Tema 2 - Moroianu Theodor - 334

Idea Din Spatele Solutiei

Pentru a rezolva problema, am decis sa folosesc o solutie bazata pe un CNN.

Flow-ul solutiei este urmatorul:

1. Pozele sunt incarcate din fisierul de testare/validare.
2. Modelul este incarcat din `model/model.th`.
3. Trece un sliding-window, cu mai multe dimensiuni, peste imagine.
4. Fiecare sliding window este trecut prin retea, pentru a decide daca este sau nu o fata.
5. Independent pe fete (fete generale / cei patru isteti / fete necunoscute), efectuez un filtru de non-maximal supresion.
6. Afisez ferestrele ramase.

Modelul

Am incercat diverse modele, atat din Keras cat si Pytorch. Modelul care mi-a dat acuratetea cea mai buna este definit in fisierul `sources/network_pytorch.py`, si este definit de codul urmator:

```
net = nn.Sequential(
    nn.Conv2d(3, 32, kernel_size=5, padding=2),
    nn.ReLU(),
    nn.Conv2d(32, 64, kernel_size=5, padding=2),
    nn.ReLU(),
    nn.MaxPool2d(2),
    nn.Conv2d(64, 128, kernel_size=3, padding=1),
    nn.ReLU(),
    nn.MaxPool2d(2),
    nn.Conv2d(128, 128, kernel_size=3, padding=1),
    nn.ReLU(),
    nn.MaxPool2d(2),
    nn.Flatten(),
    nn.Dropout(0.2),
    nn.Linear(128 * (constants.SIZE_FACE_MODEL // 8)**2, 128),
    nn.ReLU(),
    nn.Dropout(0.2),
    nn.Linear(128, 50),
    nn.ReLU(),
    nn.Linear(50, 6)
)
```

Putem vedea ca este o ingramadire clasica de convolutie + ReLU + MaxPool.

Pentru datele de antrenare, am folosit atat pozele date pentru antrenare (dupa ce am sters anotarile gresite), cat si poze anotate de mine dintr-un dataset luat de pe net cu screen-shooturi din simpsoni.

Sliding Window

Pentru sliding window, implementat in `sources/sliding_window.py`, imi aleg diferite dimensiuni ale ferestrei, pe care o trec prin CNN dupa ce o redimensionez la o dimensiune standard, definita in fisierul `sources/constants.py`.

Codul care efectueaza sliding-window-ul este:

```
window_dim = min(img.shape[0], img.shape[1])

while window_dim >= constants.MINIMAL_WINDOWS_PIXEL_SIZE:
    stride = int(window_dim * constants.SLIDING_WINDOW_STRIDE)

    for ymin in range(0, img.shape[0] - window_dim + 1, stride):
        for xmin in range(0, img.shape[1] - window_dim + 1, stride):
            sliding_window = img[ymin:ymin+window_dim,
xmin:xmin+window_dim, :]

            # procesez sliding_window
            # ...

    window_dim = window_dim * constants.SLIDING_WINDOW_RESCALE_FACTOR
    window_dim = int(window_dim)
```

Non Maximal Supresion

Am implementat non-maximal supresion si intersection-over-reunion in fisierul `sources/project_utils.py`, pentru a le putea folosi in mai multe componente ale proiectului.

Rularea Codului

Pentru a rula codul, este suficient sa urmariti instructiunile din `README.txt`.

Observatii

1. Reteaua ruleaza in ~10 secunde pe o imagine de validare pe GPU-ul meu (GTX 1060).
2. Pe datele de validare am o acuratete de `88%` pe ambele taskuri.
3. Nu am avut nevoie sa trec pozele in spatiul HLS si sa fac jmecherii pentru a procesa numai zonele galbene, avand in vedere ca atat timpul cat si performanta solutiei considerand orice zona sunt satisfacatoare.