

Facultate : Automatica si Calculatoare

Specializare: Calculatoare

Disciplina : Sisteme de recunoastere a formelor

Proiect : Recunoasterea obiectelor folosind YOLO

Student : Herman Felician-Nicu

Grupa : 30244

Cuprins

[1. Prezentarea Temei 3](#_Toc73461612)

[2.Functionalitati 3](#_Toc73461613)

[3.Detalii de implementare 3](#_Toc73461614)

[3.1 Structura proiectului 3](#_Toc73461615)

[3.2 Functii si algoritmi 4](#_Toc73461616)

[3.3 Structuri de date si biblioteci 5](#_Toc73461617)

[4.Manual de utilizare 6](#_Toc73461618)

[5.Concluzii si dezvoltari ulterioare 7](#_Toc73461619)

[6.Referinte 7](#_Toc73461620)

# 

# 1. Prezentarea Temei

Modelul Yolo(You only look once) este implementat astfel incat se parcurge imaginea de test o singura data si pe baza unor date deja memorate sunt recunoscute toate obiectele.

Proiectul ne introduce in zona de machine learning, avand un model deja antrenat, disponibil pe internet, noi trebuind sa realizam partea de configurari.

Modelul este antrenat pe o clasa de obiecte, mai exact 80, printre care caini, girafe, calculatoare, foarfece si multe altele.

# 2.Functionalitati

Ca si prima parte, am inceput prin a prelua argumentele din linia de comanda astfel, imaginea pe care o sa testam, fisierul de config, modelul antrenat si clasele. In acest mod ne este foarte usor sa adaptam proiectul si la alte modele invatate, avand in vedere rularea codului din linia de comanda.

Dupa ce o sa se ruleze in linia de comanda urmatoarele : python main.py(numele proiectului) --image test.jpg –config yolov3.cfg –weights yolov3.weights –classes yolov3.txt, proiectul o sa afiseze imaginea test in care obiectele recunoscute o sa fie incadrate intr-un chenar, cu numele lui deasupra.

# 3.Detalii de implementare

## 3.1 Structura proiectului

Pe langa sursa python main, proiectul mai contine si fisierele pe care o sa le transmitem liniei de comanda pentru a putea rula.

Folderul contine exemple de imagini care pot sa fie testate, fisierul de configurare, yolov3.cfg, fisierul cu obiectele pe care modelul le recunoaste, yolov3.txt si modelul antrenat sa recunoasca setul de obiecte, yolov3.weights.

## 3.2 Functii si algoritmi

Incep prin a realiza procesarea liniei de comada, adaug toate argumentele si le parsez, avand o structura cu tot ce avem nevoie. Mai apoi, dimensionam imagina ca sa aiba inaltimea si latimea divzibile cu 320, altfel modelul o sa esuze.

Avand imaginea pregatita,citim din fisierul claselor toate obiectele pe care o sa le recunoastem si le memoram, pentru a putea scrie deasupra chenarelor denumirea fiecaruia.

Continuam prin a incarca modelul gata antrenat si toti parametrii din fisierul configurarilor. Avand clasele citite, le sincronizam cu obiectele din fisierul de config si cu cele din model. Ca final avem surpresia non-maximelor, astfel incat daca avem mai multe dreptunghiuri care ne incardeaza un anumit model, se ia cel mai bun dintre toate si se afiseaza.

Pentru a afisa chenarul care evidentiaza obiectul si numele acestuia, am implementat functia draw\_bounding\_box() care alege o culoare random, deseneaza dreptunghi in jurul cuvantului cu functia cv2.rectangle si afiseaza textul deasupra.

In final se parcurge imaginea si pentru toate detectiile de obiecte se deseneaza dreptunghiul care inconjoara obiectul si se afiseaza denumirea acestuia. Pe langa asta, o sa salvam ultima imagine testata pe disc cu functia cv2.imwrite().

## 3.3 Structuri de date si biblioteci

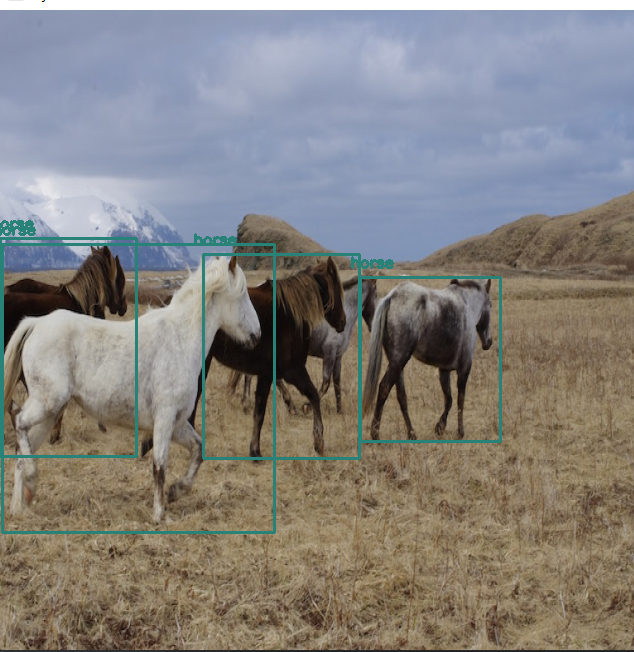
Ca si biblioteci principale, am folosit „cv2”, care cuprinde majoritatea functiilor din OpenCV pentru lucrul cu imagini, diferite transformari sau operatii find disponibile. Deasemenea, pentru importarea imaginilor avem nevoie de „os” care e o biblioteca legata de sistemul de operare unde ne este permisa listarea diferitelor directoare din anumite foldere.

Pe langa asta am folosit si numpy si argparse pentru a face parsarea liniei de comanda.

# 4.Manual de utilizare

Dupa cum ziceam, verificarea proiectului se face folosind linia de comanda. Ca de exemplu daca vrem sa afisam imaginea si toate obiectele din ea, pentru o imagine denumita sample\_horses.jpg o sa trebuiasca sa scriem in linia de comanda astfel :

>python main.py –image sample\_horses.jpg –config yolov3.cfg –weights yolov3.weights –classes yolov3.txt

Rezultatul fiind urmatorul:

# 

# 5.Concluzii si dezvoltari ulterioare

Versiunea 3 a modelului Yolo este cea pe care am ales-o, fiind disponibile mai multe versiuni, de asemenea, se poate face si o comparatie intre eficienta fiecareia in parte. Pe langa model, clasele pot sa fie modificate, modelul sa recunoasca diferite obiecte.

Proiectul poate sa fie dezvoltat implementand o noua versiunea si fiind vizibila eficienta in cazul fiecareia.

# 6.Referinte

<https://towardsdatascience.com/implementing-yolo-on-a-custom-dataset-20101473ce53>

<https://pjreddie.com/darknet/yolov2/>

<https://towardsdatascience.com/yolo-object-detection-with-opencv-and-python-21e50ac599e9>

<https://www.youtube.com/watch?v=ag3DLKsl2vk>

<https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/>

<https://appsilon.com/object-detection-yolo-algorithm/>