Brooke Ann Coco
Project 5 Summary
Signs from Above: Building Detection from Satellite Imagery
June 18, 2019

Overview:

With the ever-looming threat of climate change, the frequency of major disasters has been increasing at an alarming rate. When it comes to disaster recovery, time is of the essence, pressuring response teams to be both quick and efficient. Unfortunately, most disaster survey efforts are performed manually, which is not only time-consuming, but can also be prone to error.

To this end, the goal of this project is to train neural networks to autonomously recognize and map building footprints from satellite imagery taken before and after disaster events. Based on the difference, response teams can determine precisely where they should prioritize their rescue operations and relief efforts, thus improving the speed and efficiency with which they respond.

Tools:

Python			
Processing	**Modeling**	**Visualization**	**Storage**
Geopandas Gdal Rasterio OpenCV NumPy Re PIL	Keras Tensorflow	Matplotlib Scikit-image	Pickle H5py EC2

Data:

Training data was obtained from Spacenet, a collection of publicly available commercial satellite imagery and labelled training data hosted by Amazon Web Services (AWS) for the purpose of machine learning research. Specifically, this project utilizes 3,651 3-band (RGB) 200mx200m tiles overlooking Las Vegas at 30cm resolution. Before and after disaster imagery was obtained from DigitalGlobe.

Pre-Processing:

Before modeling, I performed several standard pre-processing techniques to prepare the data for analysis. First, all of the source images, stored as GeoTIFFs, needed to be converted from 16-bit to 8-bit. This was necessary to enable me to open and display the source images on my machine. From there, the image masks, stored as GeoJSONs, were converted to GeoPandas dataframes,

then rasterized into image masks. Next, I ensured that each source image had a corresponding image mask and saved each image into its respective folder. After that, both the source images and the image masks were converted to arrays, then reshaped to be network-compatible.

Because the disaster imagery came in all manners of shape and size, it required its own set of pre-processing.This consisted of an iterative process through which they were resized, cropped, and magnified until they were compatible with the network.

Modeling:

I opted to utilize a fully convolutional network, due to its ability to preserve the spatial arrangement of input data, which corresponds to spatial dependence in its classifications. While fully convolutional networks closely resemble conventional convolutional neural networks, fully convolutional networks replace the fully connected layers commonly located at the end of a conventional convolutional neural network with 1x1 convolutional layers. Because fully convolutional networks base decisions on local spatial input, they are ideal for image classification problems.

Because most of the work done on image segmentation has been trained using datasets (such as ImageNet) that are incongruous with geospatial data, I decided to train a model from scratch. The specific architecture used within the scope of this project was based off of a Kaggle submission by Kevin Mader. See Appendix 1 for more details about the network architecture.

Results:

Because it was unclear how balanced the pixel-level classes were, I forwent accuracy as my target metric in favor of the dice coefficient, otherwise known as an F1 score. To date, my best model yielded a validation score of 0.7021 and a validation accuracy of 0.9130, utilizing an Adam optimizer, a weighted binary cross-entropy loss function, and 10x10 filters.

In general, the network was very effective at identifying building cores. Instead, it only struggled with predictions along the edge of the image frame as well as with pinpointing building edges. See Appendix B. The model also generalized fairly well to my out-of-sample disaster imagery. See Appendix C. Technology like this has the opportunity to revolutionize the way we survey disaster sites and subsequently respond to them, saving time, resources, and ultimately, lives.

Future Work:

Going forward, there are several avenues of exploration I would like to investigate to improve my model. First, I would like to spend some more time trying out different forms of image augmentation. While I did spend some time trying to make Keras' ImageDataGenerator work with my model, my efforts were fruitless due to continued issues with space and memory. With

more time, there are other methods through which I can augment my images, which would provide my model with not only a larger quantity of data to train on, but also more diverse data.

Next, I may be able to improve my model through more advanced adjustments to the network architecture. For example, I could alter the input and output shapes of the model's hidden layers. I could also spend more time experimenting with the addition and deletion of layers. Alternatively, another option is to try a completely different architecture altogether. For instance, many have had considerable success utilizing fully convolutional U-Nets and fully convolutional ResNets.

Finally, I would like to expand my model beyond binary classification. While the presence and absence of buildings can give us a general sense of how destructive a disaster has been, we would gain a much more complete picture with a model that could identify not only buildings, but also roads, flora, bodies of water, etc.

Appendix A:

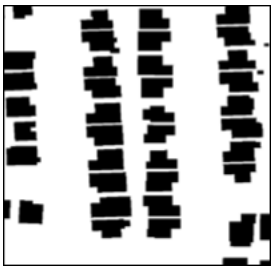
Layer (type)	Output Shape	Param #	Connected to
RGB_Input (InputLayer)	(None, 650, 650, 3)	0	
gaussian_noise_1 (GaussianNoise	(None, 650, 650, 3)	0	RGB_Input[0][0]
batch_normalization_1 (BatchNor	(None, 650, 650, 3)	12	gaussian_noise_1[0][0]
conv2d_1 (Conv2D)	(None, 650, 650, 8)	2400	batch_normalization_1[0][0]
batch_normalization_2 (BatchNor	(None, 650, 650, 8)	32	conv2d_1[0][0]
activation_1 (Activation)	(None, 650, 650, 8)	0	batch_normalization_2[0][0]
conv2d_2 (Conv2D)	(None, 650, 650, 8)	6400	activation_1[0][0]
batch_normalization_3 (BatchNor	(None, 650, 650, 8)	32	conv2d_2[0][0]
activation_2 (Activation)	(None, 650, 650, 8)	0	batch_normalization_3[0][0]
conv2d_3 (Conv2D)	(None, 650, 650, 16)	12800	activation_2[0][0]
batch_normalization_4 (BatchNor	(None, 650, 650, 16)	64	conv2d_3[0][0]
activation_3 (Activation)	(None, 650, 650, 16)	0	batch_normalization_4[0][0]
conv2d_4 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
conv2d_5 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
conv2d_6 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
conv2d_7 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
conv2d_8 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
conv2d_9 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
conv2d_10 (Conv2D)	(None, 650, 650, 16)	25600	activation_3[0][0]
concatenate_1 (Concatenate)	(None, 650, 650, 115	0	batch_normalization_1[0][0] conv2d_4[0][0] conv2d_5[0][0] conv2d_6[0][0] conv2d_7[0][0] conv2d_8[0][0] conv2d_9[0][0] conv2d_10[0][0]
spatial_dropout2d_1 (SpatialDro	(None, 650, 650, 115	0	concatenate_1[0][0]
batch_normalization_5 (BatchNor	(None, 650, 650, 115	460	spatial_dropout2d_1[0][0]
activation_4 (Activation)	(None, 650, 650, 115	0	batch_normalization_5[0][0]
conv2d_11 (Conv2D)	(None, 650, 650, 32)	368000	activation_4[0][0]
batch_normalization_6 (BatchNor	(None, 650, 650, 32)	128	conv2d_11[0][0]
activation_5 (Activation)	(None, 650, 650, 32)	0	batch_normalization_6[0][0]
conv2d_12 (Conv2D)	(None, 650, 650, 1)	33	activation_5[0][0]
cropping2d_1 (Cropping2D)	(None, 618, 618, 1)	0	conv2d_12[0][0]
zero_padding2d_1 (ZeroPadding2D	(None, 650, 650, 1)	0	cropping2d_1[0][0]

Total params: 569,561
Trainable params: 569,197
Non-trainable params: 364

Appendix B:

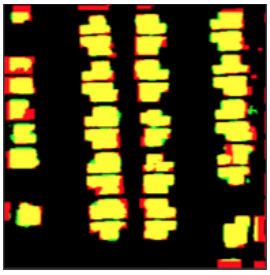
INPUT

SOURCE IMAGE MODEL PREDICTION

OUTPUT

SOURCE IMAGE MODEL PREDICTION

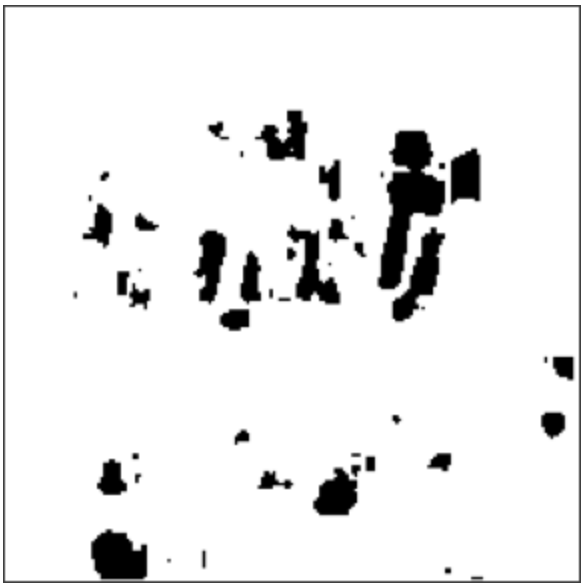



Appendix C:

BEFORE DISASTER

SOURCE IMAGE

MODEL PREDICTION

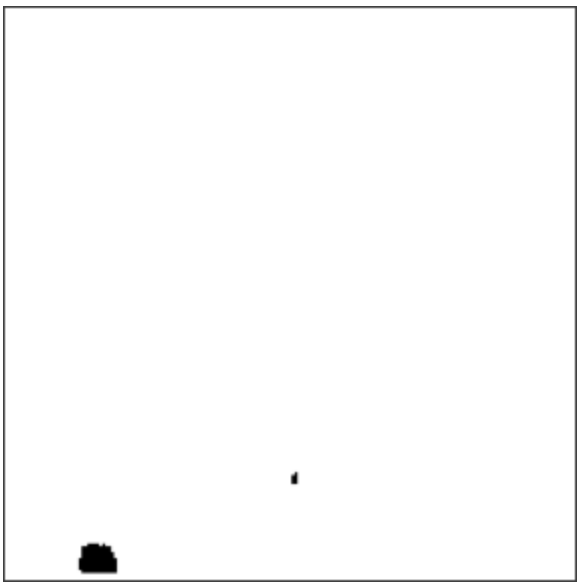


Bidgeneh Missile-Related Facility, Iran
November 3, 2011

AFTER DISASTER

SOURCE IMAGE

MODEL PREDICTION



Bidgeneh Missile-Related Facility, Iran
November 22, 2011