Retrieving Roads from Aerial Imagery

MUSA Capstone Project Final Presentation

Jiamin Tan

04/22/2022

Object

Use neural networks to segment roads from aerial images.

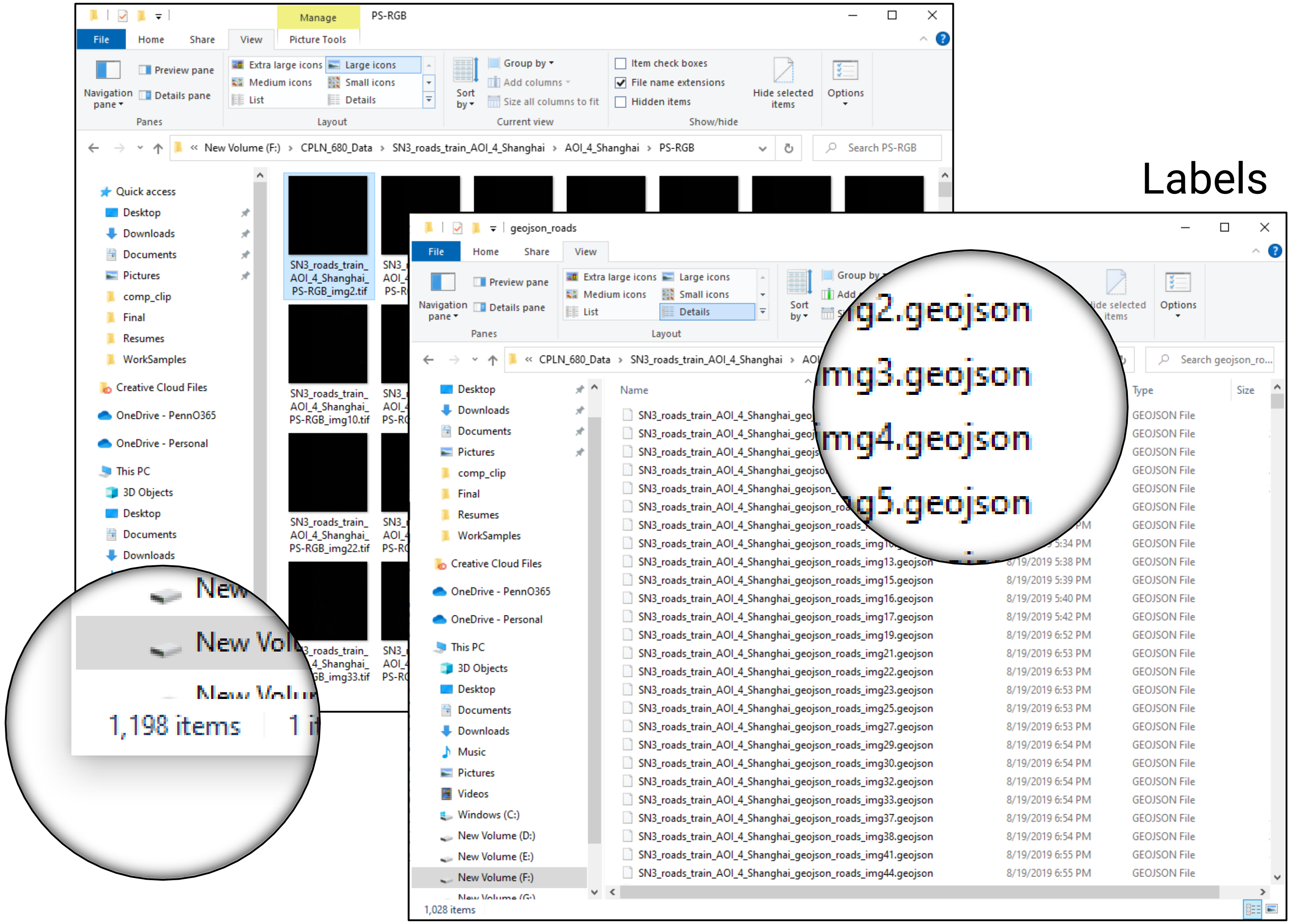
Data Source

https://spacenet.ai/

Data Overview

Training Image Set

Labels



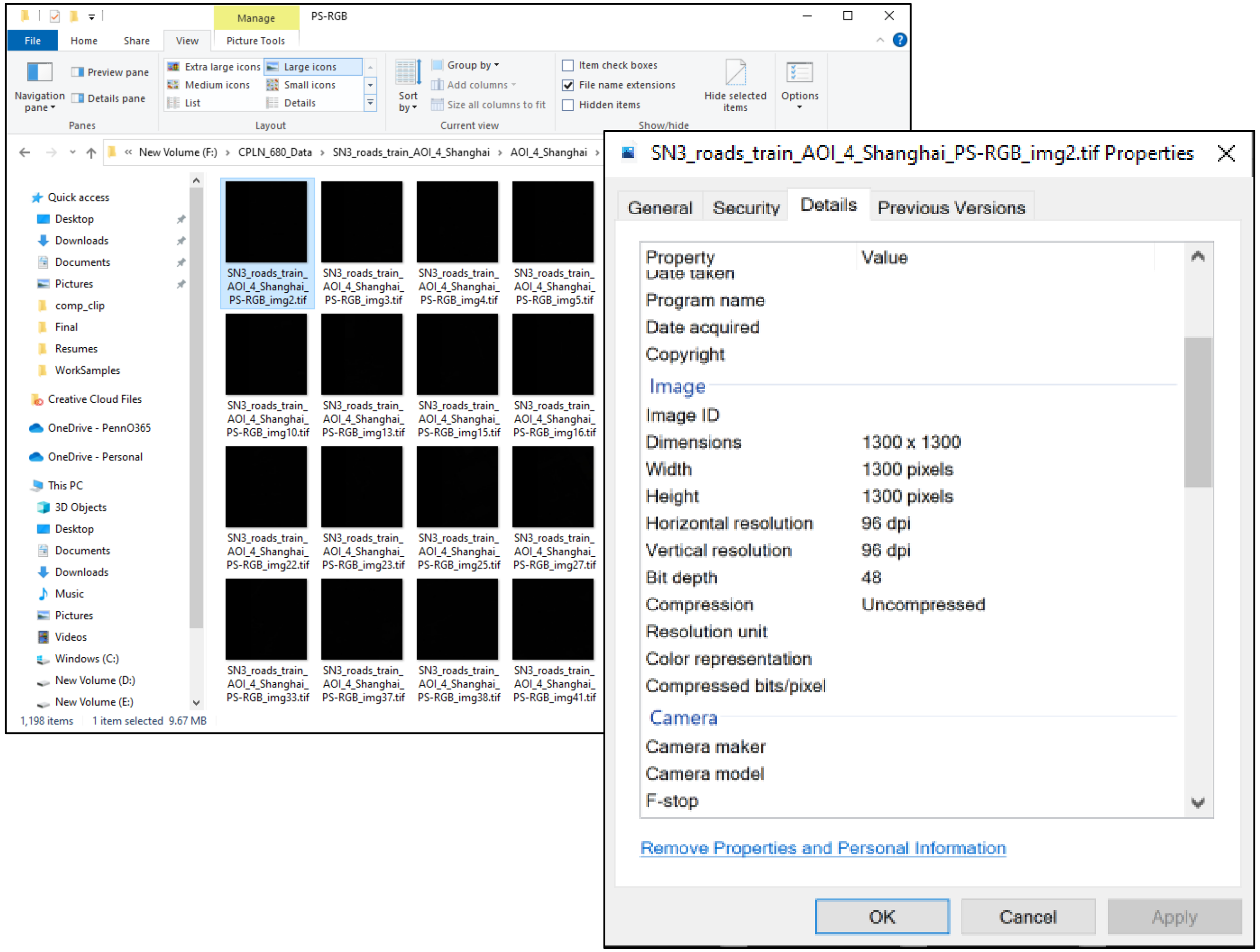
.tif files stored satellite images.

.geojson files stored strings (roads) as ground truth.

attributes such as **type** and **number of lanes** are included in the *.geojson* file.

Data Overview

Training Image Set



For each pan-sharpened Image:

1300 x 1300 pixels.

Each pixel has a spatial resolution of 0.31m x 0.31m.

Each tile is, therefore, 400m x 400m.

Bit depth is 48, so 16 bit for each band, and the value of a pixel in each band is from 0 to 65535.

Data Overview (Examples)

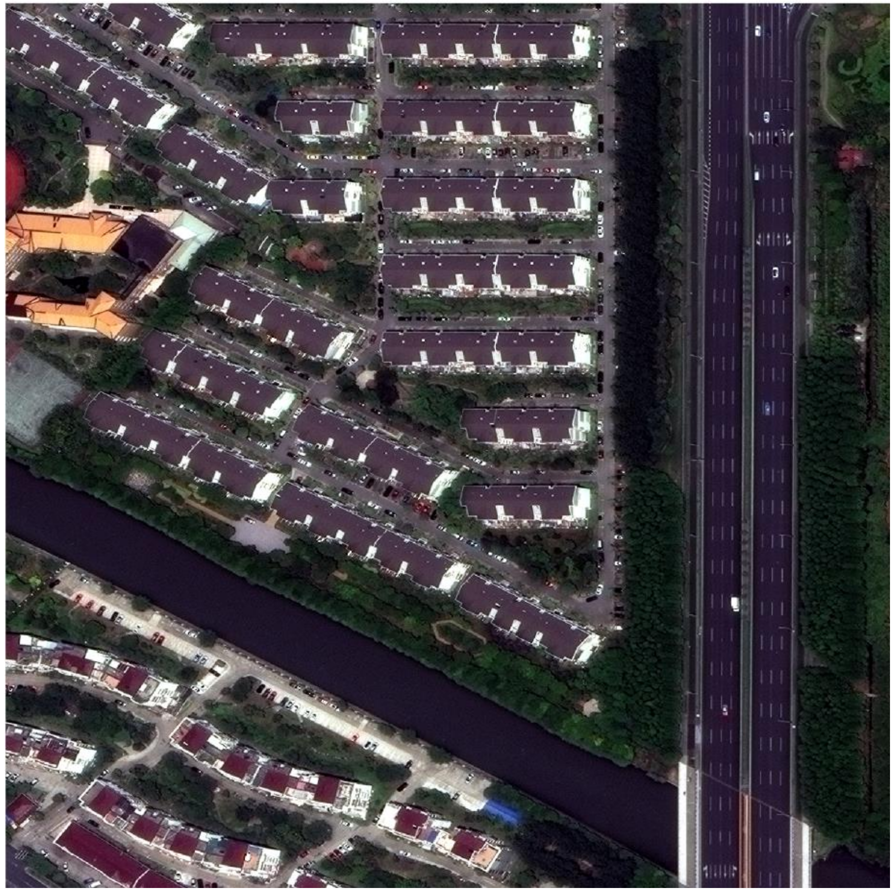


Image 73

Image 73 and its label

Data Overview (Examples)

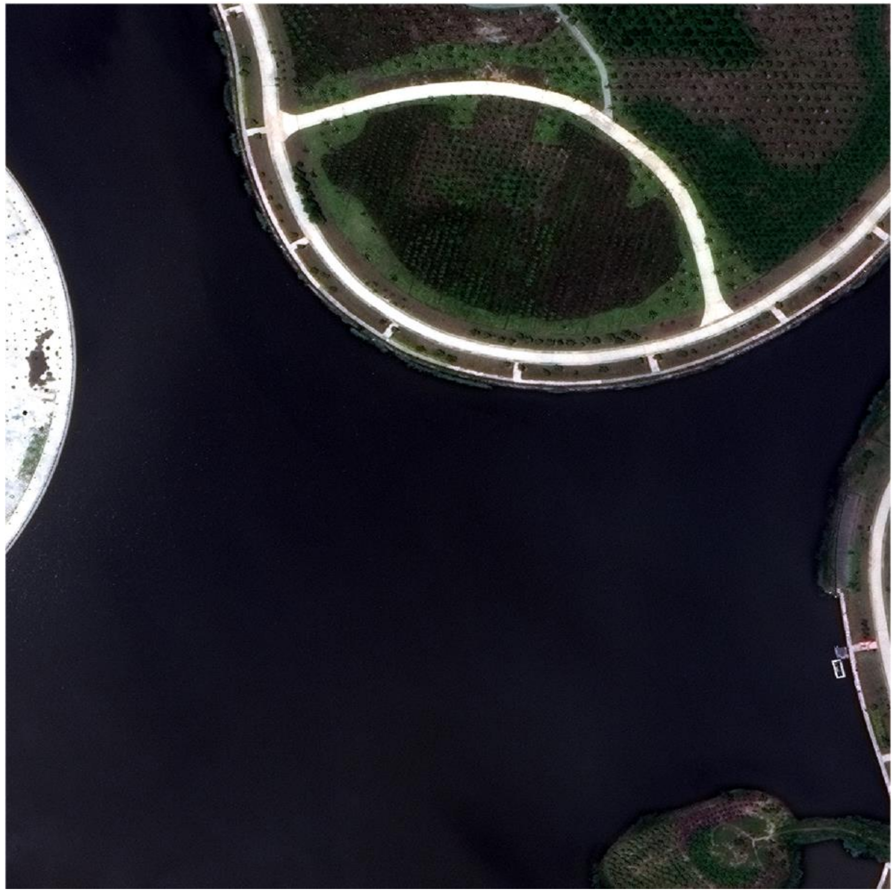


Image 52

Image 52 and its label

Data Overview (Examples)

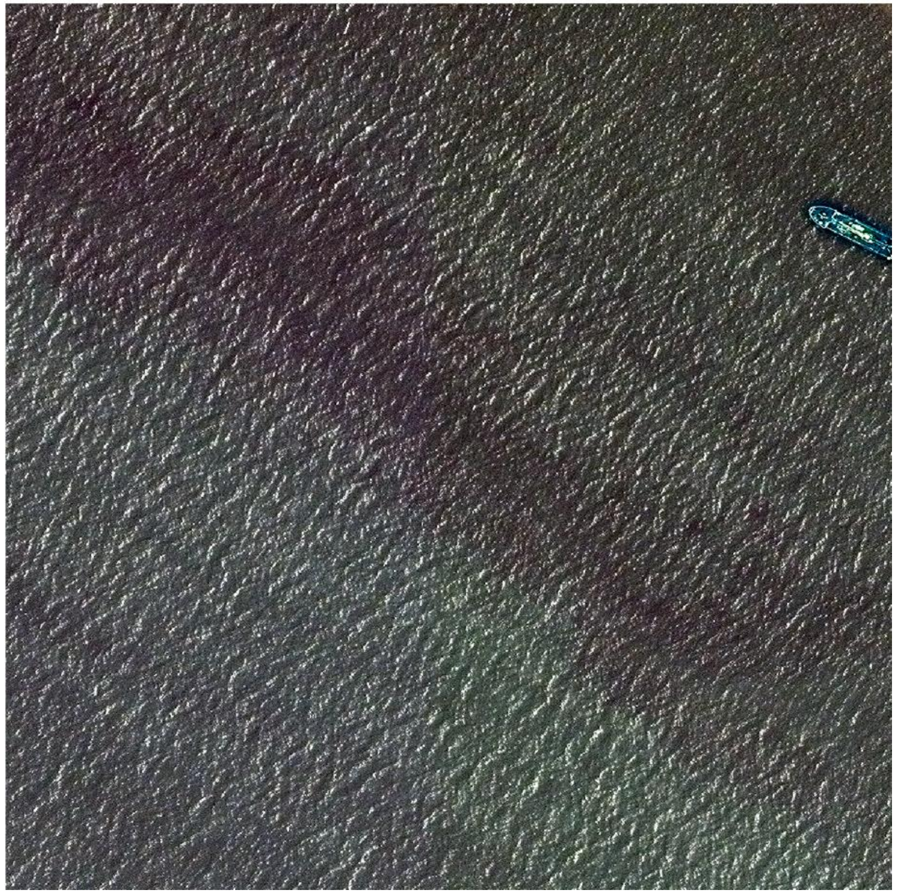


Image 432

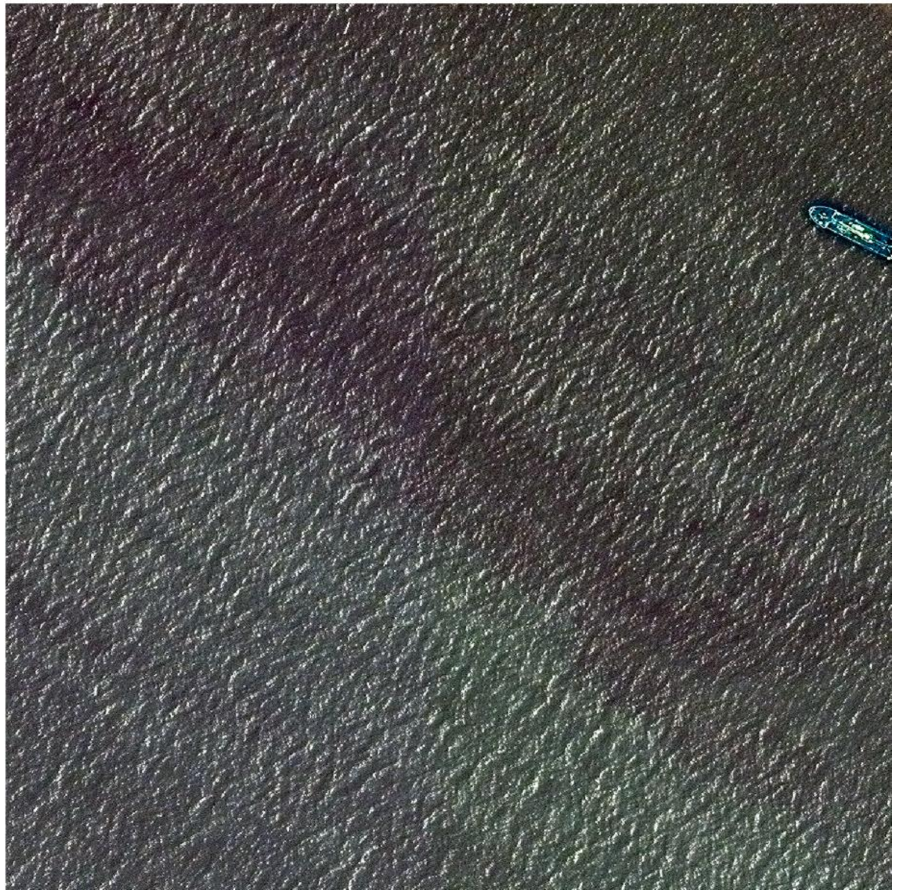


Image 432
does not have a label

Data Overview (Examples)

Image 52

Image 52 and its label

Preparation – Create Mask

A binary mask is generated for every image.

The mask is created by buffering the strings using *GeoPandas*.

The distance of the buffer is calculated by

3 meters x the number of lanes

Road types are not considered because 50% of the data were labeled as *"unclassified"* which do not have uniformed lane widths.

binary mask created for Image 435

Preparation – Create Mask (Example)

Image 435 and its label

binary mask created for Image 435

Preparation – Create Mask (Example)

Image 435 and its label

Image 435 and its binary mask

Preparation – Create Mask (Example)

Image 73 and its label

Image 73 and its binary mask

Preparation – Create Mask (Example)

Image 52 and its label

Image 52 and its binary mask

Preparation – Create Mask (Example)

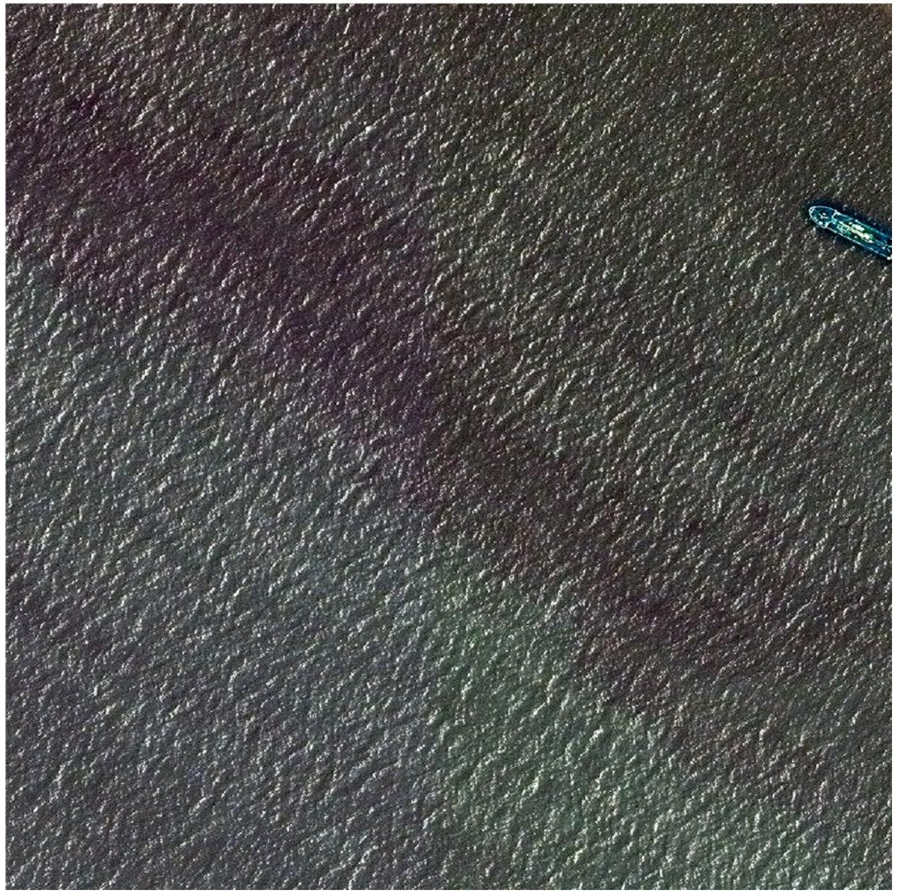
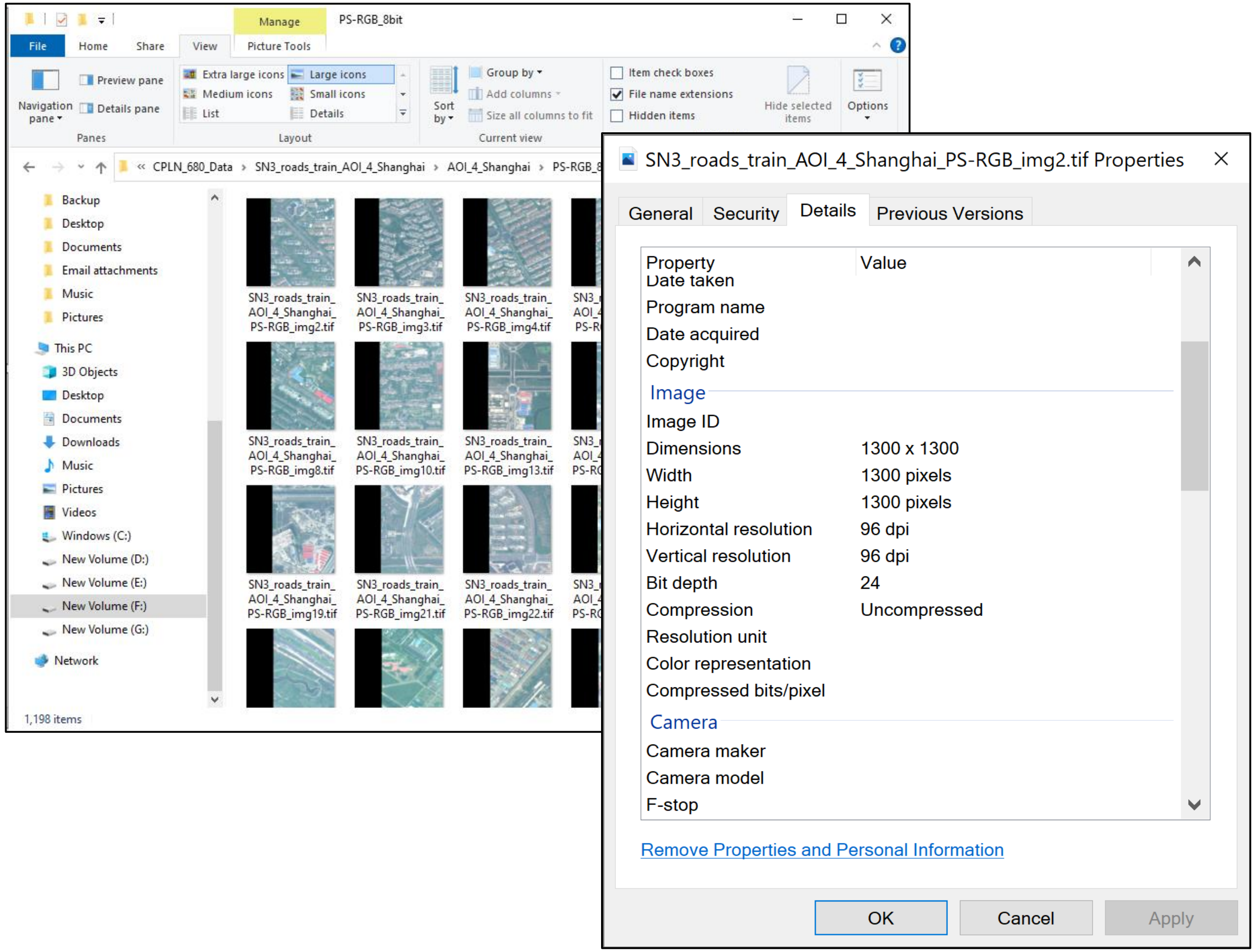


Image 432

binary mask created for Image 432

Preparation – Bit Depth Conversion

Training Image Set (8-bit)



48 bit-depth -> *24* bit-depth

8 bit for each band – value of a pixel in each band is from *0* to *255*.

Resolution and dimension remain the same.

Preparation – Training/Test Set Split

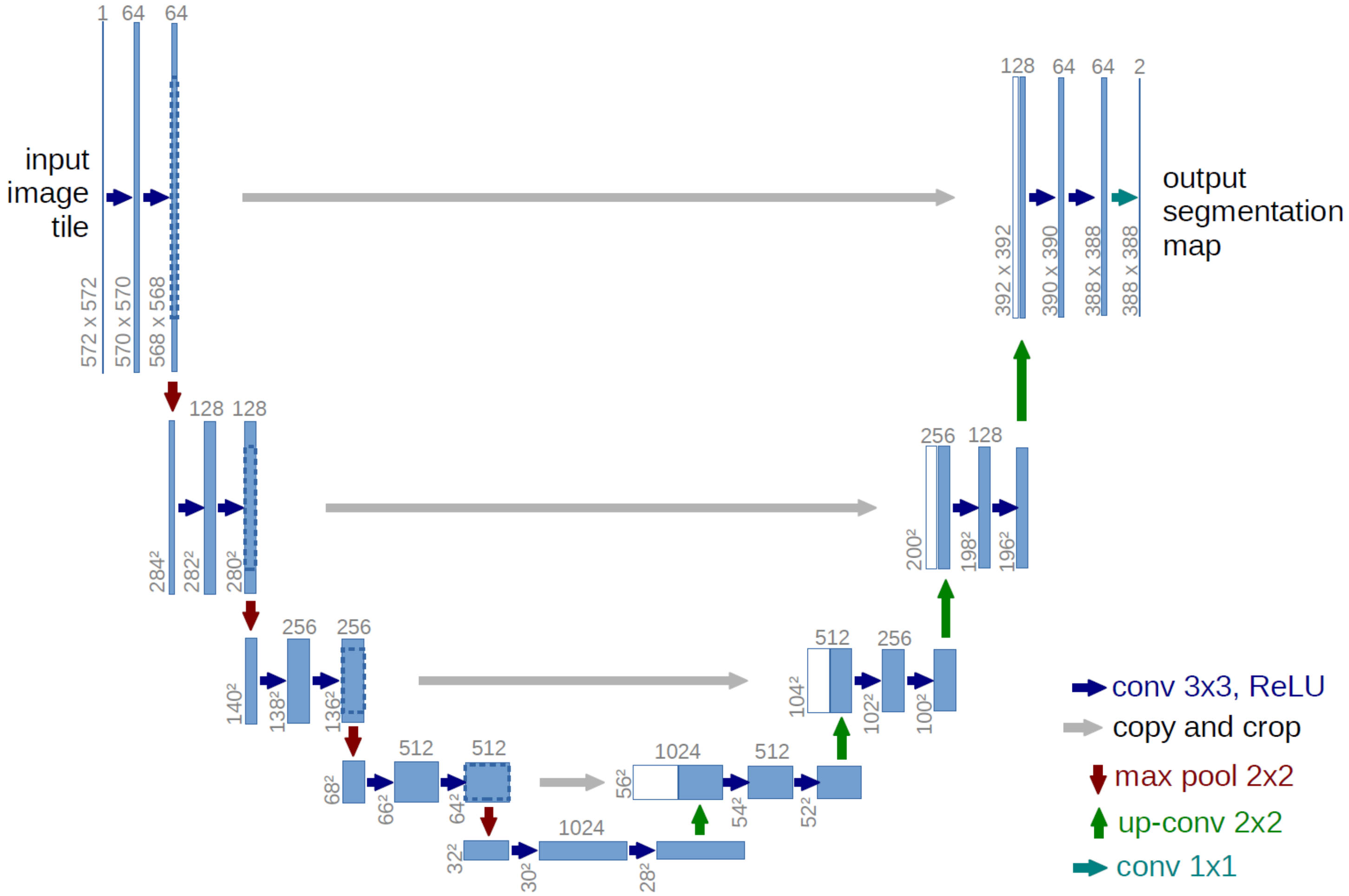
The *1198* 8-bit images and their binary masks are split into…

A *training* set with ***1070*** images (90%)

A *validation* set with ***59*** images (5%)

A *test* set with ***61*** images (5%)

Architecture – U-Net by Ronneberger et al. (2015)



https://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a/

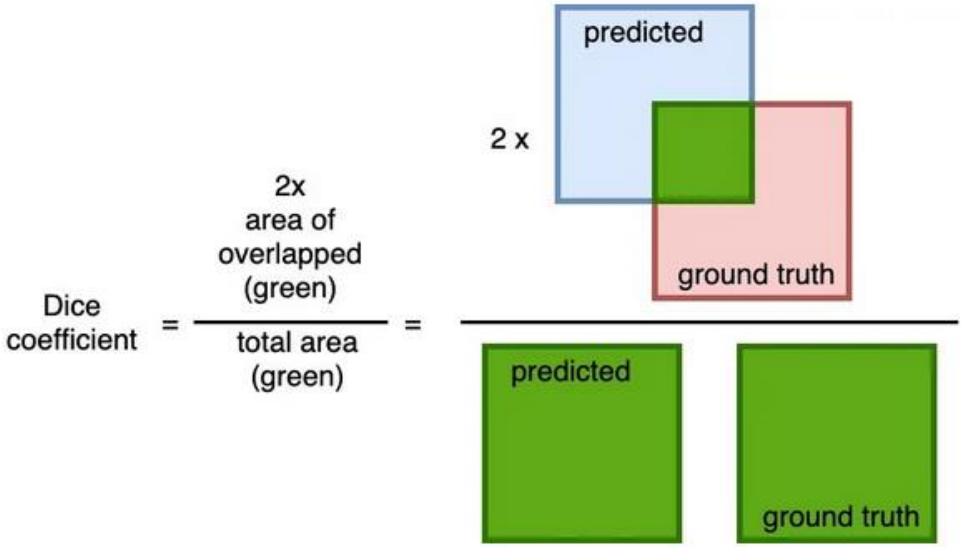
Model 1 – "Vanilla" U-Net

Using the same amount of convolutional and max-pooling layers in the original U-Net

Resize each image from (1300, 1300, 3) to (1024, 1024, 3)

Batch size = 2, epoch = 12, learning rate = 0.0001, optimizer = adam

Loss function: dice loss



https://datascience.stackexchange.com/questions/75708/neural-network-probability-output-and-loss-function-example-dice-loss

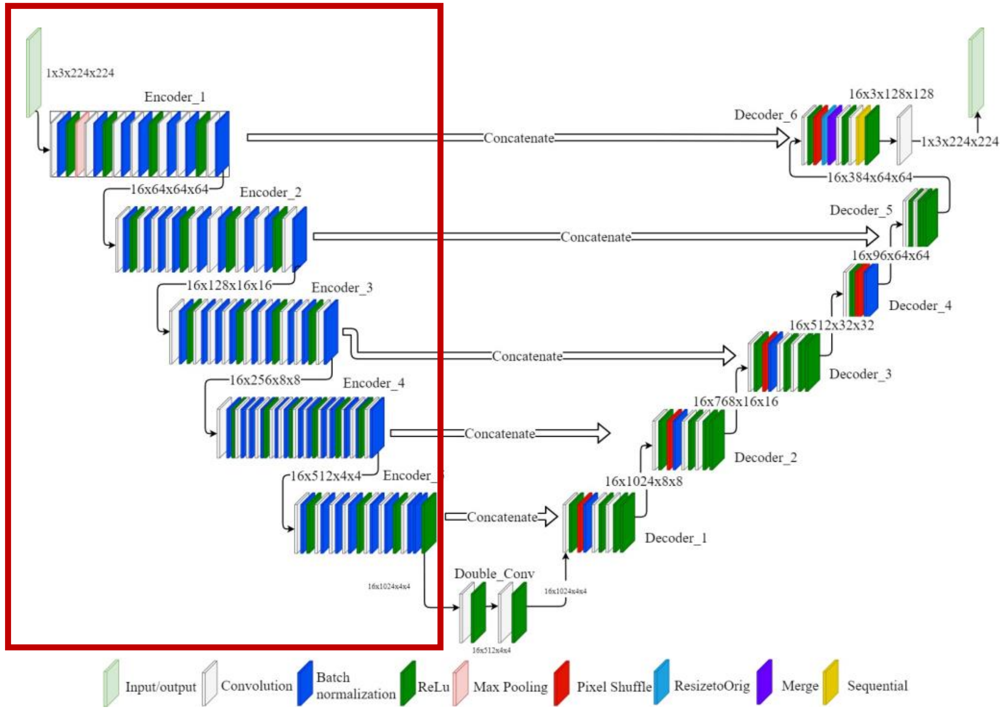
Model 2 – U-Net using Pre-trained ResNet50

For the encoding path, using a ResNet50 architecture pre-trained on “ImageNet”

Resize each image from (1300, 1300, 3) to (1024, 1024, 3)

Batch size = 2, epoch = 12, learning rate = 0.0001, optimizer = adam

Loss function: dice loss



https://forums.fast.ai/t/u-net-with-resnet34-backbone/85744

Results – F_1 Score

A F1 score was calculated on predictions generated by the test set.

F1 score is a harmonic mean of precision and recall. It is a metric used for imbalanced data.

F1 score after fitting Model 1
0.6685

F1 score after fitting Model 2
0.6569

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

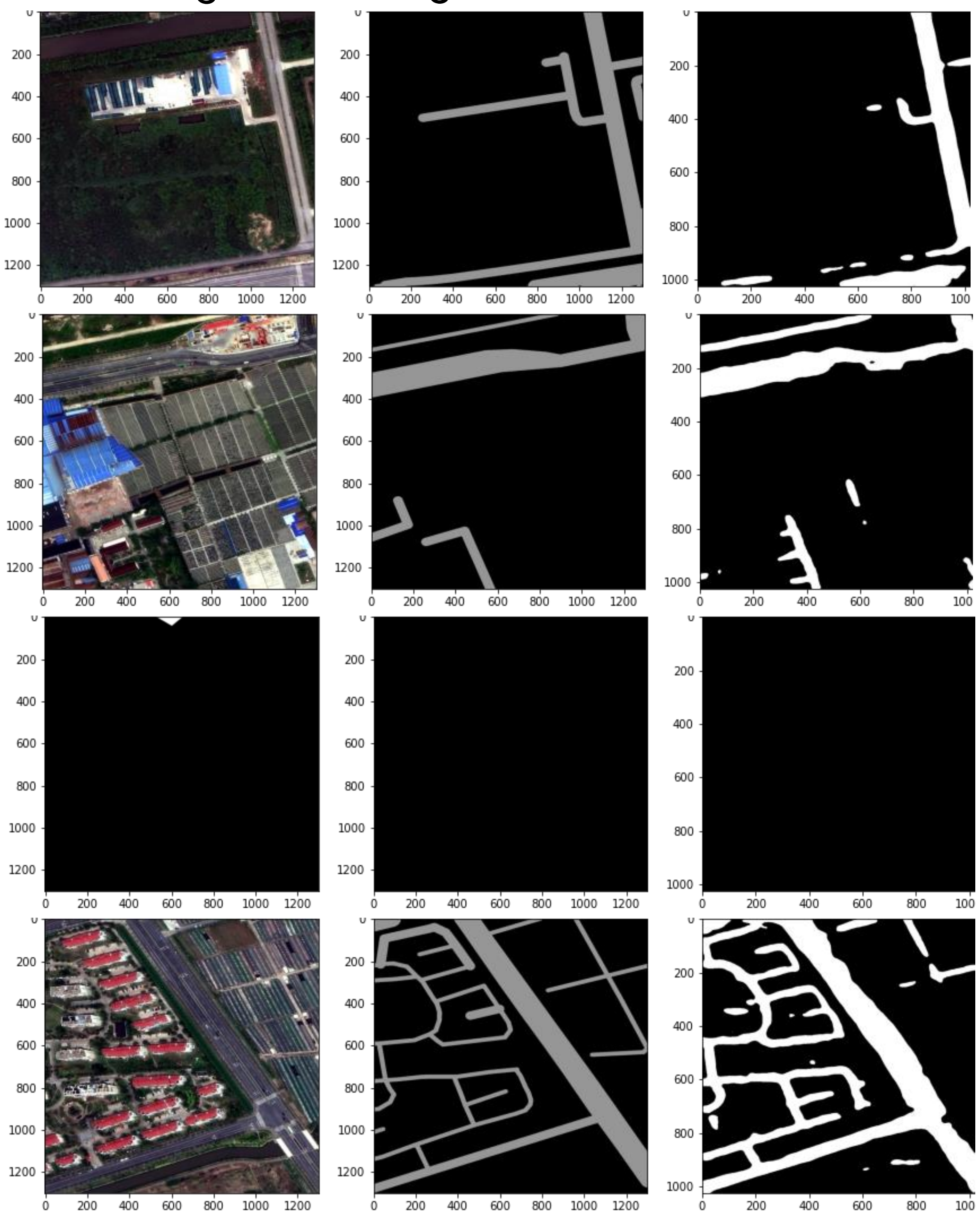
$$Recall = \frac{TP}{TP + FN}$$

$$F1\text{-}score = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

https://www.tutorialexample.com/an-introduction-to-accuracy-precision-recall-f1-score-in-machine-learning-machine-learning-tutorial/

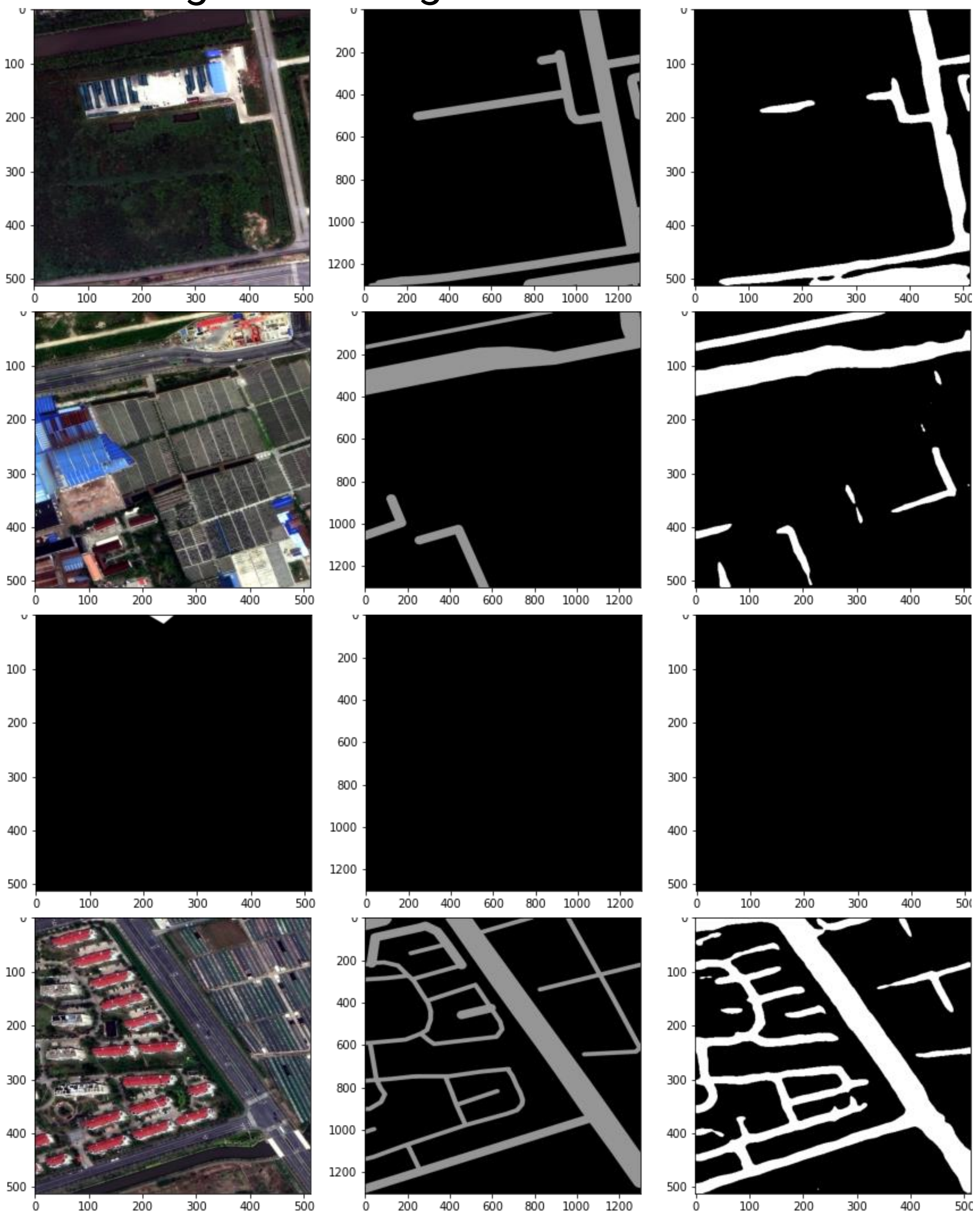
Model 1

Image | Original Mask | Prediction



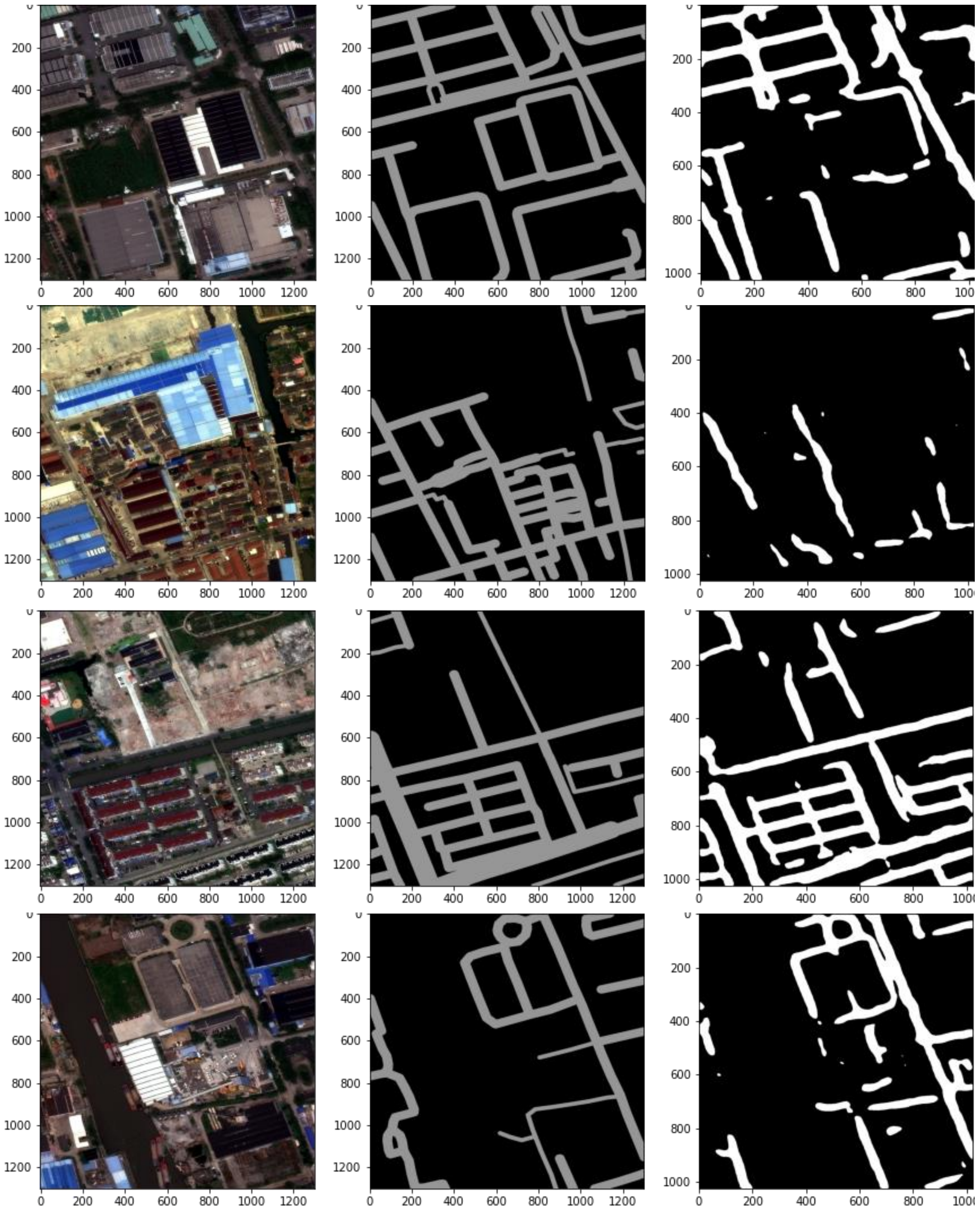
Model 2

Image | Original Mask | Prediction



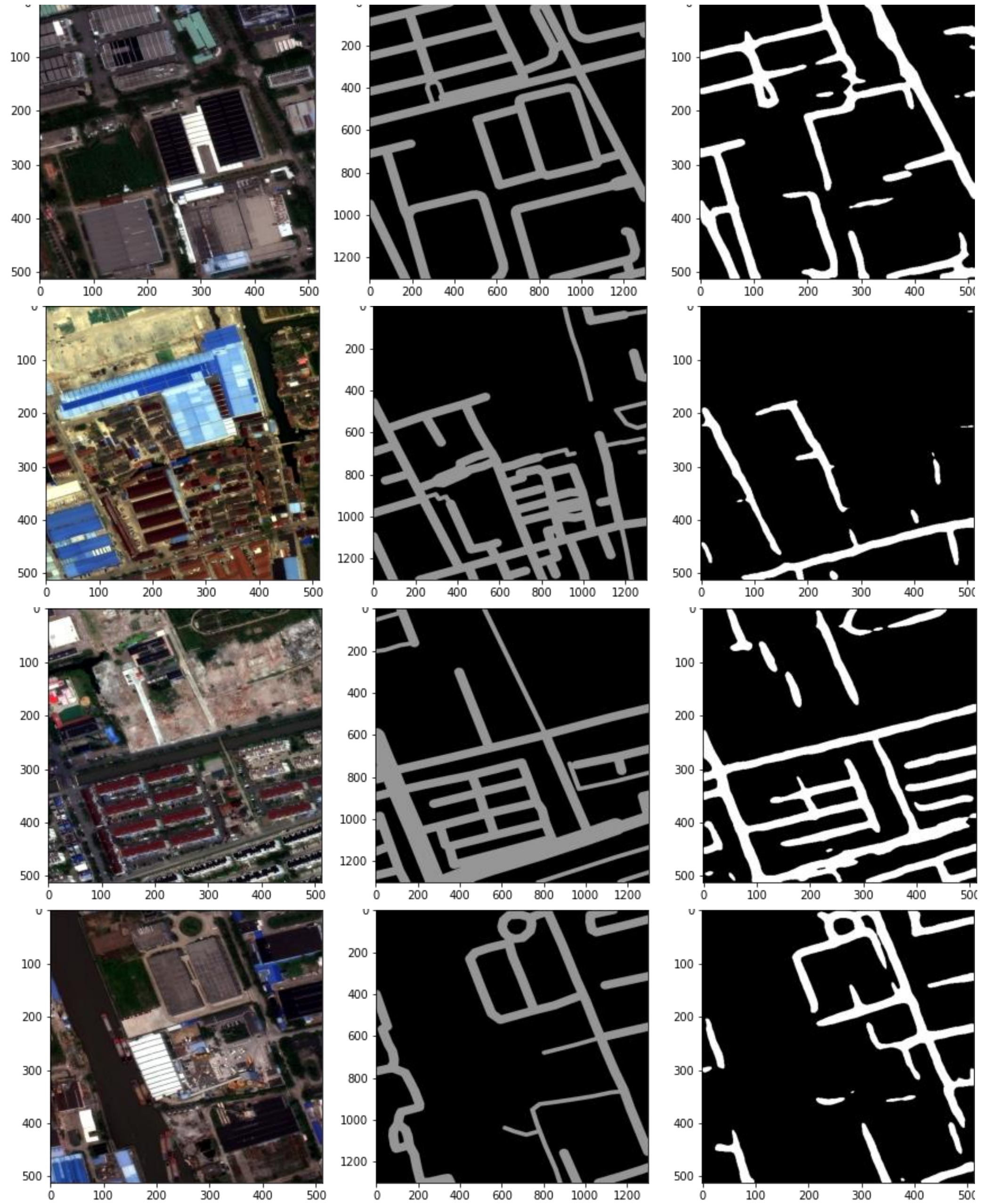
Model 1

Image | Original Mask | Prediction



Model 2

Image | Original Mask | Prediction



Alternatively – Training/Test Set Split

The *1198* 8-bit images and their binary masks are split into…

A *training* set with ***1070*** images (90%)

A *validation* set with ***59*** images (5%)

A *test* set with ***61*** images (5%)

Randomly select *120 (1300, 1300)* images

Split each of them into *25 (256, 256)* images

A *training* set with ***3000*** images (95%)

A *validation* set with ***150*** images (5%)

Also split each of them into *25 (256, 256)* images

A *test* set with ***1525*** images

Alternatively – Training/Test Set Split (Example)

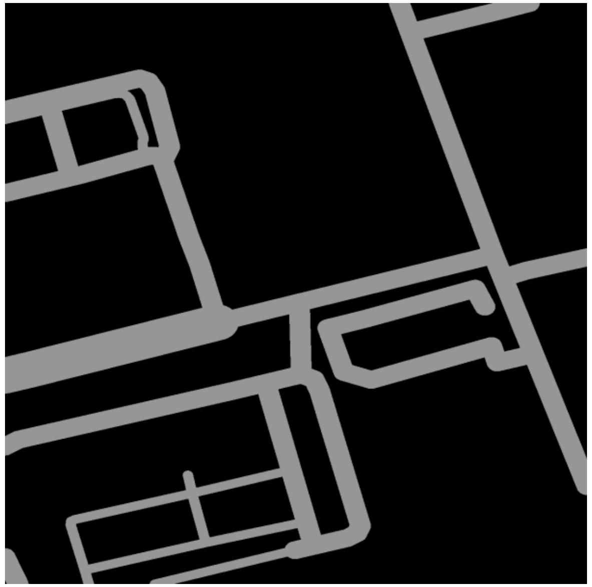
Run Models on Split Images

The input of each model is now (256, 256, 3). Other parameters remain the same

(1300, 1300) Images	(256, 256) Images
F1 score after fitting Model 1 0.6685	F1 score after fitting Model 1 0.6269
F1 score after fitting Model 2 0.6569	F1 score after fitting Model 2 0.6292

Run Models on Split Images (One Example)

Image 1497



Original Mask

Model 2 Prediction
Using the Whole Image

Concatenation of
Model 2 Prediction
Using Split Images

Future Directions

Both models used in the project are existing architectures. Keep learning about neural networks.

Find a more precise way for buffering the strings.

Think about how the raster result can be effectively transferred to vectors.

How to preserve Connectivity?