**PROCESS DOCUMENT (Modified in 16/06/2022)**

Data available:

* Originial 3-band or 4-band imageries
* Geocoded shapefile or geojson of each imagery

Data preparation:

* Convert the geojson or shapefile or coded buildings to json file, which includes the colomn and rows information of the buildings (vertices).
* Split the data into training and validation, and create an integrated json file for each data folder called annotation.json
* DG\_data\_processing.ipynb is the script. Before running the code, create a images folder and an annotation folder at the destination folder where saves the training/val data output.

Visualization of the data created:

* Inspect\_building\_data.ipynb is the script to double check and visualize the data generated in the data preparation step. Do not forget to change the data directory.
* This allows you to visualize the data and json annotation that are created.

Training:

* Run the buildings.py in a terminal to train the building detection model.
* Sample code:
  + python buildings.py train --dataset /rapids/notebooks/sciclone/geograd/Miranda/github/Building\_Detection\_CN/processing\_data\_DG --weights coco
  + Check the buildings.py file to see the training options.
  + In the train function, you can set the train parameters, such as whether train the entire network, or just train the heads, or certain layers in the network, setting epoch or learning rate, etc.
  + In the **mrcnn** folder, the config.py has a list of parameters about the network that we can change, such as the size of the images to crop (if the input image is larger than the max size allowed, it will be cropped to the max size, if the input image is smaller than the minimum size allowed, it will be cropped to the min size image). **WARNNING**: when changing these parameters, **DO NOT change the config.py** **(Do not change any python code in the mrcnn library/folder),** add the parameter name to the building.py (under the BuildingConfig class) file to overwrite the default parameter values.

Validation/Checking on trained model results

* The log folder (folder name is set in the building.py file) has the trained results for each epoch, the result can be visualized by tensorflowboard: tensorboard --logdir=’the log directory’. Download the results to local directory, and run tensorboard --logdir=’the log directory’. Then go to <http://127.0.0.1:6006/> or what the terminal link it pops out to check on the training accuracy/loss changes over epochs.
* Check on the inspect\_building\_models.ipynb, and assign the model weights to the directory of the best model epoch result (.h5), and use the model to check on the prediction results for sampled dataset.