Laborator 7 SRF: Folosirea rețelelor neuronale pentru clasificarea obiectelor. Libraria DeepVision

Scopul lucrării

Acest laborator are scopul de familiarizare a studenților cu folosirea rețelelor neuronale deja optimizate și antrenate. Librăria ce definește toate rețelele pe care studenții le vor folosi în acest laborator este librăria DeepVision. La finalul acestui laborator studenții vor fi capabili să încarce și să testeze imagini pentru a calcula eficiența unei rețele neuronale.

1. Breviar teoretic:

O rețea neuronală funcționează analog sistemului nervos al ființelor vii. Nodurile acestei rețele, prin analogie cu biologia, se vor numi neuroni, în timp ce intrările în acest neuron au rolul dendritelor, iar ieșirea reprezintă axonul. Sinapsele sunt punctele de contact între axonul unui neuron de pe un strat precedent și una din dendritele neuronului de pe stratul următor. Ponderile și funcțiile sunt analogice felului în care se formează potențialul de activare în rețelele biologice.

In Processing, cea mai completă librărie de rețele neuronale cu aplicații in Computer Vision este librăria DeepVision. După instalarea ei asemănător felului în care s-a făcut instalarea librăriilor Video și OpenCV în laboratoarele precedente, se va urma o serie de pași pentru a putea folosi metodele definite de această clasă.

Pas1: Importarea librariilor:
ch.bildspur.vision.*; ch.bildspur.vision.result.*; ch.bildspur.vision.network.*;

Pas2: declararea variabilelor:
- un obiect din clasa DeepVision: DeepVision vision;
- o rețea neuronală: YOLONetwork network;
- o listă de obiecte detectate: ResultList<ObjectDetectionResult> detections;

Pas3: inițializarea variabilelor:
- obiectul vision: vision=new DeepVision(this);
- obiectul network: network=vision.createYOLOv3();

Pas4: instalarea rețelei neuronale predefinite și deja antrenate:

```
network.setup();
network.setConfidenceThreshold(0.2);
//doar obiectele care sunt detectate cu un nivel de încredere mai ridicat decât pragul,
// vor fi returnate la pasul următor
```

Pas5: returnarea obiectelor detectate în lista detections:

```
detections=network.run(imagine); // imaginea imagine trebuie sa fie de tip PImage
```

Pas6: parcurgerea tuturor obiectelor detectate:

```
for(ObjectDetectionResult detection : detections){
```

```
// Aici scrieți codul, apelând la metodele disponibile fiecărui element din lista detections
}
```

Metodele disponibile obiectelor din lista *detections* sunt următoarele:
detection.getClassId() - returnează id-ul clasei din care face parte obiectul
detection.getClassName() - numele clasei din care face parte obiectul
detection.getConfidence() - cat de desigur este că acel obiect face parte din categoria detectată
detection.getX() - poziția pe axa orizontală a coltului din stânga sus a cadrului obiectului
detection.getY() - poziția pe axa verticală a coltului din stânga sus a cadrului obiectului
detection.getWidth() - lățimea cadrului obiectului
detection.getHeight() -> înălțimea cadrului obiectului

Rețelele implementate în această librărie sunt următoarele:

A. Detecția obiectelor:
   - YOLOv3-tiny
   - YOLOv3-tiny-prn
   - EfficientNetB0-YOLOv3
   - YOLOv3 OpenImages Dataset
   - YOLOv3-spp (spatial pyramid pooling)
   - YOLOv3
   - YOLOv4
   - YOLOv4-tiny
   - YOLO Fastest & XL
   - SSDMobileNetV2
   - Handtracking based on SSDMobileNetV2
   - TextBoxes
   - Ultra-Light-Fast-Generic-Face-Detector-1MB RFB (~30 FPS on CPU)
   - Ultra-Light-Fast-Generic-Face-Detector-1MB Slim (~40 FPS on CPU)
   - Cascade Classifier
B. Object Segmentation
   - Mask R-CNN
C. Recunoașterea obiectelor:
   - Tesseract LSTM
D. Detecția punctelor cheie
   - Facial Landmark Detection
   - Single Human Pose Detection based on lightweight openpose
E. Clasificare:
   - MNIST CNN
   - FER+ Emotion
   - Age Net
   - Gender Net
F. Estimarea adâncimii (Depth):
   - MidasNet
G. Procesarea imaginilor:
   - Style Transfer
   - Multiple Networks for x2 x3 x4 Superresolution

2. Desfășurarea lucrării:

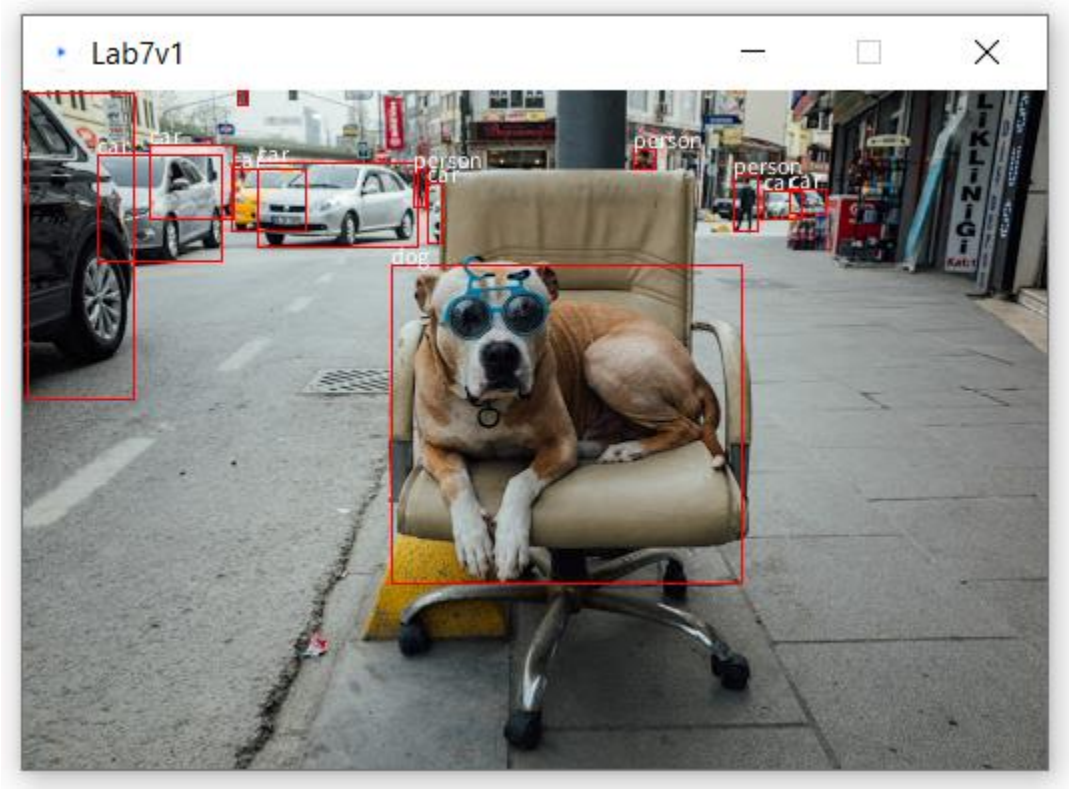
Deschideți în Processing IDE programul Lab7v1, al cărui cod este trecut mai jos:

```
import ch.bildspur.vision.*;
import ch.bildspur.vision.result.*;
import ch.bildspur.vision.network.*;

DeepVision vision;
YOLONetwork network;
ResultList<ObjectDetectionResult> detections;
PImage src;

void settings(){
 src=loadImage("imagine.jpg");
 size(src.width, src.height);
}

void setup(){
 vision=new DeepVision(this);
 network=vision.createYOLOv3();
 network.setup();
 network.setConfidenceThreshold(0.2);
}

void draw(){
 image(src,0,0);
 detections=network.run(src);
 for(ObjectDetectionResult detection : detections){
    int x=detection.getX();
    int y=detection.getY();
    int w=detection.getWidth();
    int h=detection.getHeight();
    String nume=detection.getClassName();
    noFill();
    stroke(255,0,0);
    rect(x,y,w,h);
    text(nume,x,y);
 }
 noLoop();
}
```

Cerințe :

1) **Analizați codul programului și explicați funcționarea lui folosind comentarii.**

2) **Descărcați 10 imagini stradale (care să conțină mai mulți oameni, mașini, obiecte, plante, semne de circulație, animale, etc.) și testați eficiența algoritmului pentru fiecare dintre categoriile enumerate.**



3) **Introduceți eficiențele în tabelul de mai jos, în dreptul liniei YOLOv3:**

Categorie	oameni	animale	semne	obiecte	mașini	plante	...
Eficiență YOLOv3 (%)							
Eficiență YOLOv4 (%)							

4) **Modificați linia** *network = vision.createYOLOv3();* **cu linia** *network = vision.createYOLOv4().* **Această schimbare va activa o versiune mai nouă a rețelei**

neuronale testate. Repetați măsurătorile pe cele 10 imagini testate anterior. Treceți noile rezultate în dreptul liniei YOLOv4.

5) **Comparați eficiențele obținute de voi cu cele obținute de ceilalți colegi. Cum explicați diferențele înregistrate?**

6) **Verificați și alte rețele neuronale implementate în librăria DeepVision. Exemple cu acestea sunt găsite în meniul File –> Examples –> Contributed Libraries -> DeepVision. Testați de exemplu eficiența *FaceAgeGenderWebcam* și *FaceAndEmotionWebcam* într-un mod asemănător cu felul în care ați lucrat pentru rețeaua *YOLONetwork*.**