**ImageAI : Videoobjektum-felismerés, -követés és –elemzés**

Az ImageAI kényelmes, rugalmas és hatékony módszereket kínál a videókon történő objektumfelismeréshez. A rendelkezésre bocsátott videóobjektum-felismerő osztály csak a RetinaNet, a YOLOv3 és a TinyYOLOv3 modelleket támogatja. Az ImageAI ezen verziója kereskedelmi szintű videoobjektum-felismerési funkciókat biztosít, amelyek többek között, de nem kizárólagosan eszköz/IP-kamera bemenetek, képkockánként, másodpercenként, percenként és teljes videóelemzésre terjednek ki az adatbázisokban és/vagy valós idejű vizualizációkban való tároláshoz és a jövőbeli meglátásokhoz.

Miután letöltötte az objektumfelismerő modellfájlt, másolja a modellfájlt a projekt mappájába, ahol a .py fájlok lesznek. Ezután hozzon létre egy python fájlt, és adjon neki egy nevet; egy példa erre a FirstVideoObjectDetection.py. Ezután írja be az alábbi kódot a python fájlba:

**FirstVideoObjectDetection.py**

from imageai.Detection import VideoObjectDetection

import os

execution\_path = os.getcwd()

detector = VideoObjectDetection()

detector.setModelTypeAsRetinaNet()

detector.setModelPath( os.path.join(execution\_path , "retinanet\_resnet50\_fpn\_coco-eeacb38b.pth"))

detector.loadModel()

video\_path = detector.detectObjectsFromVideo(input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_detected")

, frames\_per\_second=20, log\_progress=True)

print(video\_path)

Végezzük el a fent használt objektumfelismerő kód lebontását.

from imageai.Detection import VideoObjectDetection

import os

execution\_path = os.getcwd()

A fenti 3 sorban , az első sorban importáljuk az \*\*ImageAI video object detection\*\* osztályt, a második sorban importáljuk az os-t, és megkaptuk a mappa elérési útját, ahol a python fájlunk fut.

detector = VideoObjectDetection()

detector.setModelTypeAsRetinaNet()

detector.setModelPath( os.path.join(execution\_path , "retinanet\_resnet50\_fpn\_coco-eeacb38b.pth"))

detector.loadModel()

A fenti 4 sorban az első sorban létrehoztuk a VideoObjectDetection osztály új példányát, a második sorban a modell típusát RetinaNet-re állítottuk, a harmadik sorban a modell elérési útját a letöltött és a python fájlmappába másolt RetinaNet modellfájlra állítottuk, a negyedik sorban pedig betöltöttük a modellt.

video\_path = detector.detectObjectsFromVideo(input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_detected"),

frames\_per\_second=20, log\_progress=True)

print(video\_path)

A fenti 2 sorban futtattuk a detectObjectsFromVideo() függvényt, és elemezzük a videó elérési útvonalát,az új videó elérési útvonalát (kiterjesztés nélkül, alapértelmezés szerint .avi videót ment), amelyet a függvény el fog menteni, a másodpercenkénti képkockák számát (fps), amelyet a kimeneti videónak kell lennie, és a konzolban a felismerés előrehaladásának naplózását. Ezután a függvény visszaadja a mentett videó elérési útvonalát, amely tartalmazza a videóban észlelt objektumok dobozait és százalékos valószínűségét.

Egyéni videoobjektum-érzékelés

Az ImageAI által támogatott videoobjektum-felismerő modell (RetinaNet) 80 különböző típusú objektumot képes felismerni. Ezek közé tartoznak:

person, bicycle, car, motorcycle, airplane, bus, train, truck, boat, traffic light, fire hydrant, stop\_sign,

parking meter, bench, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra,

giraffe, backpack, umbrella, handbag, tie, suitcase, frisbee, skis, snowboard,

sports ball, kite, baseball bat, baseball glove, skateboard, surfboard, tennis racket,

bottle, wine glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange,

broccoli, carrot, hot dog, pizza, donot, cake, chair, couch, potted plant, bed,

dining table, toilet, tv, laptop, mouse, remote, keyboard, cell phone, microwave,

oven, toaster, sink, refrigerator, book, clock, vase, scissors, teddy bear, hair dryer,

toothbrush.

Érdekes, hogy az ImageAI lehetővé teszi, hogy a fenti elemek közül egy vagy több felismerést végezzen. Ez azt jelenti, hogy testre szabhatja, hogy milyen típusú tárgy(ak)at szeretne észlelni a videóban. Nézzük meg az alábbi kódot:

from imageai.Detection import VideoObjectDetection

import os

execution\_path = os.getcwd()

detector = VideoObjectDetection()

detector.setModelTypeAsRetinaNet()

detector.setModelPath( os.path.join(execution\_path , "retinanet\_resnet50\_fpn\_coco-eeacb38b.pth"))

detector.loadModel()

custom\_objects = detector.CustomObjects(person=True, bicycle=True, motorcycle=True)

video\_path = detector.detectCustomObjectsFromVideo(

custom\_objects=custom\_objects,

input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_custom\_detected"),

frames\_per\_second=20, log\_progress=True)

print(video\_path)

Nézzük meg a kódnak azt a részét, amely ezt lehetővé tette.

custom\_objects = detector.CustomObjects(person=True, bicycle=True, motorcycle=True)

video\_path = detector.detectCustomObjectsFromVideo(

custom\_objects=custom\_objects,

input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_custom\_detected"),

frames\_per\_second=20, log\_progress=True)

A fenti kódban a modell betöltése után (a modell betöltése előtt is megtehetjük) definiáltunk egy új változót custom\_objects = detector.CustomObjects(), amelyben a person, car és motorcycle tulajdonságait True értékre állítjuk. Ezzel azt mondjuk a modellnek, hogy csak az általunk True értékre beállított objektumokat érzékelje. Ezután meghívjuk a detector.detectCustomObjectsFromVideo() függvényt, amely lehetővé teszi számunkra az egyéni objektumok felismerését. Ezután a custom\_objects értéket az általunk definiált custom objects változóra állítjuk.

Kamera / Live Stream videó észlelése

Az ImageAI mostantól lehetővé teszi az élő videó észlelését a kamerabemenetek támogatásával. Az OpenCV VideoCapture() függvényének használatával betöltheti az élő videostreameket egy eszközkamerából, kábellel csatlakoztatott kamerákból vagy IP-kamerákból, és elemezheti azokat az ImageAI detectObjectsFromVideo() és detectCustomObjectsFromVideo() függvényeibe. A videofájlban lévő objektumok észleléséhez támogatott összes funkció elérhető a kamera élő videós adatfolyamában lévő objektumok észleléséhez is. Az alábbiakban talál egy példát az eszköz kamerájából származó élő videókapcsolat észlelésére.

from imageai.Detection import VideoObjectDetection

import os

import cv2

execution\_path = os.getcwd()

camera = cv2.VideoCapture(0)

detector = VideoObjectDetection()

detector.setModelTypeAsRetinaNet()

detector.setModelPath(os.path.join(execution\_path , "retinanet\_resnet50\_fpn\_coco-eeacb38b.pth"))

detector.loadModel()

video\_path = detector.detectObjectsFromVideo(

camera\_input=camera,

output\_file\_path=os.path.join(execution\_path, "camera\_detected\_video"),

frames\_per\_second=20, log\_progress=True, minimum\_percentage\_probability=40)

A különbség a fenti kód és a videófájl felismerésének kódja között az, hogy definiáltunk egy OpenCV VideoCapture példányt, és betöltöttük az alapértelmezett eszközkamerát. Ezután a meghatározott kamerát elemeztük a camera\_input paraméterbe, amely a videófájlhoz használt input\_file\_path-ot helyettesíti.

Videó elemzés

Az ImageAI mostantól kereskedelmi szintű videóelemzést kínál a Video Object Detection osztályban, mind a videófájlok, mind a kamerák bemeneteihez. Ez a funkció lehetővé teszi a fejlesztők számára, hogy mély betekintést nyerjenek az ImageAI-val feldolgozott videókba. Ezek a meglátások valós időben megjeleníthetők, és egy NoSQL adatbázisban tárolhatók a későbbi felülvizsgálat vagy elemzés céljából.

A videóelemzéshez a detectObjectsFromVideo() és detectCustomObjectsFromVideo() mostantól lehetővé teszi, hogy megadhassa a saját definiált függvényeit, amelyek az észlelt videó minden egyes képkockájára, másodpercére és/vagy percére végrehajtásra kerülnek, valamint egy olyan függvényt, amely a videó észlelés végén kerül végrehajtásra. Miután ezeket a függvényeket megadták, nyers, de átfogó elemzési adatokat kapnak a képkocka/másodperc/perc indexéről, az észlelt objektumokról (név, percentage\_probability és box\_points), az egyes észlelt egyedi objektumok példányainak számáról és az egyes észlelt egyedi objektumok átlagos előfordulási számáról egy másodperc/perc és a teljes videó során.

A videóelemzéshez mindössze annyit kell tennie, hogy megadja a függvényt, megadja a megfelelő paramétereket, amelyeket kapni fog, és a függvény nevét a per\_frame\_function, per\_second\_function, per\_minute\_function és video\_complete\_function paraméterekbe elemzi az észlelési függvényben. Az alábbiakban példákat talál a videóelemző függvényekre.

def forFrame(frame\_number, output\_array, output\_count):

print("FOR FRAME " , frame\_number)

print("Output for each object : ", output\_array)

print("Output count for unique objects : ", output\_count)

print("------------END OF A FRAME --------------")

def forSeconds(second\_number, output\_arrays, count\_arrays, average\_output\_count):

print("SECOND : ", second\_number)

print("Array for the outputs of each frame ", output\_arrays)

print("Array for output count for unique objects in each frame : ", count\_arrays)

print("Output average count for unique objects in the last second: ", average\_output\_count)

print("------------END OF A SECOND --------------")

def forMinute(minute\_number, output\_arrays, count\_arrays, average\_output\_count):

print("MINUTE : ", minute\_number)

print("Array for the outputs of each frame ", output\_arrays)

print("Array for output count for unique objects in each frame : ", count\_arrays)

print("Output average count for unique objects in the last minute: ", average\_output\_count)

print("------------END OF A MINUTE --------------")

video\_detector = VideoObjectDetection()

video\_detector.setModelTypeAsYOLOv3()

video\_detector.setModelPath(os.path.join(execution\_path, "yolov3.pt"))

video\_detector.loadModel()

video\_detector.detectObjectsFromVideo(

input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_detected"),

frames\_per\_second=10,

per\_second\_function=forSeconds,

per\_frame\_function=forFrame,

per\_minute\_function=forMinute,

minimum\_percentage\_probability=30

)

Minden olyan függvény esetében, amelyet a per\_frame\_functionba elemezel, a függvény minden egyes videokép feldolgozása után végrehajtásra és elemzésre kerül.

Frame Index: Ez a képkocka pozíciójának száma a videón belül (pl. 1 az első képkocka és 20 a huszadik képkocka).

Kimeneti tömb: Ez egy szótárakból álló tömb. Minden szótár megfelel a képen található minden egyes felismert objektumnak, és tartalmazza az objektum "name", "percentage\_probabaility" és "box\_points"(x1,y1,x2,y2) értékeit.

Kimeneti szám: Ez egy szótár, amelynek kulcsa minden egyes felismert objektum neve, értékei pedig a felismert objektumok példányainak száma.

A fenti eredményben a videó feldolgozása és mentése 10 képkocka/másodperc (FPS) sebességgel történt. A per\_second\_functionba elemzett bármelyik függvény esetében a függvény a feldolgozott videó minden egyes másodperce után végrehajtásra és elemzésre kerül.

Második index: Ez a másodperc pozíciójának száma a videón belül (pl. 1 az első másodperchez és 20 a huszadik másodperchez).

Kimeneti tömb: A fenti példában 10 tömb van, amely megfelel az egy másodpercben található 10 képkockának). Minden egyes tartalmazott tömb szótárakat tartalmaz. Minden szótár megfelel a képen található minden egyes felismert objektumnak, és tartalmazza az objektum "neve", "százalékos valószínűsége" és "box\_points" (x1,y1,x2,y2) értékeit.

Számtáblák: Ez egy szótárakból álló tömb. Minden szótár és annak pozíciója (tömbindex + 1) megfelel a videó utolsó másodpercének megfelelő képkockának. Minden egyes szótár kulcsként tartalmazza minden egyes észlelt egyedi objektum nevét, értékként pedig az észlelt objektumok példányainak számát.

Átlagos kimeneti szám: Ez egy olyan szótár, amelynek kulcsai az utolsó másodpercben észlelt minden egyes egyedi objektum neve, értékei pedig az észlelt objektumok példányainak átlagos száma a képkockák számában.

A perc funkció eredményei A fenti 4 paraméter, amelyek a feldolgozott videó minden másodpercére visszakerülnek, ugyanazok a paraméterek, amelyek a feldolgozott videó minden percére visszakerülnek. A különbség az, hogy a visszaadott index megfelel a percindexnek, az output\_arrays egy tömb, amely az FPS \* 60 tömb számát tartalmazza (a fenti kódpéldában 10 képkocka másodpercenként(fps) \* 60 másodperc = 600 képkocka = 600 tömb), a count\_arrays pedig egy olyan tömb, amely az FPS \* 60 szótárak számát tartalmazza (a fenti kódpéldában 10 képkocka másodpercenként(fps) \* 60 másodperc = 600 képkocka = 600 szótár), az average\_output\_count pedig egy olyan szótár, amely az elmúlt percben található összes képkockában észlelt összes objektumot tartalmazza.

A Video Complete funkció eredményei Az ImageAI lehetővé teszi, hogy a teljes feldolgozott videó teljes elemzését megkapja. Mindössze egy olyan függvényt kell definiálnia, mint a forSecond vagy forMinute függvény, és a video\_complete\_function paramétert kell beállítania a .detectObjectsFromVideo() vagy .detectCustomObjectsFromVideo() függvénybe. A per\_second-funkció és a per\_minute\_funkció ugyanazokat az értékeket fogja visszaadni. A különbség az, hogy a funkció nem ad vissza indexet, hanem a másik 3 értéket adja vissza, és a 3 érték a videó összes képkockájára kiterjed. Az alábbiakban egy mintafunkciót mutatunk be:

def forFull(output\_arrays, count\_arrays, average\_output\_count):

#Perform action on the 3 parameters returned into the function

video\_detector.detectObjectsFromVideo(

input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_detected"),

frames\_per\_second=10,

video\_complete\_function=forFull,

minimum\_percentage\_probability=30

)

VÉGREHAJTÁS A VIDEÓANALÍZISRÓL : Az ImageAI lehetővé teszi, hogy az észlelt videóképkockát Numpy tömbként kapja meg minden egyes képkocka, másodperc és perc funkcióban. Mindössze egy további paramétert kell megadni a függvényben, és a return\_detected\_frame=True értéket kell beállítani a detectObjectsFromVideo() vagy detectCustomObjectsFrom() függvényben. Ha ez be van állítva, akkor a függvényedben külön megadott extra paraméter lesz a detektált képkocka Numpy tömbje. Lásd az alábbi mintát:

def forFrame(frame\_number, output\_array, output\_count, detected\_frame):

print("FOR FRAME " , frame\_number)

print("Output for each object : ", output\_array)

print("Output count for unique objects : ", output\_count)

print("Returned Objects is : ", type(detected\_frame))

print("------------END OF A FRAME --------------")

video\_detector.detectObjectsFromVideo(

input\_file\_path=os.path.join(execution\_path, "traffic.mp4"),

output\_file\_path=os.path.join(execution\_path, "traffic\_detected"),

frames\_per\_second=10,

per\_frame\_function=forFrame,

minimum\_percentage\_probability=30,

return\_detected\_frame=True

)

Képkocka észlelési intervallumok

A fenti videoobjektum-felismerési feladatot a képkocka valós idejű objektumfelismerésre optimalizálták, ami biztosítja, hogy a videó minden egyes képkockájában felismerjük az objektumokat. Az ImageAI lehetőséget biztosít a videóképkocka észlelés beállítására, ami felgyorsíthatja a videó észlelési folyamatot. A .detectObjectsFromVideo() vagy .detectCustomObjectsFromVideo() meghívásakor megadhatja, hogy a detektálást milyen képkocka intervallumban végezze el. A frame\_detection\_interval paraméter 5 vagy 20 értékűre állításával ez azt jelenti, hogy a videóban lévő objektumdetektálások 5 vagy 20 képkocka után frissülnek. Ha a kimeneti videó frames\_per\_second értéke 20, ez azt jelenti, hogy a videóban lévő objektumdetektálások negyed másodpercenként vagy másodpercenként egyszer frissülnek. Ez olyan esetekben hasznos, amikor a rendelkezésre álló számítási kapacitás kisebb, és a mozgó objektumok sebessége alacsony. Ez biztosítja, hogy az objektumok észlelése másodperces valós idejű , fél másodperces valós idejű vagy az Ön igényeinek megfelelő módon történjen. A videoobjektum-érzékelést ugyanazon a bemeneti videón végeztük el, amelyet mindvégig használtunk, 5 értékű frame\_detection\_interval értéket alkalmazva.

###Videó észlelési időkorlát

Az ImageAI mostantól lehetővé teszi, hogy másodpercben megadjon egy időkorlátot a videókban vagy a kamera élő adásában lévő objektumok észlelésére. A videóérzékelési kódodban az időkorlát beállításához csak annyit kell tenned, hogy a detectObjectsFromVideo() függvényben a detection\_timeout paramétert a kívánt másodpercek számára állítod be. Az alábbi példakódban a detection\_timeout paramétert 120 másodpercre (2 perc) állítjuk be.

from imageai.Detection import VideoObjectDetection

import os

import cv2

execution\_path = os.getcwd()

camera = cv2.VideoCapture(0)

detector = VideoObjectDetection()

detector.setModelTypeAsRetinaNet()

detector.setModelPath(os.path.join(execution\_path , "retinanet\_resnet50\_fpn\_coco-eeacb38b.pth"))

detector.loadModel()

video\_path = detector.detectObjectsFromVideo(camera\_input=camera,

output\_file\_path=os.path.join(execution\_path, "camera\_detected\_video"),

frames\_per\_second=20,

log\_progress=True,

minimum\_percentage\_probability=40,

detection\_timeout=120)

Forrás:

Moses Olafenwa - ImageAI : Video Object Detection, Tracking and Analysis (https://github.com/OlafenwaMoses/ImageAI/blob/master/imageai/Detection/VIDEO.md)