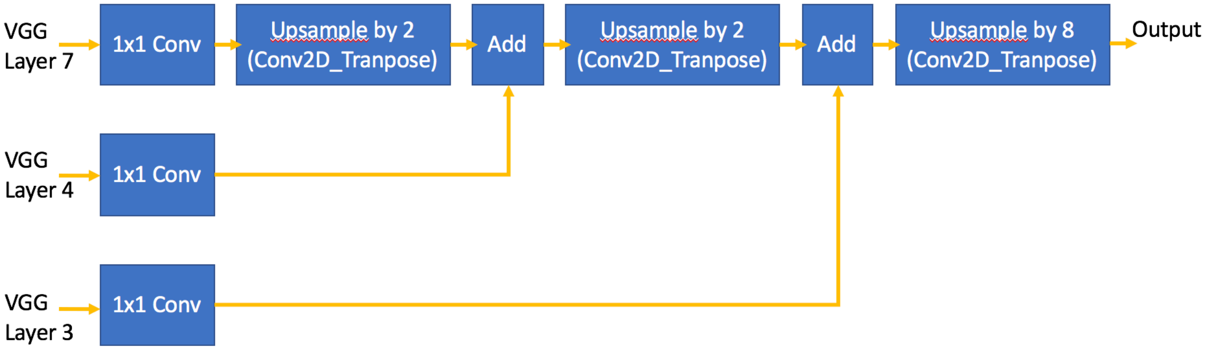
SEMANTIC SEGMENTATION PROJECT

The following sections outline my implementation of the FCN8 neural network architecture as well as examples of my network's segmentation on test images.

NEURAL NETWORK ARCHITECTURE AND TRAINING

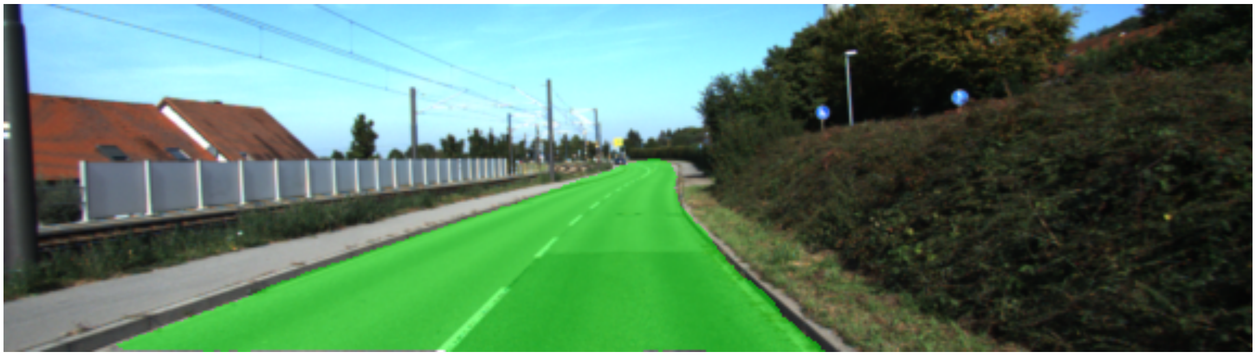
The diagram below illustrates how the outputs of the VGG neural network were processed in order to arrive at a pixel-wise labeling of the input image. Exact details of the neural network layers can be found in the *layers* method in *main.py*. First, all VGG layers are passed through 1x1 Convolutions so that their 3rd dimension (also known as the features or channels dimension) is made equal to the number of target classes (in this project that is 2). Next, the image resulting from processing VGG layer 7 is upsamped by 2 (using transpose convolution) and summed with the image resulting from processing VGG layer 4. Next, that output is upsampled again by a factor 2 and summed with the image resulting from processing VGG layer 3. Finally, the result is upsampled by a factor of 8 to produce the segmented image.

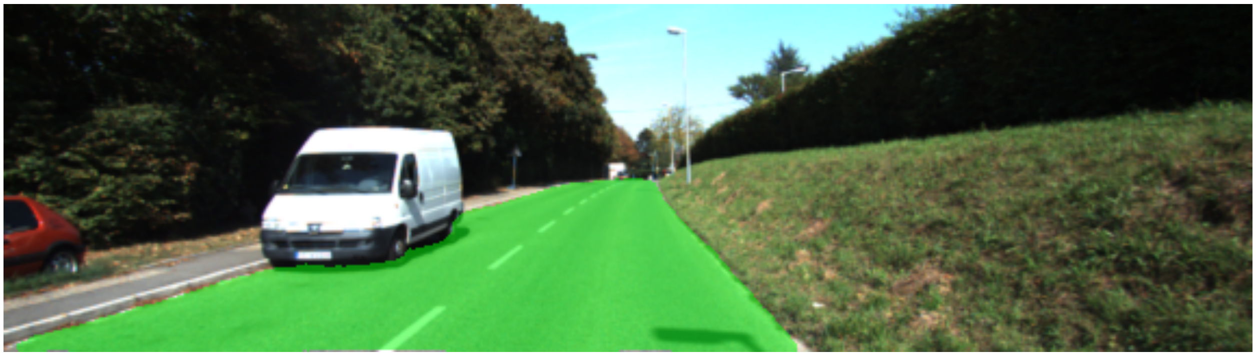

VGG Layer 7
1x1 Conv
Upsample by 2 (Conv2D_Tranpose)
Add
Upsample by 2 (Conv2D_Tranpose)
Add
Upsample by 8 (Conv2D_Tranpose)
Output
VGG Layer 4
1x1 Conv
VGG Layer 3
1x1 Conv

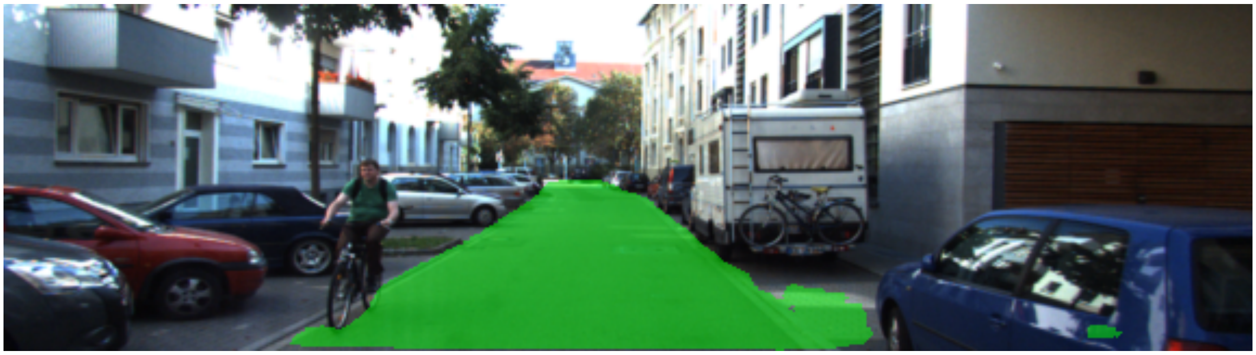
The above architecture was trained for 100 epochs with a batch size of 2. I was limited to using a batch size of 2 due to the memory constraints on the GPU I used (g2.2xlarge instance on Amazon AWS).

NEURAL NETWORK PERFORMANCE

The following three images illustrate examples of excellent segmentations by the trained network.







However, the network did work perfectly on all images. The following images illustrated poor segmentation most likely due to the shadow of the buildings cast on the road.