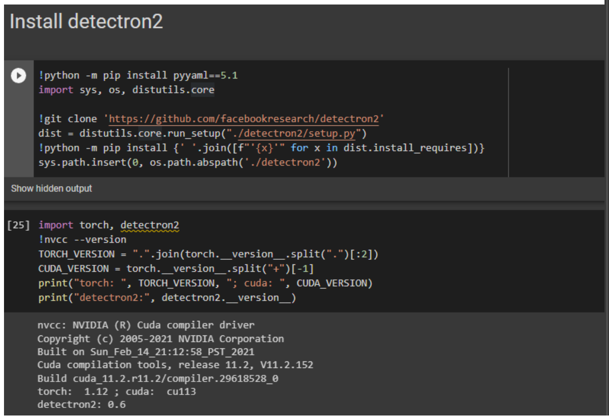
Johnathan Hager

11/9/2022

Applied Ai Midterm

This project was semi completed using google colab. Since it was unable to be ran on 3 different local machines I have tried. You can review the issues in the "Issues running locally" section at the end. This implementation used a pre-trained model. The below on the left, shows how we install detectron2 and initialize it. Then we check the versions and set the paths.



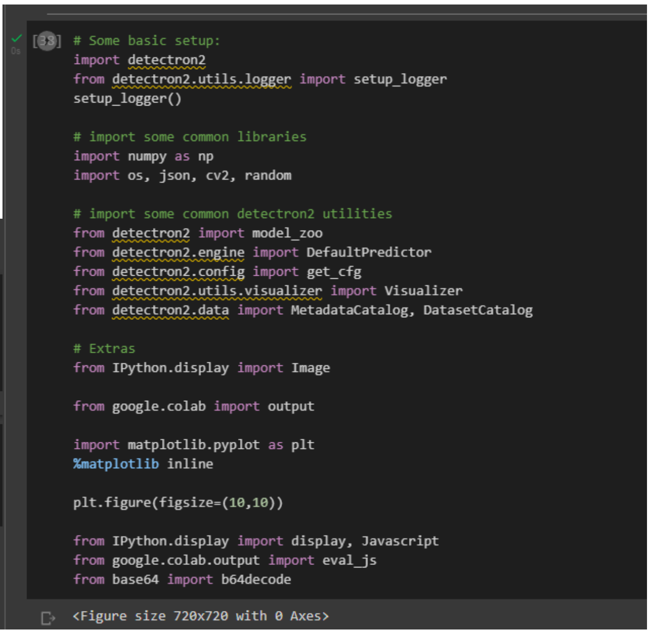
Install detectron2

```
!python -m pip install pyyaml==5.1
import sys, os, distutils.core

!git clone 'https://github.com/facebookresearch/detectron2'
dist = distutils.core.run_setup("./detectron2/setup.py")
!python -m pip install {' '.join([f"'{x}'" for x in dist.install_requires])}
sys.path.insert(0, os.path.abspath('./detectron2'))
```

Show hidden output

```
[25] import torch, detectron2
     !nvcc --version
     TORCH_VERSION = ".".join(torch.__version__.split(".")[:2])
     CUDA_VERSION = torch.__version__.split("+")[-1]
     print("torch: ", TORCH_VERSION, "; cuda: ", CUDA_VERSION)
     print("detectron2:", detectron2.__version__)
```

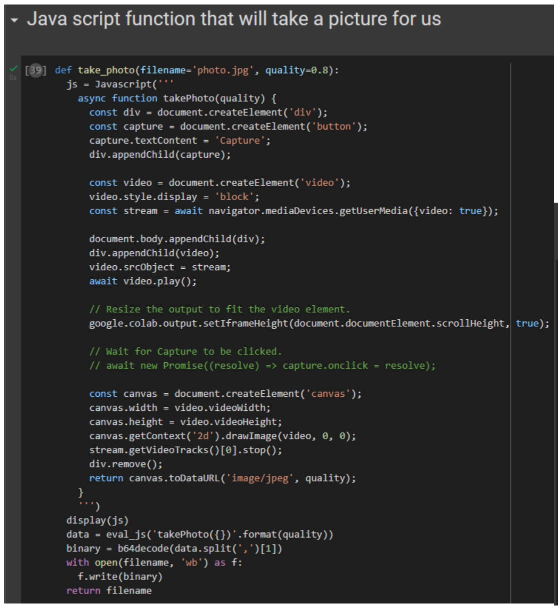
```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2021 NVIDIA Corporation
Built on Sun_Feb_14_21:12:58_PST_2021
Cuda compilation tools, release 11.2, V11.2.152
Build cuda_11.2.r11.2/compiler.29618528_0
torch:  1.12 ; cuda:  cu113
detectron2: 0.6
```



```
[33] # Some basic setup:
     import detectron2
     from detectron2.utils.logger import setup_logger
     setup_logger()

     # import some common libraries
     import numpy as np
     import os, json, cv2, random

     # import some common detectron2 utilities
     from detectron2 import model_zoo
     from detectron2.engine import DefaultPredictor
     from detectron2.config import get_cfg
     from detectron2.utils.visualizer import Visualizer
     from detectron2.data import MetadataCatalog, DatasetCatalog

     # Extras
     from IPython.display import Image

     from google.colab import output

     import matplotlib.pyplot as plt
     %matplotlib inline

     plt.figure(figsize=(10,10))

     from IPython.display import display, Javascript
     from google.colab.output import eval_js
     from base64 import b64decode
```

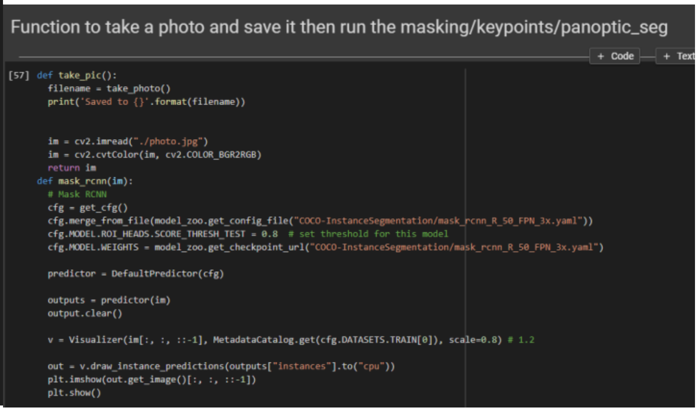
```
<Figure size 720x720 with 0 Axes>
```

Above on the right, we import libraries to be use and set up matplotlib. Below on the left, we must use a Java script to allow us to stream our webcam to google colab and take a picture.



Java script function that will take a picture for us

```
[39] def take_photo(filename='photo.jpg', quality=0.8):
       js = Javascript('''
         async function takePhoto(quality) {
           const div = document.createElement('div');
           const capture = document.createElement('button');
           capture.textContent = 'Capture';
           div.appendChild(capture);

           const video = document.createElement('video');
           video.style.display = 'block';
           const stream = await navigator.mediaDevices.getUserMedia({video: true});

           document.body.appendChild(div);
           div.appendChild(video);
           video.srcObject = stream;
           await video.play();

           // Resize the output to fit the video element.
           google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);

           // Wait for Capture to be clicked.
           // await new Promise((resolve) => capture.onclick = resolve);

           const canvas = document.createElement('canvas');
           canvas.width = video.videoWidth;
           canvas.height = video.videoHeight;
           canvas.getContext('2d').drawImage(video, 0, 0);
           stream.getVideoTracks()[0].stop();
           div.remove();
           return canvas.toDataURL('image/jpeg', quality);
         }
         ''')
       display(js)
       data = eval_js('takePhoto({})'.format(quality))
       binary = b64decode(data.split(',')[1])
       with open(filename, 'wb') as f:
         f.write(binary)
       return filename
```



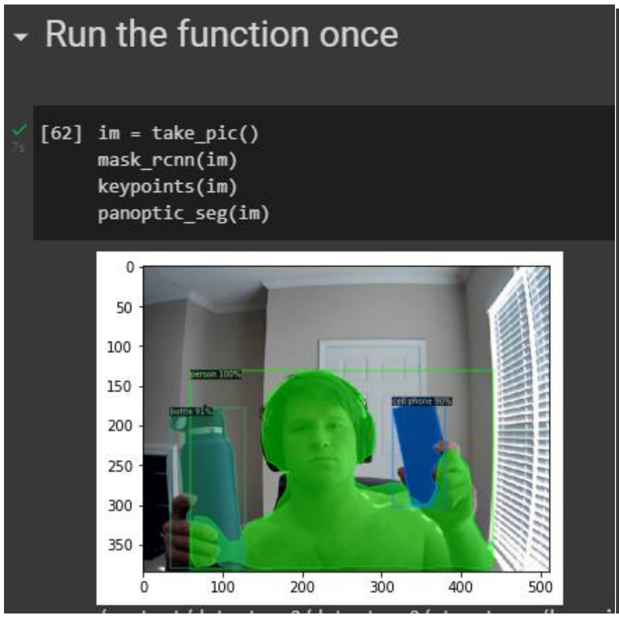
Function to take a photo and save it then run the masking/keypoints/panoptic_seg

```
[57] def take_pic():
       filename = take_photo()
       print('Saved to {}'.format(filename))


       im = cv2.imread("./photo.jpg")
       im = cv2.cvtColor(im, cv2.COLOR_BGR2RGB)
       return im
     def mask_rcnn(im):
       # Mask RCNN
       cfg = get_cfg()
       cfg.merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"))
       cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.8  # set threshold for this model
       cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")

       predictor = DefaultPredictor(cfg)

       outputs = predictor(im)
       output.clear()

       v = Visualizer(im[:, :, ::-1], MetadataCatalog.get(cfg.DATASETS.TRAIN[0]), scale=0.8) # 1.2

       out = v.draw_instance_predictions(outputs["instances"].to("cpu"))
       plt.imshow(out.get_image()[:, :, ::-1])
       plt.show()
```

Here is where the definitions are for our picture function which will use the java function. This saves the file in jpg format then converts it to RGB format for use in plotting.

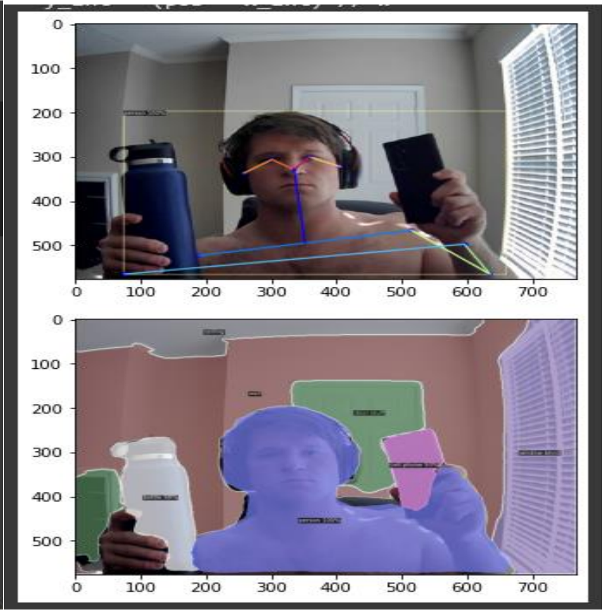
We also have our Mask RCNN function which does all the masking of the image taken and predictions.

This also plots the function on a graph.

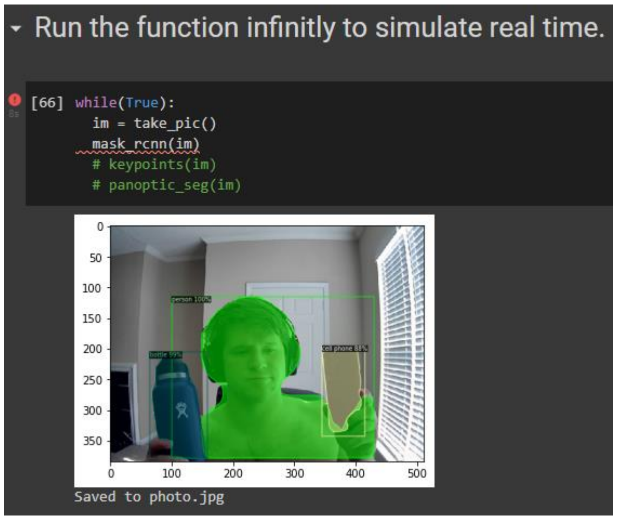


Run the function once

```
[62] im = take_pic()
     mask_rcnn(im)
     keypoints(im)
     panoptic_seg(im)
```



Above is running the functions one time and the results. First is Mask RCNN, second is key points, and third is panoptic segmentation.



Run the function infinitly to simulate real time.

```
[66] while(True):
       im = take_pic()
       mask_rcnn(im)
       # keypoints(im)
       # panoptic_seg(im)
```

```
Saved to photo.jpg
```

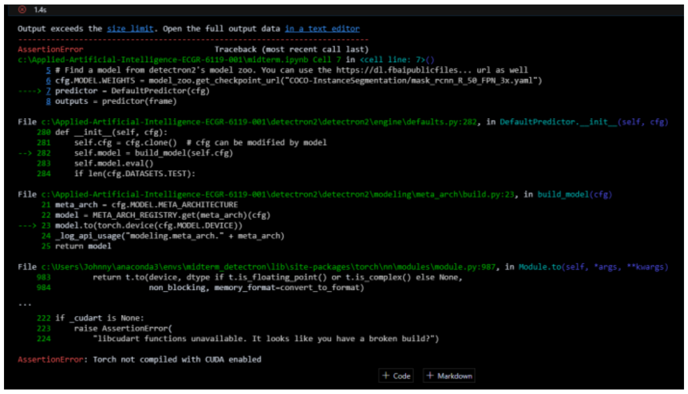
Now we run the function in a loop to help simulate running this real-time. Google colab runs to slow to get a real-time implementation working.

Issues running locally:

There was a TON of issues I had when trying to install and set up this project on my local machine.

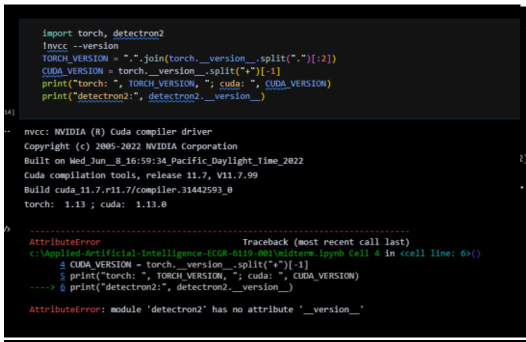
Two of the major issues were: installing detectron and Torch enabled with CUDA

After many days of uninstalling and reinstalling I was able to get torch and CUDA installed together.

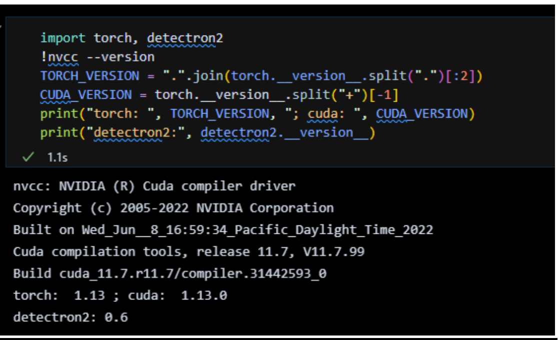


```
1.4s

Output exceeds the size limit. Open the full output data in a text editor
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
c:\Applied-Artificial-Intelligence-ECGR-6119-001\midterm.ipynb Cell 7 in <cell line: 7>()
      5 # Find a model from detectron2's model zoo. You can use the https://dl.fbaipublicfiles... url as well
      6 cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml")
----> 7 predictor = DefaultPredictor(cfg)
      8 outputs = predictor(frame)

File c:\Applied-Artificial-Intelligence-ECGR-6119-001\detectron2\detectron2\engine\defaults.py:282, in DefaultPredictor.__init__(self, cfg)
    280 def __init__(self, cfg):
    281     self.cfg = cfg.clone()  # cfg can be modified by model
--> 282     self.model = build_model(self.cfg)
    283     self.model.eval()
    284     if len(cfg.DATASETS.TEST):

File c:\Applied-Artificial-Intelligence-ECGR-6119-001\detectron2\detectron2\modeling\meta_arch\build.py:23, in build_model(cfg)
     21 meta_arch = cfg.MODEL.META_ARCHITECTURE
     22 model = META_ARCH_REGISTRY.get(meta_arch)(cfg)
---> 23 model.to(torch.device(cfg.MODEL.DEVICE))
     24 _log_api_usage("modeling.meta_arch." + meta_arch)
     25 return model

File c:\Users\Johnny\anaconda3\envs\midterm_detectron\lib\site-packages\torch\nn\modules\module.py:987, in Module.to(self, *args, **kwargs)
    983         return t.to(device, dtype if t.is_floating_point() or t.is_complex() else None,
    984                     non_blocking, memory_format=convert_to_format)

...
    222 if _cudart is None:
    223     raise AssertionError(
    224         "libcudart functions unavailable. It looks like you have a broken build?")

AssertionError: Torch not compiled with CUDA enabled
```

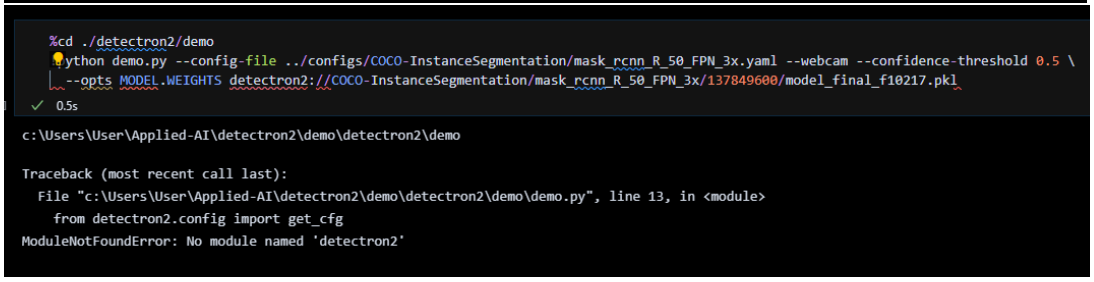
At one point I was able to what I thought was get detectron2 installed in which one of the photos below show saying that "detectron2: 0.6" which I thought would mean that it was installed. But then later when I tried to import detectron2 to use it, I would get an error saying "ModuleNotFoundError: No module named 'detectron2'".



```
import torch, detectron2
!nvcc --version
TORCH_VERSION = ".".join(torch.__version__.split(".")[:2])
CUDA_VERSION = torch.__version__.split("+")[-1]
print("torch: ", TORCH_VERSION, "; cuda: ", CUDA_VERSION)
print("detectron2:", detectron2.__version__)

nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2022 NVIDIA Corporation
Built on Wed_Jun__8_16:59:34_Pacific_Daylight_Time_2022
Cuda compilation tools, release 11.7, V11.7.99
Build cuda_11.7.r11.7/compiler.31442593_0
torch:  1.13 ; cuda:  1.13.0

---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
c:\Applied-Artificial-Intelligence-ECGR-6119-001\midterm.ipynb Cell 4 in <cell line: 6>()
      4 CUDA_VERSION = torch.__version__.split("+")[-1]
      5 print("torch: ", TORCH_VERSION, "; cuda: ", CUDA_VERSION)
----> 6 print("detectron2:", detectron2.__version__)

AttributeError: module 'detectron2' has no attribute '__version__'
```



```
import torch, detectron2
!nvcc --version
TORCH_VERSION = ".".join(torch.__version__.split(".")[:2])
CUDA_VERSION = torch.__version__.split("+")[-1]
print("torch: ", TORCH_VERSION, "; cuda: ", CUDA_VERSION)
print("detectron2:", detectron2.__version__)
✓ 1.1s

nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2022 NVIDIA Corporation
Built on Wed_Jun__8_16:59:34_Pacific_Daylight_Time_2022
Cuda compilation tools, release 11.7, V11.7.99
Build cuda_11.7.r11.7/compiler.31442593_0
torch:  1.13 ; cuda:  1.13.0
detectron2: 0.6
```



```
%cd ./detectron2/demo
ython demo.py --config-file ../configs/COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml --webcam --confidence-threshold 0.5 \
 --opts MODEL.WEIGHTS detectron2://COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x/137849600/model_final_f10217.pkl
✓ 0.5s

c:\Users\User\Applied-AI\detectron2\demo\detectron2\demo

Traceback (most recent call last):
  File "c:\Users\User\Applied-AI\detectron2\demo\detectron2\demo\demo.py", line 13, in <module>
    from detectron2.config import get_cfg
ModuleNotFoundError: No module named 'detectron2'
```

I went through the process of installing and uninstall detectron2, pytorch, CUDA, python. And tried to create environments in anaconda, tried jupyter notebooks, tired with VS code.

The only thing I found which may have fixed my issues were to install it on a Linux machine, which of the 3 computers I have access to they are all on windows.

I also tried install a VM on my desktop machine to run Linux. This ended up cause more issues.

So, in the end I just decided to stick with google colab. Its not able to fast enough to allow a real-time implementation, but I was able to set it up, so it takes pictures on a loop runs the masking and outputs the image on a matplotlib.