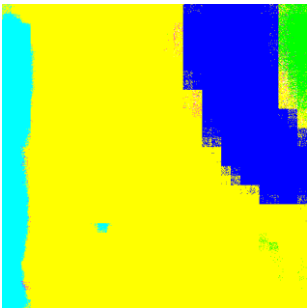
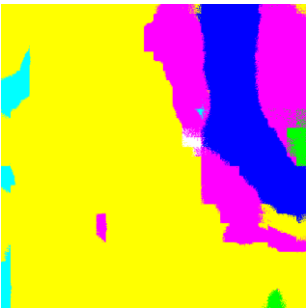

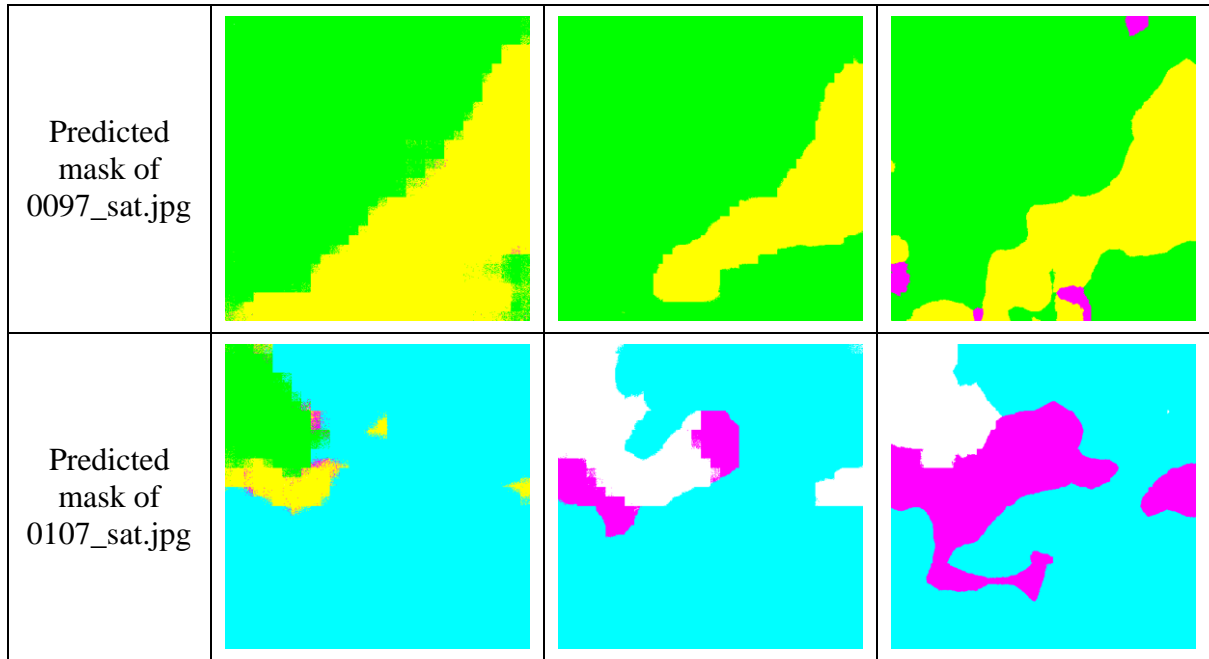


1. (5%) Print the network architecture of your VGG16-FCN32s model.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 512, 512, 3)	0
block1_conv1 (Conv2D)	(None, 512, 512, 64)	1792
block1_conv2 (Conv2D)	(None, 512, 512, 64)	36928
block1_pool (MaxPooling2D)	(None, 256, 256, 64)	0
block2_conv1 (Conv2D)	(None, 256, 256, 128)	73856
block2_conv2 (Conv2D)	(None, 256, 256, 128)	147584
block2_pool (MaxPooling2D)	(None, 128, 128, 128)	0
block3_conv1 (Conv2D)	(None, 128, 128, 256)	295168
block3_conv2 (Conv2D)	(None, 128, 128, 256)	590080
block3_conv3 (Conv2D)	(None, 128, 128, 256)	590080
block3_pool (MaxPooling2D)	(None, 64, 64, 256)	0
block4_conv1 (Conv2D)	(None, 64, 64, 512)	1180160
block4_conv2 (Conv2D)	(None, 64, 64, 512)	2359808
block4_conv3 (Conv2D)	(None, 64, 64, 512)	2359808
block4_pool (MaxPooling2D)	(None, 32, 32, 512)	0
block5_conv1 (Conv2D)	(None, 32, 32, 512)	2359808
block5_conv2 (Conv2D)	(None, 32, 32, 512)	2359808
block5_conv3 (Conv2D)	(None, 32, 32, 512)	2359808
block5_pool (MaxPooling2D)	(None, 16, 16, 512)	0
fully_connected1 (Conv2D)	(None, 16, 16, 4096)	102764544
dropout_1 (Dropout)	(None, 16, 16, 4096)	0
fully_connected2 (Conv2D)	(None, 16, 16, 4096)	16781312
dropout_2 (Dropout)	(None, 16, 16, 4096)	0
fully_connected3 (Conv2D)	(None, 16, 16, 7)	28679
conv2d_transpose_1 (Conv2DTr	(None, 512, 512, 7)	200704
reshape_1 (Reshape)	(None, 262144, 7)	0
activation_1 (Activation)	(None, 262144, 7)	0
Total params: 134,489,927		
Trainable params: 134,489,927		
Non-trainable params: 0		

2. (10%) Show the predicted segmentation mask of validation/0008\_sat.jpg, validation/0097\_sat.jpg, validation/0107\_sat.jpg during the early, middle, and the final stage during the training stage. (For example, results of 1st, 10th, 20th epoch)

Description	Early (3 <sup>rd</sup> epoch)	Middle (11 <sup>th</sup> epoch)	Final (28 <sup>th</sup> epoch)
Predicted mask of 0008_sat.jpg			

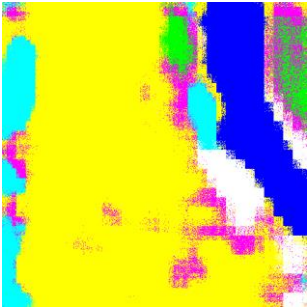
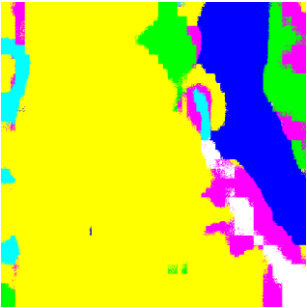

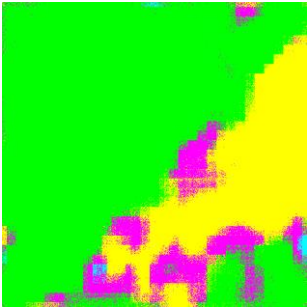
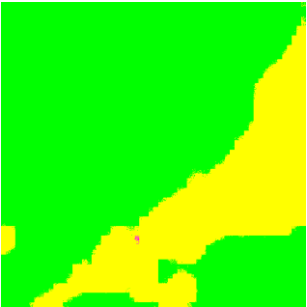

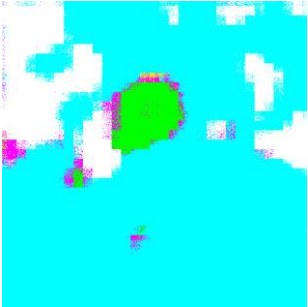
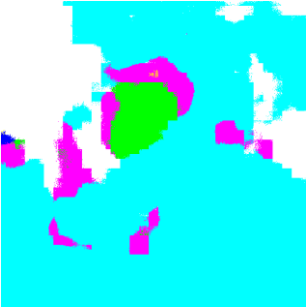



3. (15%) Implement an improved model which performs better than your baseline model. Print the network architecture of this model.

The improved model I implemented is VGG16-FCN16s model.

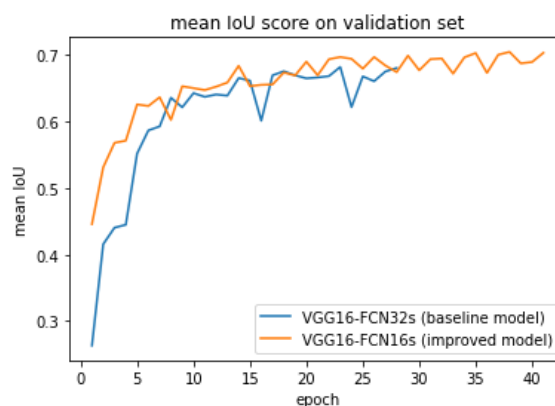
Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 512, 512, 3)	0	
block1_conv1 (Conv2D)	(None, 512, 512, 64)	1792	input_1[0][0]
block1_conv2 (Conv2D)	(None, 512, 512, 64)	36928	block1_conv1[0][0]
block1_pool (MaxPooling2D)	(None, 256, 256, 64)	0	block1_conv2[0][0]
block2_conv1 (Conv2D)	(None, 256, 256, 128)	73856	block1_pool[0][0]
block2_conv2 (Conv2D)	(None, 256, 256, 128)	147584	block2_conv1[0][0]
block2_pool (MaxPooling2D)	(None, 128, 128, 128)	0	block2_conv2[0][0]
block3_conv1 (Conv2D)	(None, 128, 128, 256)	295168	block2_pool[0][0]
block3_conv2 (Conv2D)	(None, 128, 128, 256)	590080	block3_conv1[0][0]
block3_conv3 (Conv2D)	(None, 128, 128, 256)	590080	block3_conv2[0][0]
block3_pool (MaxPooling2D)	(None, 64, 64, 256)	0	block3_conv3[0][0]
block4_conv1 (Conv2D)	(None, 64, 64, 512)	1180160	block3_pool[0][0]
block4_conv2 (Conv2D)	(None, 64, 64, 512)	2359808	block4_conv1[0][0]
block4_conv3 (Conv2D)	(None, 64, 64, 512)	2359808	block4_conv2[0][0]
block4_pool (MaxPooling2D)	(None, 32, 32, 512)	0	block4_conv3[0][0]
block5_conv1 (Conv2D)	(None, 32, 32, 512)	2359808	block4_pool[0][0]
block5_conv2 (Conv2D)	(None, 32, 32, 512)	2359808	block5_conv1[0][0]
block5_conv3 (Conv2D)	(None, 32, 32, 512)	2359808	block5_conv2[0][0]
block5_pool (MaxPooling2D)	(None, 16, 16, 512)	0	block5_conv3[0][0]
fully_conv1 (Conv2D)	(None, 16, 16, 4096)	102764544	block5_pool[0][0]
dropout_1 (Dropout)	(None, 16, 16, 4096)	0	fully_conv1[0][0]
fully_conv2 (Conv2D)	(None, 16, 16, 4096)	16781312	dropout_1[0][0]
dropout_2 (Dropout)	(None, 16, 16, 4096)	0	fully_conv2[0][0]
fully_conv4 (Conv2D)	(None, 16, 16, 7)	28679	dropout_2[0][0]
fully_conv3 (Conv2D)	(None, 32, 32, 7)	3591	block4_pool[0][0]
conv2d_transpose_1 (Conv2DTrans	(None, 32, 32, 7)	784	fully_conv4[0][0]
add_1 (Add)	(None, 32, 32, 7)	0	fully_conv3[0][0] conv2d_transpose_1[0][0]
conv2d_transpose_2 (Conv2DTrans	(None, 512, 512, 7)	50176	add_1[0][0]
reshape_1 (Reshape)	(None, 262144, 7)	0	conv2d_transpose_2[0][0]
activation_1 (Activation)	(None, 262144, 7)	0	reshape_1[0][0]
Total params: 134,343,774			
Trainable params: 134,343,774			
Non-trainable params: 0			

4. (10%) Show the predicted segmentation mask of validation/0008\_sat.jpg, validation/0097\_sat.jpg, validation/0107\_sat.jpg during the early, middle, and the final stage during the training process of this improved model.

Description	Early (3 <sup>rd</sup> epoch)	Middle (11 <sup>th</sup> epoch)	Final (41 <sup>st</sup> epoch)
Predicted mask of 0008_sat.jpg			
Predicted mask of 0097_sat.jpg			
Predicted mask of 0107_sat.jpg			

5. (15%) Report mIoU score of both models on the validation set. Discuss the reason why the improved model performs better than the baseline one. You may conduct some experiments and show some evidences to support your discussion.

Here is the plot that demonstrates the mIoU scores of both models at every epoch on the validation set.



In the training process, the cross entropy loss of the baseline model (VGG16-FCN32s) converges at about the 28<sup>th</sup> epoch, and the mean IoU score reaches at 0.680566.

On the other hand, the cross entropy loss of the improved model (VGG16-FCN16s) converges approximately at the 41<sup>st</sup> epoch, and the mean IoU score achieves 0.703396.

The improved model I implemented is VGG16-FCN16s, which combines the predictions from the 4<sup>th</sup> max pooling layer and the last convolutional layer, containing both coarse and fine information. As a consequence, comparing to the single stream VGG16-FCN32s model, it is reasonable to obtain a better performance on the improved VGG-FCN16s model than the baseline model.

6. (5%) [bonus] Calculate the result of  $d/dw$   $G(w)$ :

**objective function:**

$$G(\mathbf{w}) = -\sum_n [t^{(n)} \log x(\mathbf{z}^{(n)}; \mathbf{w}) + (1 - t^{(n)}) \log (1 - x(\mathbf{z}^{(n)}; \mathbf{w}))] \geq 0$$

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} G(\mathbf{w}) \quad \text{choose the weights that minimise the network's surprise about the training data}$$

$$\frac{d}{d\mathbf{w}} G(\mathbf{w}) = \sum_n \frac{dG(\mathbf{w})}{dx^{(n)}} \frac{dx^{(n)}}{d\mathbf{w}} = -\sum_n (t^{(n)} - x^{(n)}) \mathbf{z}^{(n)} = \text{prediction error} \times \text{feature}$$

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \frac{d}{d\mathbf{w}} G(\mathbf{w}) \quad \text{iteratively step down the objective (gradient points up hill)} \quad 39$$