Ivan Flores Martinez AI and Deep Learning

Midterm exam

Step 1 – Get data to train the model.

I downloaded the Pascal VOC 2012.v3-416x416.coco dataset from roboflow. Then, I unzipped it and checked the files. The dataset contains 13690 images in COCO format for 21 categories of objects (i.e. person, airplane, chair, etc.).

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Step 2 – Register the dataset.

I registered the dataset with coco format for use with detectron

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Step 3 – Train the model.

I trained the model for 20 epochs using the faster RCNN model from detectron. This model is used for object detection in images. The hyperparameters to run the model are described in the python script (see train\_detectron\_VOC.ipynb).

Texto

Descripción generada automáticamente

Step 4 – Test the model

I tested the model on a few images to make sure it was working. The model performed well on the test data.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Step 5 – Record video from webcam

I recorded a video from my webcam using the VideoCapture() function from opencv. The video was recorded for 47 seconds with some volunteers in my office. The video is available in the videos folder of the midterm repository. The name of the file is video\_raw.mp4

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Step 6 – Use trained model to detect object in video

I passed the recorded video through the detectron\_VOC model to detect objects. First, I created a predictor with the weights of my detectronVOC trained model. Then, I initialize the visualizer, using the VideoVisualizer function.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Later, I created a function that passed the video frames (1412) to the predictor and stored the output in the file “video\_obj\_det.mp4” (see “videos” folder in repository). I made sure that all frames had color and that all frames were processed by the detectron model.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Step 7 – Display original video and the video with the predictions simultaneously

I used open-cv to show the original video and the video with the predictions simultaneously. To do this, I had to create an object with a different name for each video. Then, I had to resize the windows to display the videos side by side in the screen. Finally, I made sure the videos were displayed in color and that I could stop the streaming with the keyboard.

Texto

Descripción generada automáticamente

This is how the videos look side by side.

Imagen que contiene interior, hombre, espejo, parado

Descripción generada automáticamente

**Bonus**

I also trained a model for instance segmentation. You can find it in the “Videos” folder of the repository. The file name is “video\_inst\_segmention.mp4”. Here is a sample of the video using the same procedure described above.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

See next page for implementation of the model in Colab.

Training detectron in COLAB

I included a jupyter notebook that runs the model in Colab. You can find it in the “jupyter notebook” folder with the name: train\_detectron\_in\_Colab\_+webcam.ipynb. The notebook includes a javascript helper function that is used to record a video with a webcam in Colab.

Texto, Carta

Descripción generada automáticamente

To start recording, I clicked on the start recording button and then saved all frames into a mp4 file.

Captura de pantalla de un celular

Descripción generada automáticamente

The recorded video was passed through the detectron model and the output was stored in the file named “video\_obj\_det.mp4. Then, I used a modified version of the imshow function (cv2\_imshow) created for COLAB to display the video frames with the detected objects. The frames are displayed below each other with the detections (the model can only detect persons and chairs in an office setting).

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente

This was the closer I could get to completing the task.