TRAINING AN EYE IN THE SKY

Image Detection with Machine Learning

4-22-2020

Nick Harbeck

XinyuYao

OUR GOAL

Distinguish between five sets of similar features

Roads and Tracks

Buildings and Misc. Structures

Trees and Crops

Large and Small Vehicles

Standing and Moving Water

TRAINING AN EYE IN THE SKY

DISTINGUISHING SIMILAR FEATURES IN SATELLITE IMAGERY

AGENDA

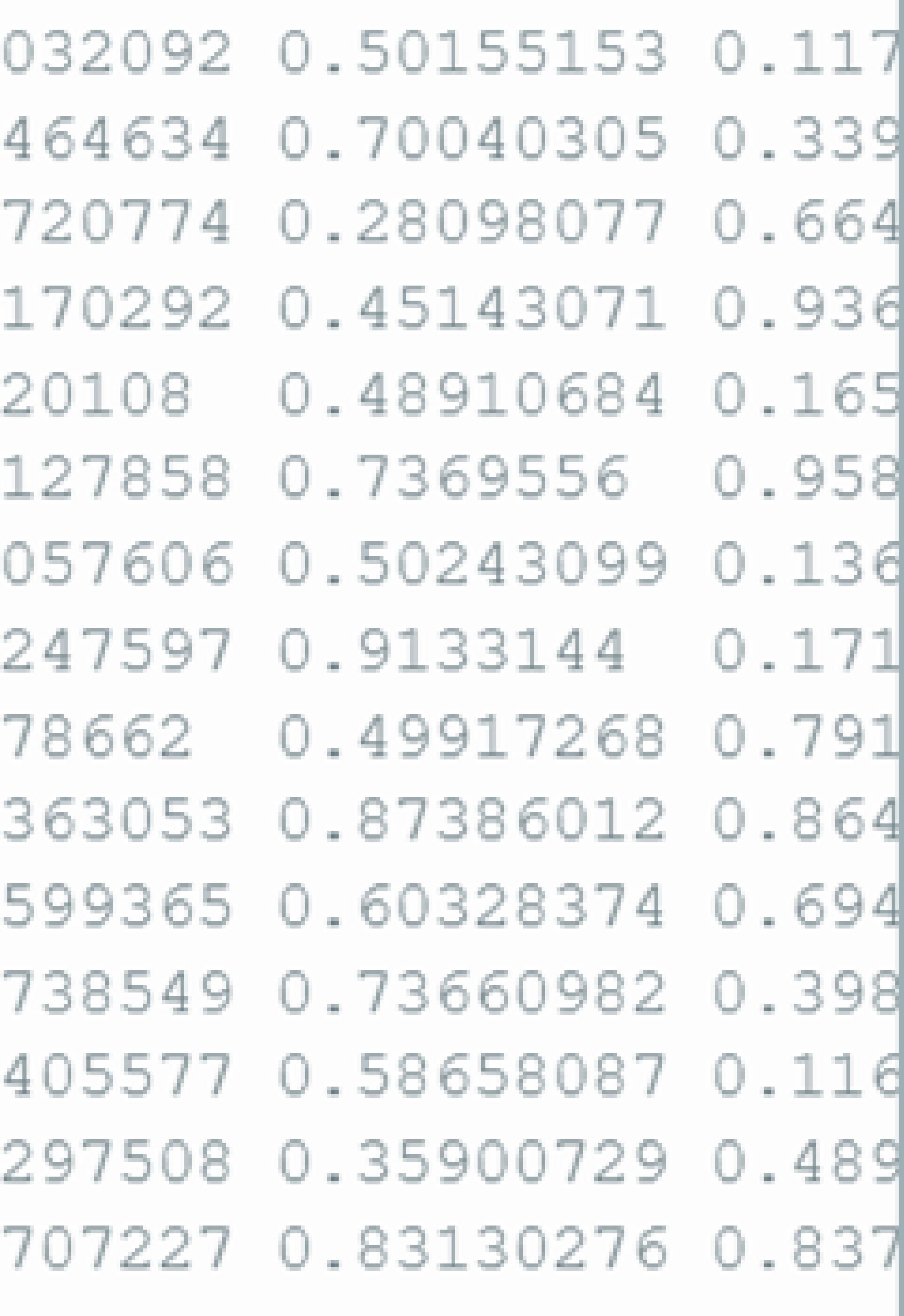
- What are we trying to detect?
- Why we want to use machine learning
- Data background
- Methodology
- Results and conclusions

WHAT ARE WE TRYING TO DETECT?

The United Kingdom's Defence Science and Technology Laboratory (DSTL) collects large amounts of satellite image data

DSTL created a Kaggle competition in December 2016 that included multi-spectrum satellite images containing key features:

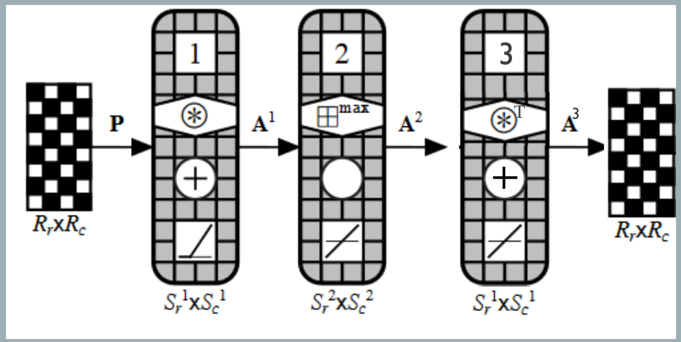
Buildings, Miscellaneous Manmade Structures, Roads, Tracks, Trees, Crops, Waterways, Standing Water, and Vehicles

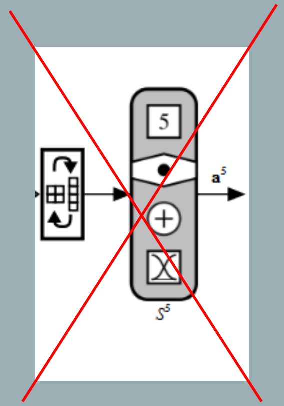


WHY WE WANT TO USE ML

Neural networks are powerful tools for finding the relationships between data and some target output.

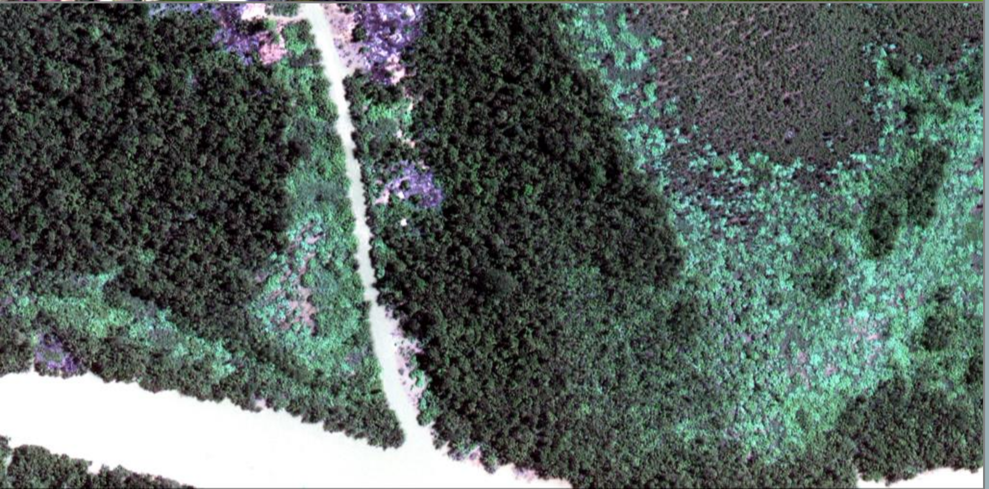
In this case, a U-Net (a type of CNN built for image segmentation) is a good network for our data.





Convolution → Upsampling

No Dense Layers

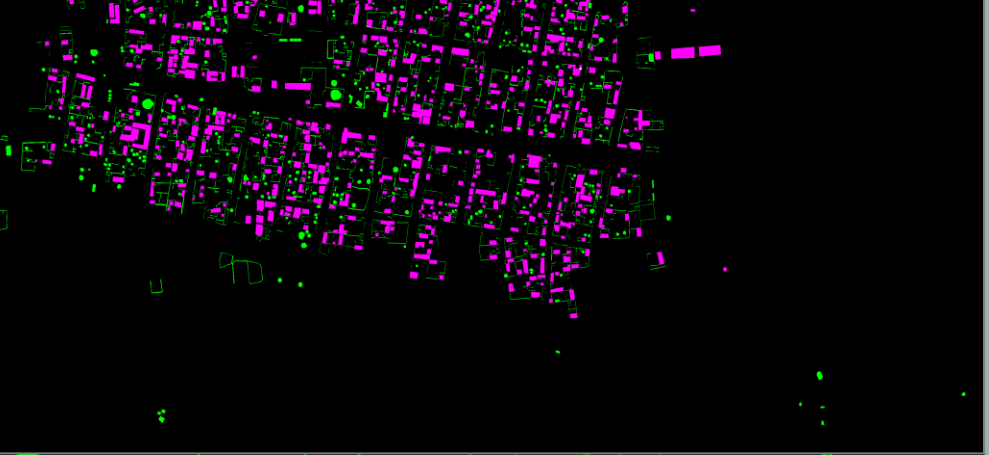
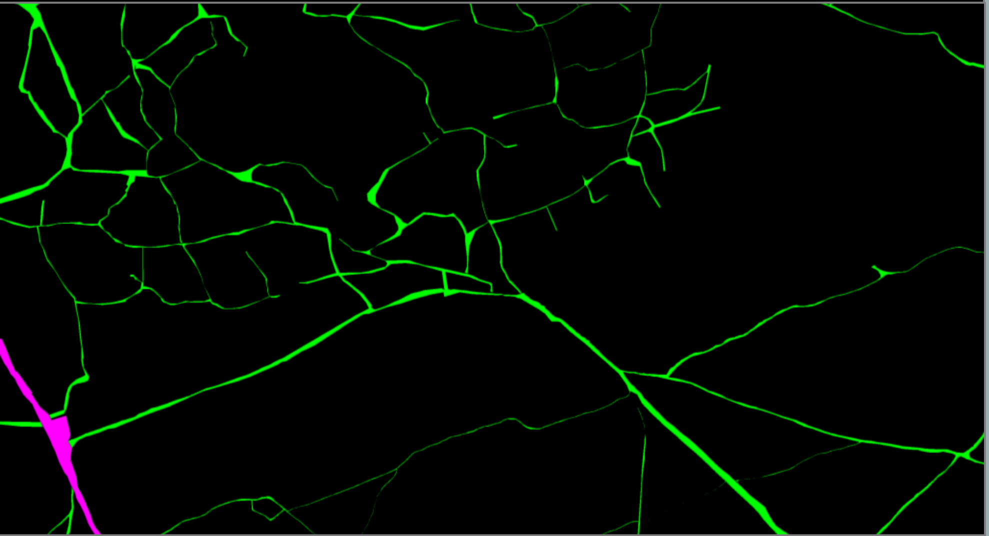
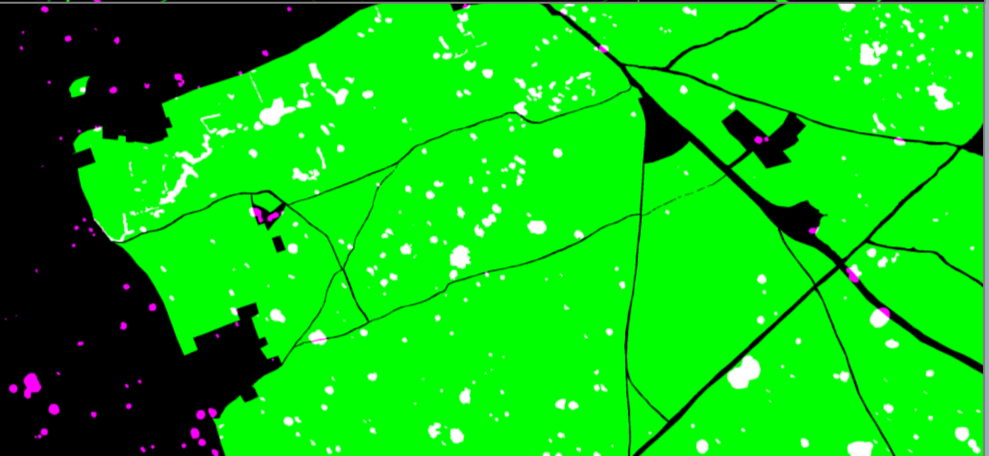


DATA BACKGROUND

The DSTL dataset contains 150 3-band and 16-band satellite images of various locations in the United Kingdom

Each training image has labeled polygon coordinates that correspond to a target – 10 targets in total.

Since some data types were similar (e.g. large and small vehicles) we started preprocessing by grouping together those training sets and masking the images with labeled polygons.

METHODOLOGY

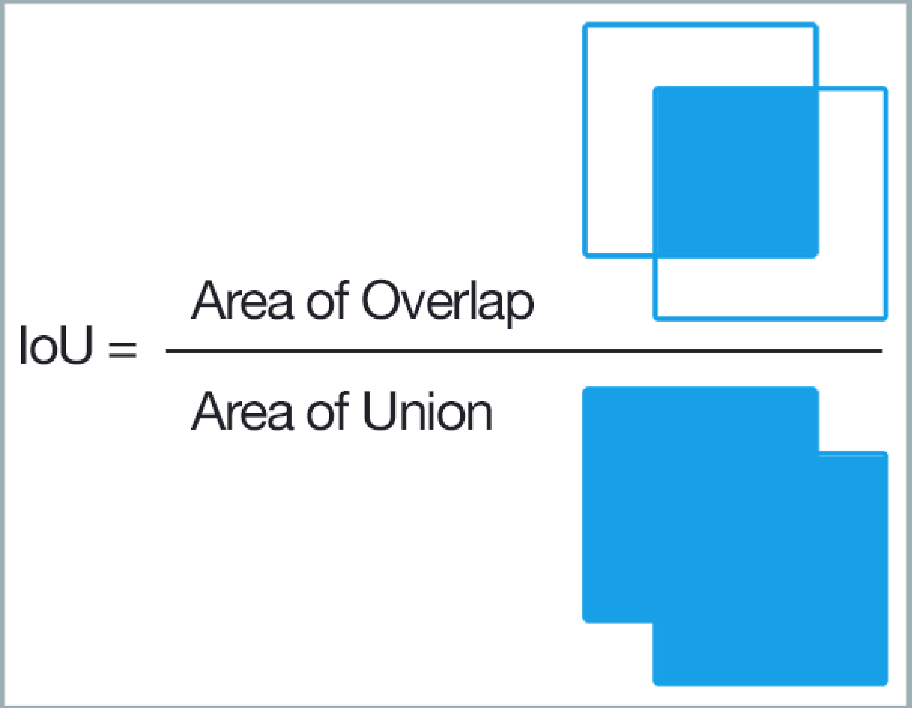
After preprocessing, we had five sets of masked training data that had been transposed and could be split into different specialized image segmentation U-Nets.

This data was fed into U-Nets built in the Keras ML library.

A generator within the network randomly splits the input images into smaller chunks via cropping for better training.

The U-Net had 10 convolutional layers and included normalization, Exponential Linear Unit (ELU) activations, pooling, and cropping to achieve better performance.

METHODOLOGY



The Jaccard index was our performance metric and it is a statistical measure of the similarity between sample sets.

After training each of the U-Net feature models, they could be merged to predict the presence of each of our target classes.

Pytorch was also tested as a framework, but the performance of the Keras models were able to achieve our goals.

Image from Wikipedia

RESULTS AND CONCLUSIONS

Type of Feature Detection	Training Jaccard Index	Test Jaccard Index
Roads and Tracks	0.47	0.40
Buildings and Misc. Manmade Structures	0.54	0.49
Trees and Crops	0.55	0.48
Large and Small Vehicles	0.26	0.11
Standing and Moving Water	0.21	0.04

Some features were easier to distinguish than others, but this is consistent with existing research on this dataset.

These results could also be consistent with the scores on the leaderboard.

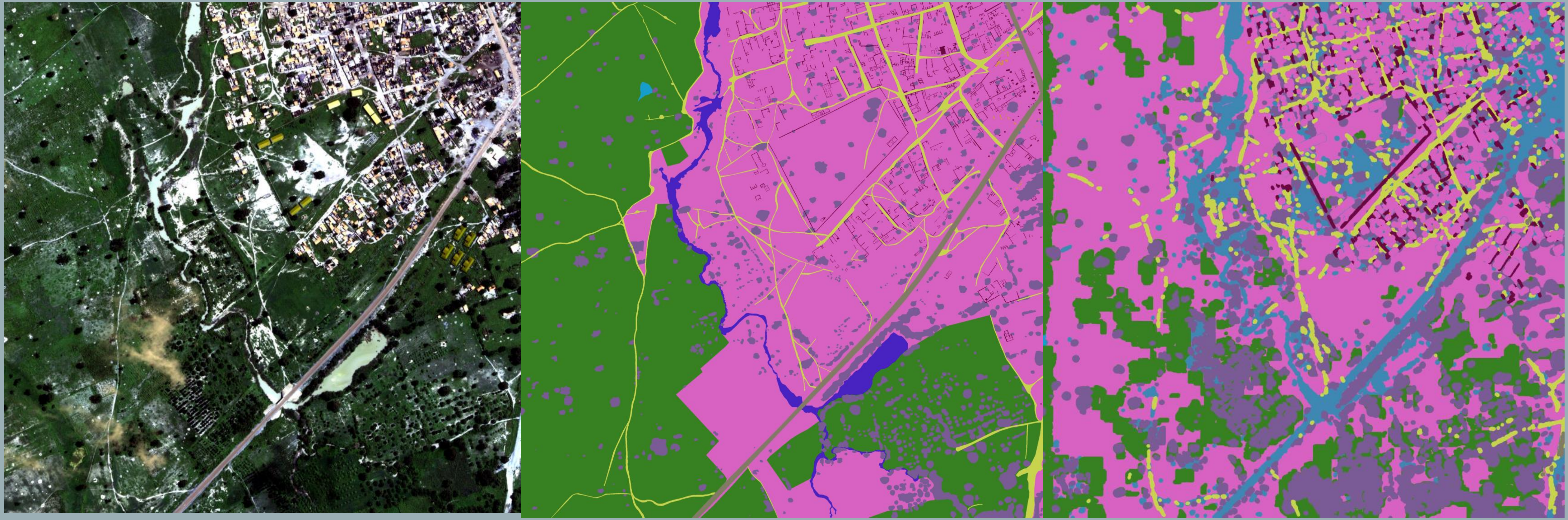
RESULTS AND CONCLUSIONS

Example Raw image of 6100_2_3 in the training set (left), mask generated from labeled polygon (middle) and segmented image predicted (right)

RESULTS AND CONCLUSIONS

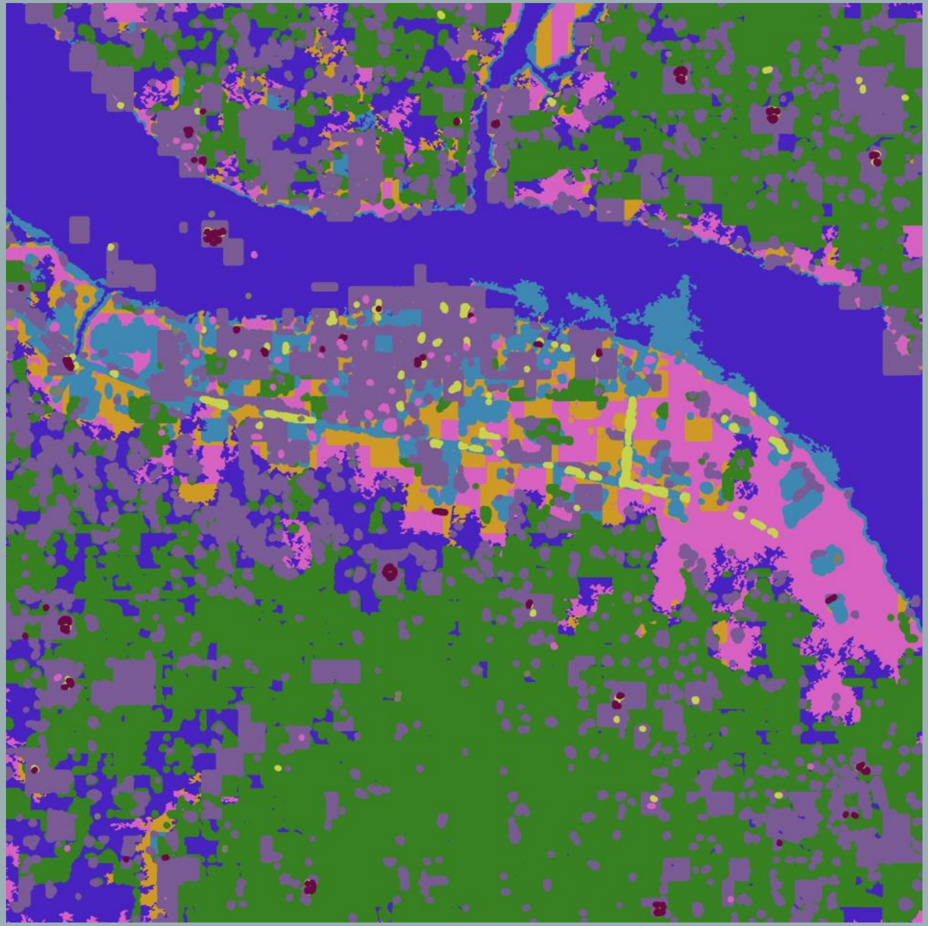
Example Raw image of 6110_3_1 in the training set (left), mask generated from labeled polygon (middle) and segmented image predicted (right)

RESULTS AND CONCLUSIONS

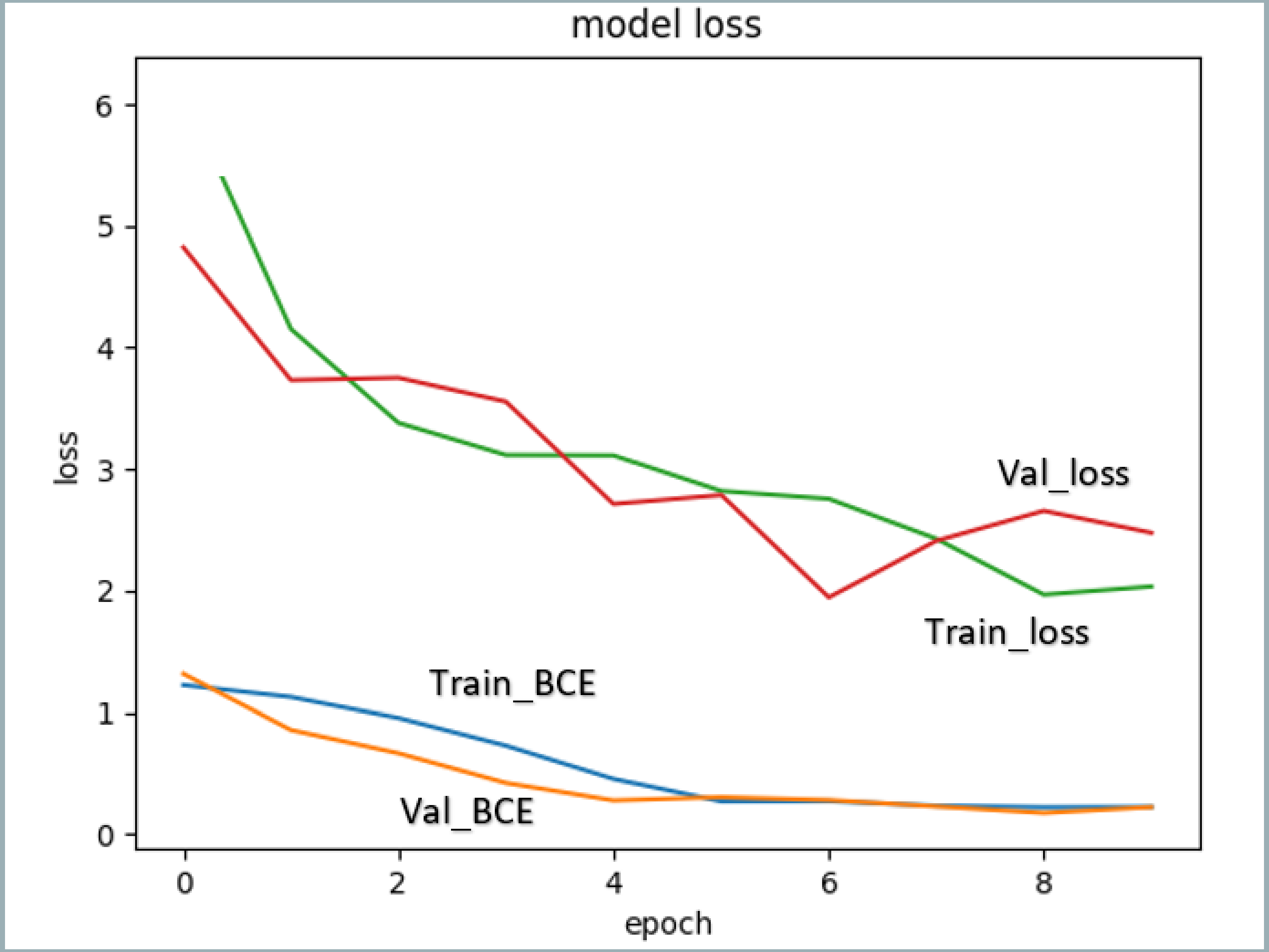


Example Raw image of 6100_2_2 in the training set (left), mask generated from labeled polygon (middle) and segmented image predicted (right)

RESULTS AND CONCLUSIONS



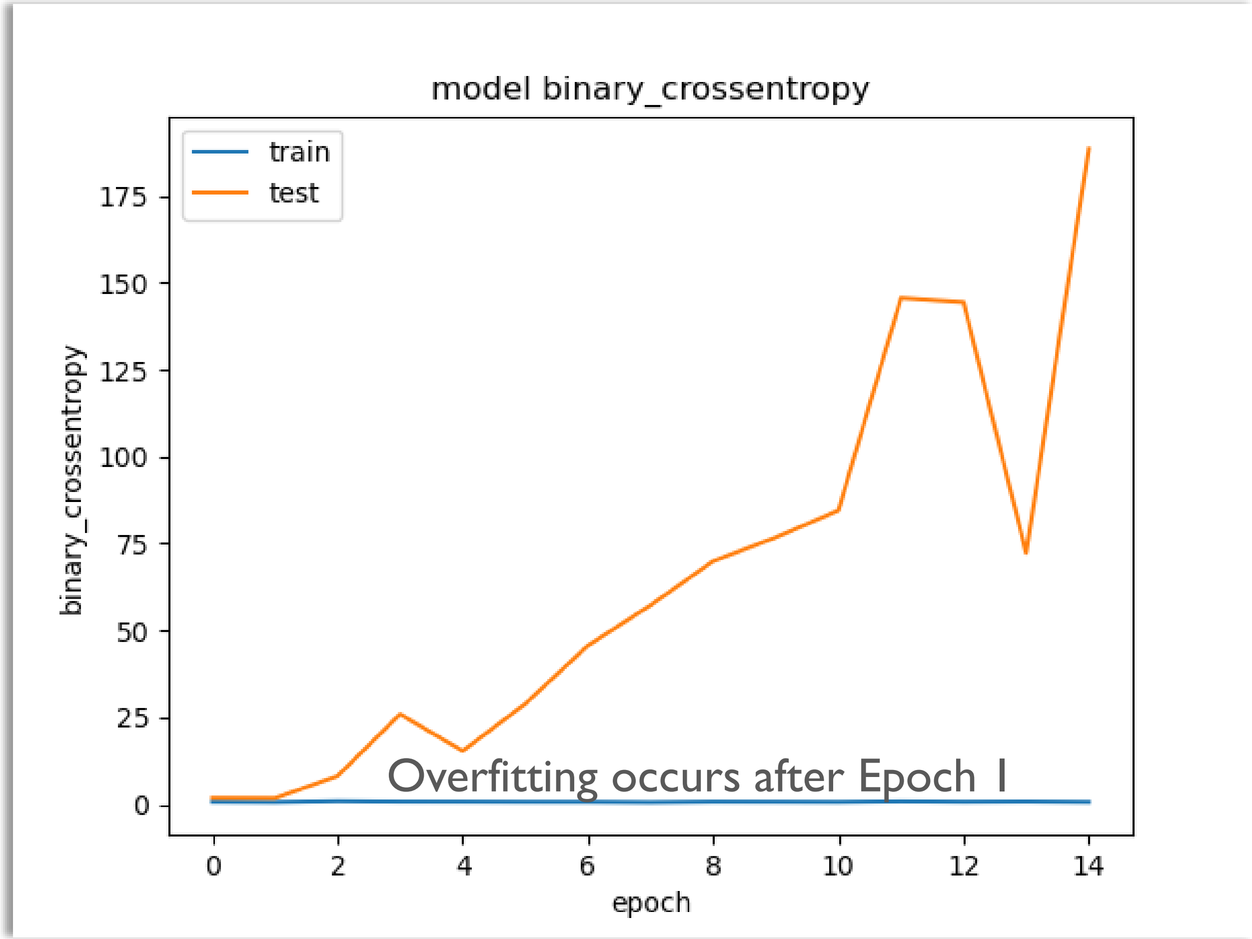
Example Raw image of 6050_4_4 in the test set (left), and segmented image predicted (right)



Example of a BCE loss function scores for the building and structure data.

RESULTS AND CONCLUSIONS

Model converge as loss function (-log(jaccard) + BCE) and BCE loss decrease with epochs.



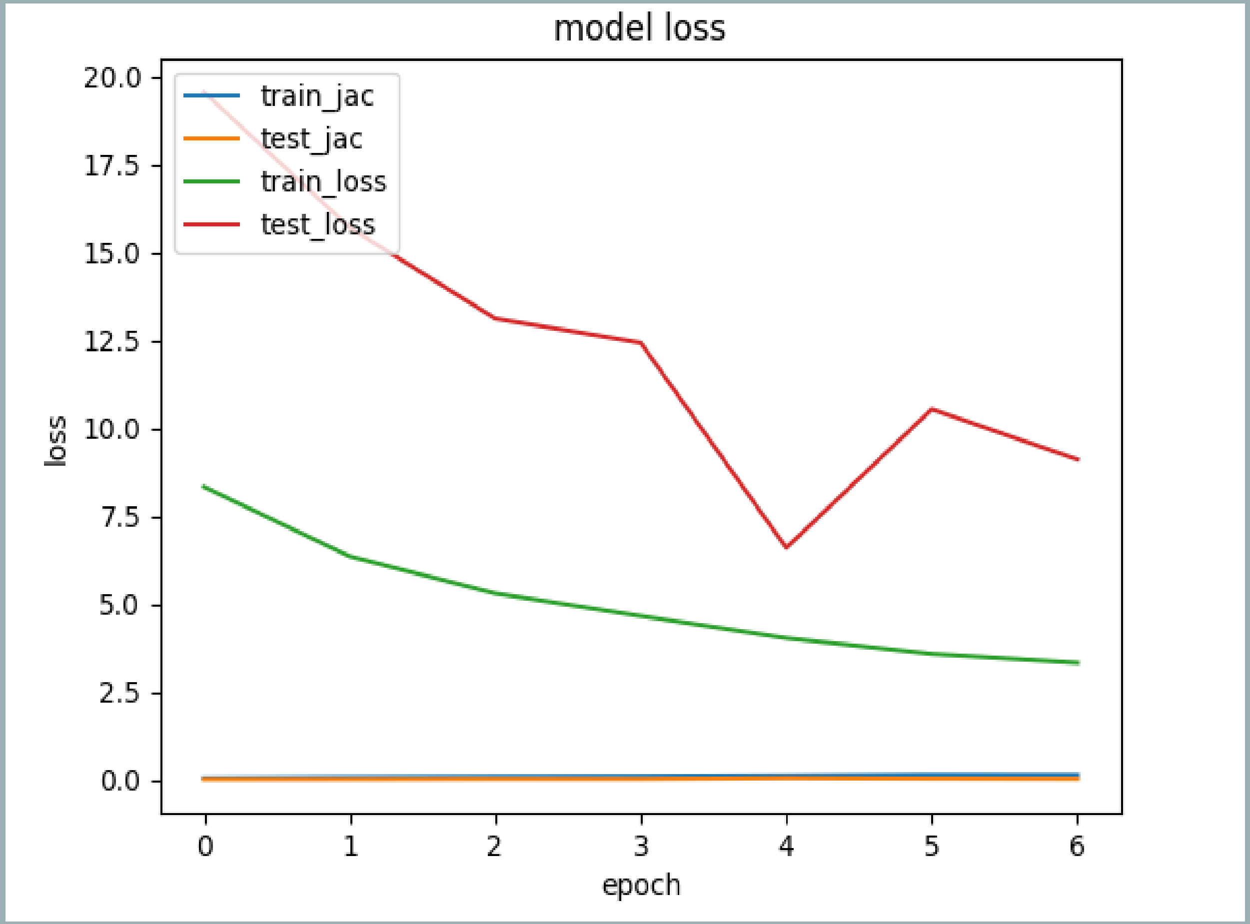
Example of a BCE score for the trees and crops data.
Epochs stop before overfitting occurs.

RESULTS AND CONCLUSIONS

The number of epochs and the amount of samples in each epoch played a significant role in improving performance.

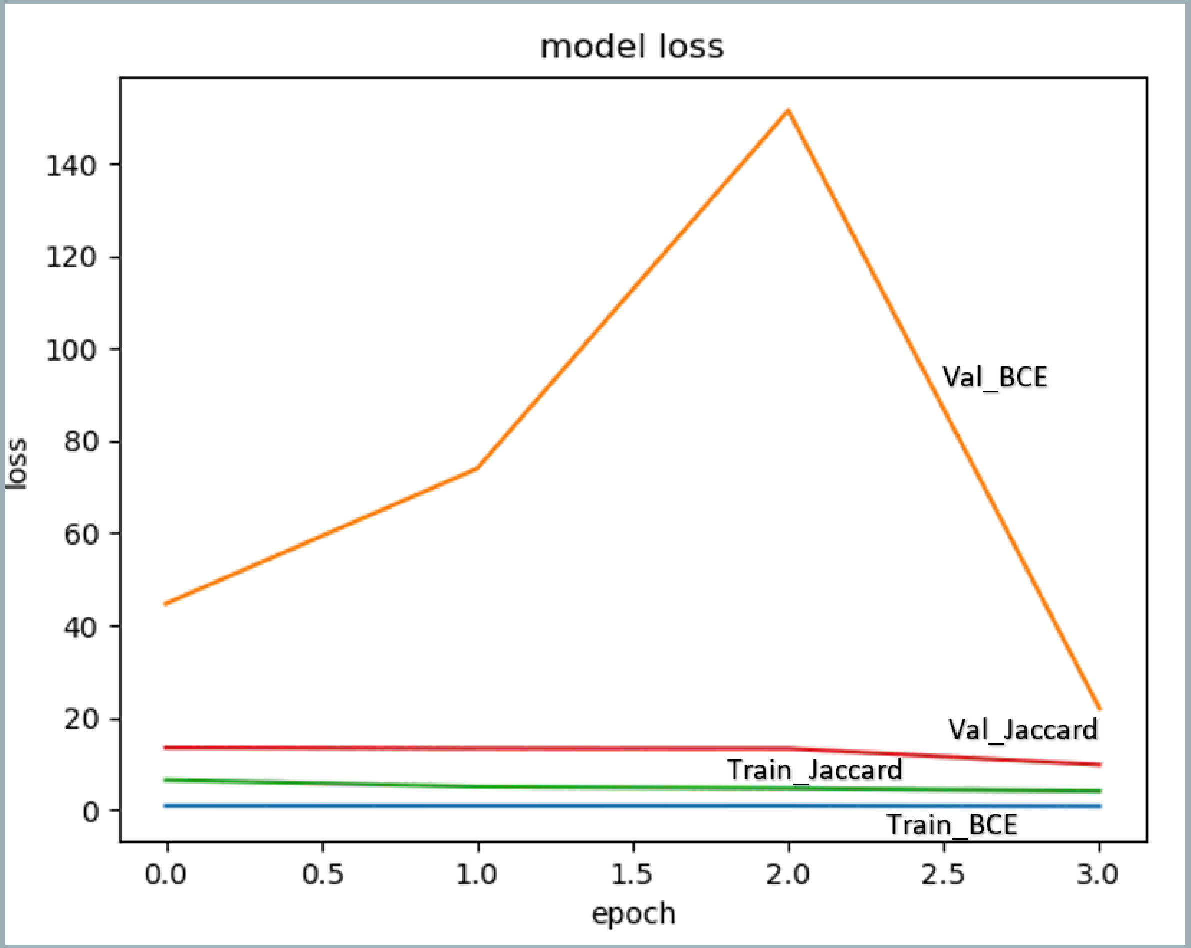
RESULTS AND CONCLUSIONS

Road and Tracks



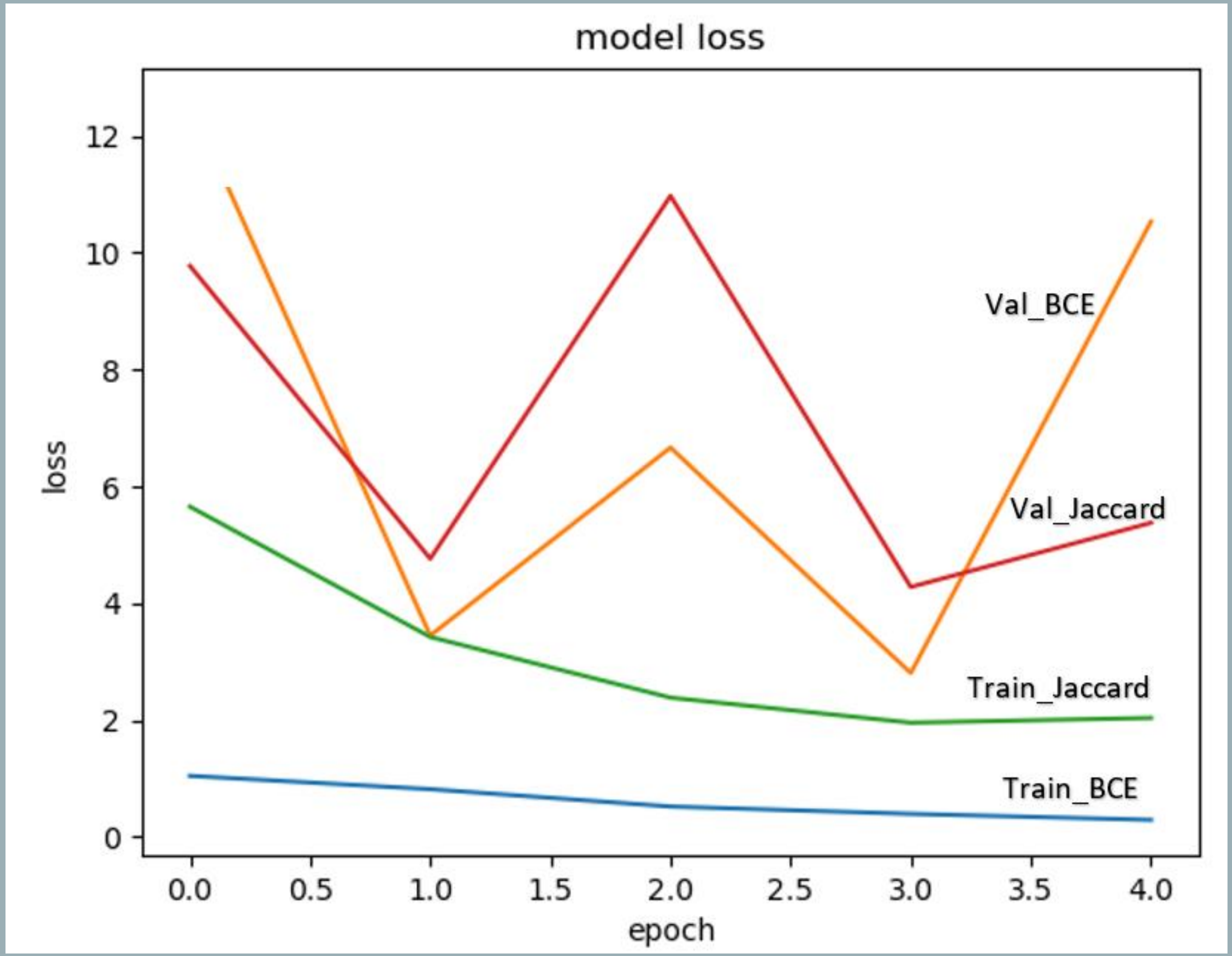
RESULTS AND CONCLUSIONS

Trees and Crops



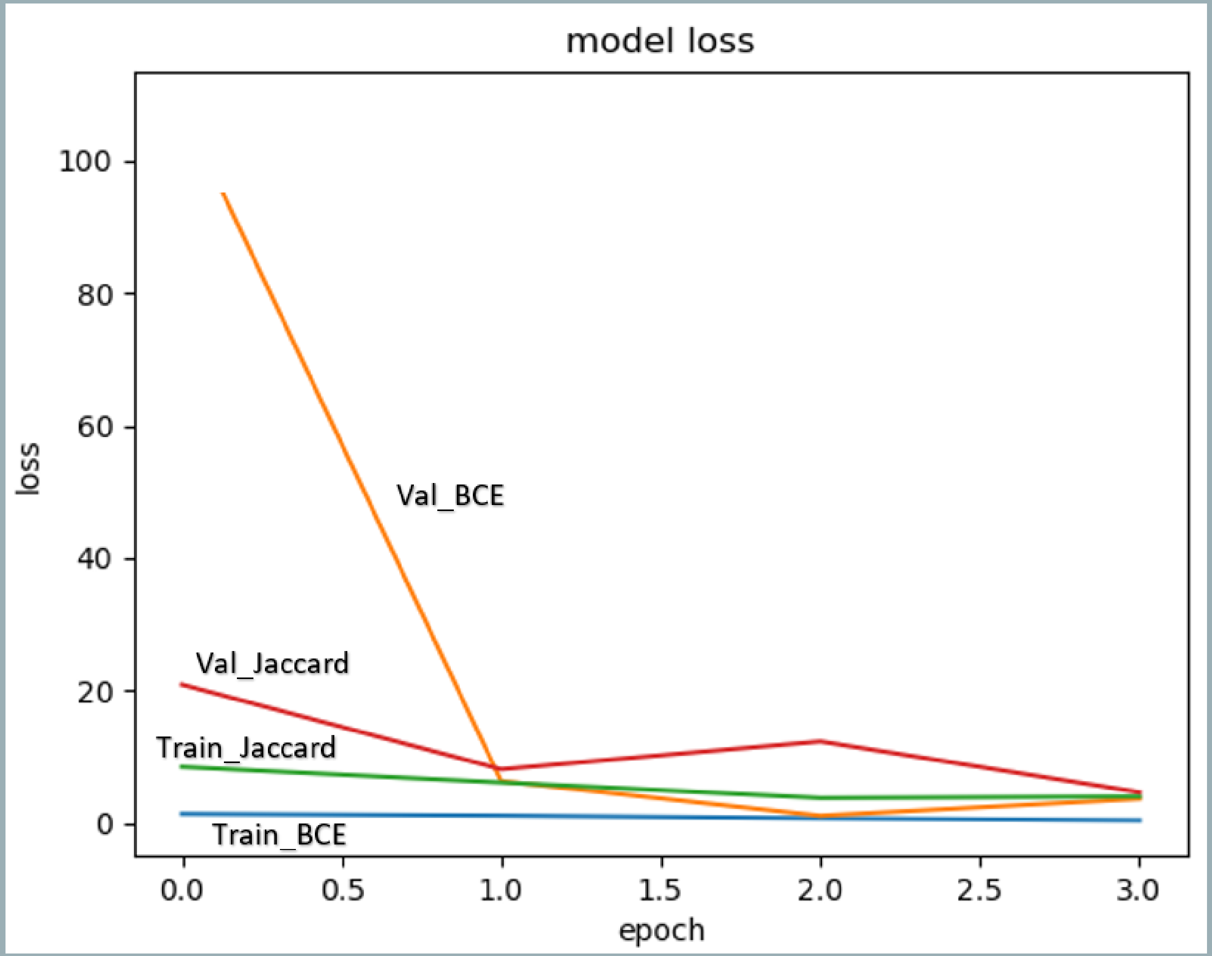
RESULTS AND CONCLUSIONS

Large and Small Vehicles



RESULTS AND CONCLUSIONS

Standing and Moving Water



RESULTS AND CONCLUSIONS

Limitations and Challenges

Time

The most impactful limitation to this project was a lack of time.

Because our goal was to distinguish between two similar features, we had to build many models. The large size of the data meant it took a very long time to run these models and perform our analysis, even with the help of GPU!

Data

A surprising challenge was building models from the training data.

Each picture was massive, but there were not that many unique images. This increased the likelihood of overfitting and reduced performance because subsamples of our training data were augmented and duplicated multiple times. Some features were also remarkably similar and hard to distinguish.

RESULTS AND CONCLUSIONS

Learnings and Future Opportunities

The U-Net architecture lent itself well to new techniques such as batching the images from our dataset or implementing new types of augmentation.

There were many existing resources and research to help guide this project.

This project could be the first step of a multi-tiered pipeline build to take advantage of image segmentation and explore other relationships in the data.

Image segmentation is a unique style of NN learning and techniques learned in this project could apply to many other problems.