TopoFeatures Overview

Background:

The goal of this project is to implement a machine learning model that can detect and recognize the outlines of various types of landform features (summits, valleys, islands, etc.) autonomously from a given set of images. The images we are using to set up and test the workflow are from the Historical Topographic Map Collection (HTMC). We extend Facebook's detectron2 (https://github.com/facebookresearch/detectron2) architecture to make the training and evaluation process easier. This comes with some setup, which is described below.

Installation:

Install instructions and download links:
https://detectron2.readthedocs.io/en/latest/tutorials/install.html

Tutorial: https://colab.research.google.com/drive/16jcaJoc6bCFAQ96jDe2HwtXj7BMD_-m5

Blog post: https://ai.facebook.com/blog/-detectron2-a-pytorch-based-modular-object-detection-library-

COCO format (official): https://cocodataset.org/#format-data

To install and setup the TopoFeatures project within your home directory on Serenity:

1. Setup a new conda environment so we can keep downloaded packages contained.
   ```
   $conda create -n TopoFeatures
   $conda activate TopoFeatures
   ```
2. Install python=3.7, pytorch=1.10.2, torchvision and cudatoolkit=10.2 packages into the conda env:
   ```
   $conda install python=3.7 pytorch torchvision
   cudatoolkit=10.2 -c pytorch -c conda-forge
   ```
3. Install other needed packages:
   ```
   $conda install gdal -c conda-forge
   ```
4. Run the command below to install the detectron2 code base as a Python package for CUDA 10.2 and torch 1.10:
   ```
   $python -m pip install detectron2 -f
   https://dl.fbaipublicfiles.com/detectron2/wheels/cu102
   /torch1.10/index.html --trusted-host
   dl.fbaipublicfiles.com
   ```
   Now we can import detectron2 code into Python scripts for use.
5. Clone detectron2 from github:
   ```
   $ git clone
   https://github.com/facebookresearch/detectron2
   ```

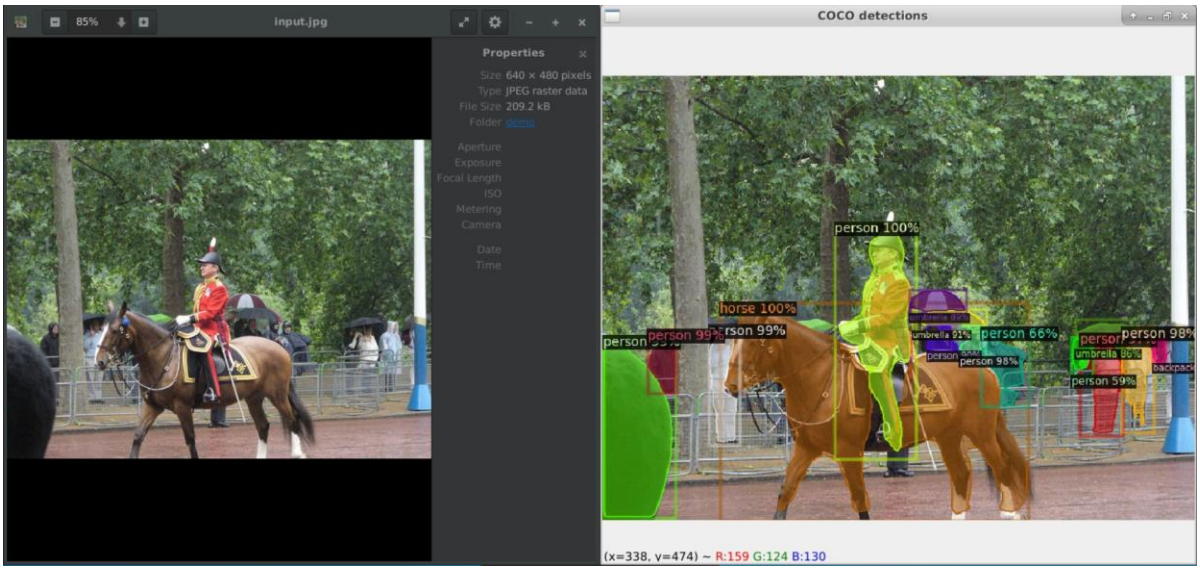
Current Progress:

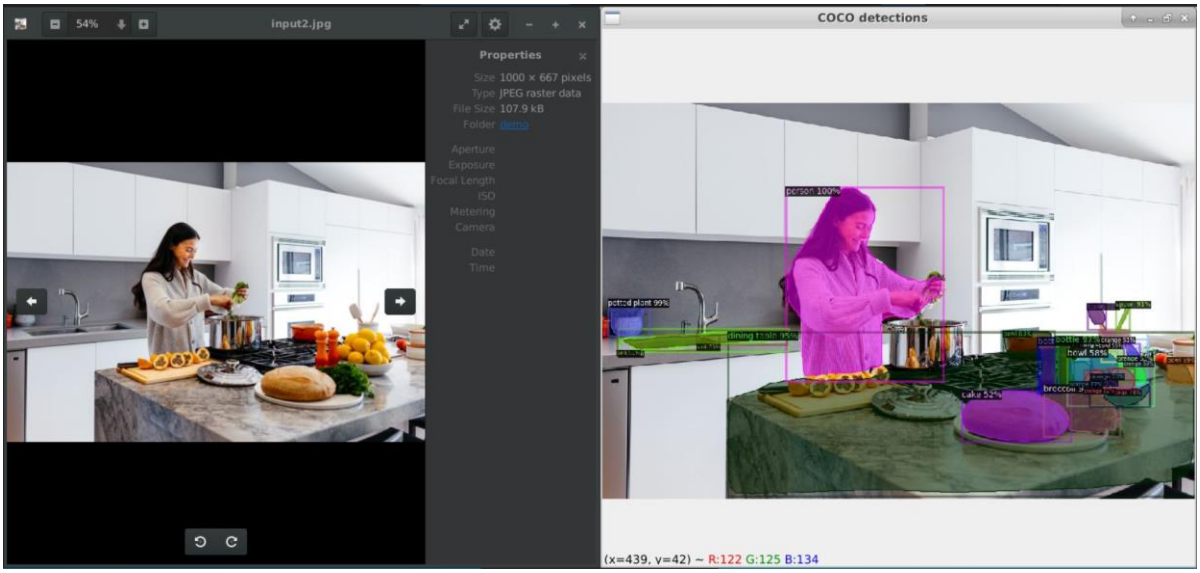
Setup:

I've been able to get the detectron2 architecture setup and run a demo script. To run demo script with a benchmark mask_rcnn model and given input "input.jpg":

```
$cd detectron2/demo/
```

```
$python demo.py --config-file ../configs/COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml --input input.jpg --opts MODEL.WEIGHTS detectron2://COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x/137849600/model_final_f10217.pkl
```





Custom Data Creation:

Started getting labels for individual images according to COCO format, then generalized to every downloaded image in the dataset. Labels are formatted in the "TF_label_maker.py" script, which is called by and gets the necessary information from "TF_create_dataset.py". Then "TF_create_dataset.py" proceeds to save the label into a text file in JSON string format and append the information to the COCO dataset dictionary "images" field, which contains the labels for all the dataset images. After that, it runs the loops that downloads the corresponding image from the HTMC service, then finally plots the segmentation and bbox and saves that in the "plot" folder.

Created new script ("TF_register_dataset.py") to register the dataset in detectron2 so it can be accessed by the model. The detectron2 format is slightly different than the COCO format, but they have a built-in function ("register_coco_instances") that can register COCO-formatted datasets.

Use custom datasets: https://detectron2.readthedocs.io/en/latest/tutorials/datasets.html

Next steps: Need to setup training config and run training/eval with detectron2 functions. Might need to setup a dataloader too?

Training Documentation: https://detectron2.readthedocs.io/en/latest/tutorials/training.html

Troubleshooting:

This helped with a specific PyCharm issue when I was using my entire home directory as the project directory, but I can't remember exactly what went wrong:

```
yes|keytool -importkeystore -srcstorepass changeit -deststorepass changeit -
srckeystore /etc/ssl/certs/java/cacerts -destkeystore
/home/tpmorgan/.config/JetBrains/PyCharmCE2021.3/ssl/cacerts
```

For help accessing attributes of the shapefile sources used to get the segmentation data:

https://gdal.org/python/osgeo.ogr.Feature-class.html.

For help with .gdb files:

https://support.esri.com/en/technical-article/000011973

https://gis.stackexchange.com/a/72661