce la o dimensiune mai apropiata de pătrat, deoarece sliding window-ul meu este de forma pătrată.

```
scale = [0.15, 0.12]
while max(scale) <= 1.5:
    img = cv.resize(original_image, (0, 0), fx=scale[0], fy=scale[1])
```

```
    scale[0] *= 1.02
    scale[1] *= 1.02
```

redimensionarea dinamică

Bouruc Petru-Liviu
Grupa 334

Filtrul pe galben l-am făcut cel din laborator. Pentru fiecare imagine, am calculat mask-ul său după filtrare. Dacă atunci când fac sliding window în patch-ul corespunzător poziției actuale a ferestrei, în *original_mask_yellow_hsv* am o medie mai mare decât 70, atunci pot considera că am mult galben în patch și continua astfel cu clasificarea pentru a-mi spune dacă acolo există o față.

```
for y in range(0, num_rows-num_cell_in_template):
    for x in range(0, num_cols-num_cell_in_template):
        x_min = int(x * self.params.dim_hog_cell * 1//scale[0])
        y_min = int(y * self.params.dim_hog_cell * 1//scale[1])
        x_max = int((x * self.params.dim_hog_cell + self.params.dim_window) * 1//scale[0])
        y_max = int((y * self.params.dim_hog_cell + self.params.dim_window) * 1//scale[1])

        if original_mask_yellow_hsv[y_min:y_max, x_min:x_max].mean() > 70:
            descr = hog_descriptor[y:y+num_cell_in_template, x:x+num_cell_in_template].flatten()
            score = np.dot(descr, w)[0] + bias

            if score > self.params.threshold:
                image_detections.append([x_min, y_min, x_max, y_max])
                image_scores.append(score)
```

sliding window cu filtru pe galben

La final, detectiile, scorurile și numele fișierelor dupa ce este aplicat *non_maximal_surpression* sunt trimise către *RunProject.py* unde se face evaluarea.

Taskul 2 – Recunoașterea personajelor

Pentru această parte, am creat o nouă clasă în care am implementat un SVC pentru clasificarea personajelor.

Acest clasificator îl inițializez înainte să încep sliding window-ul și totodată încep antrenarea sa.

Pentru train, parcurg toate imaginile date ca exemple pozitive, adnotate, iau fiecare față de personaj și o adaug în datele de train, cu label-ul corespunzător: fiecărui personaj îi corespunde un număr de la 0 la 3 (bart, homer, lisa, marge), urmând ca apoi să folosesc aceste date la train. Deoarece trebuie să clasific doar cele 4 personaje principale, personajele unknown nu le iau în considerare. Un personaj care nu este principal va fi clasificat cu una din cele 4 clase, doar nu ne interesează.

```
train_images = np.array(train_images)
train_labels = np.array(train_labels).astype('int')

self.svm_classifier.fit(train_images, train_labels)
```

antrenarea clasificatorului pentru personaje

Bouruc Petru-Liviu
Grupa 334

Următorul pas este ca atunci când prin sliding window clasificatorul care îmi spune dacă este față sau nu îmi detectează o față, să o trimit pe aceasta la clasificatorul pentru recognition. Acest lucru nu îl fac chiar atunci când îmi detectează o față care trece de threshold, deoarece este posibil să am foarte multe detecții care pot fi suprimate prin non maximal surpression. Astfel, clasificarea personajelor o realizez după, iar pentru fiecare personaj salvez detecțiile, scorurile și file name-urile corespunzătoare pentru a le putea trimite ulterior la *eval_detections*.

```
def classify(self, image):
    img = cv.resize(image, (36, 36)).flatten()
    img = img.reshape(1, -1)
    return self.svm_classifier.predict(img)
```

metoda care îmi clasifica un personaj

Pentru acest clasificator, am ales să îi dau ca input imagini grayscale, 36x36.

La final, apelez funcția *eval_detections* pentru fiecare personaj în parte, cu detecțiile, scorurile și file name-urile corespunzătoare.