Please use this report template, and upload it in the PDF format. Reports in other forms/formats will result in ZERO point. Reports written in either Chinese or English is acceptable. The length of your report should NOT exceed 6 pages (**excluding bonus**).

Name: 郭笛萱 Dep.:電機四 Student ID:b03901009

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

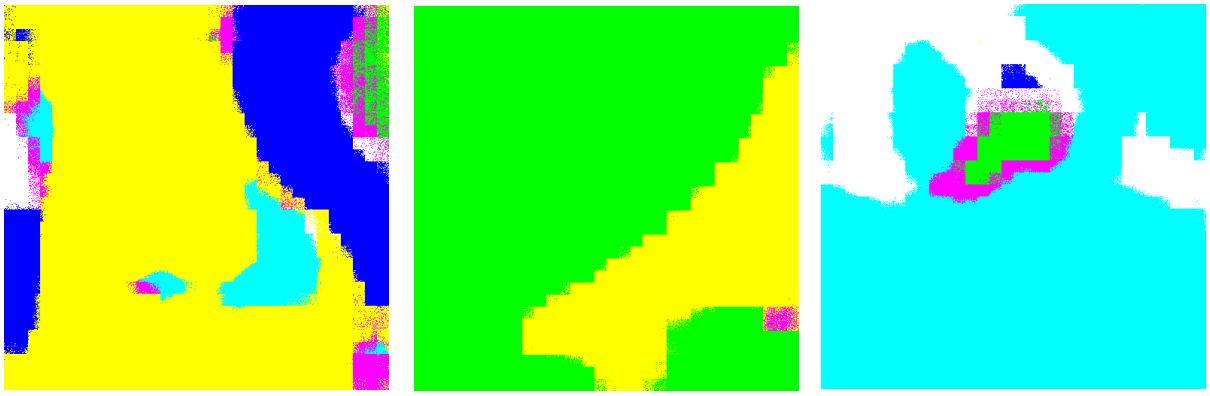
Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 512, 512, 3)	0
block1_conv1 (Conv2D)	(None, 512, 512, 64)	1792
block1_conv2 (Conv2D)	(None, 512, 512, 64)	36928
block1_pool (MaxPooling2D)	(None, 256, 256, 64)	0
block2_conv1 (Conv2D)	(None, 256, 256, 128)	73856
block2_conv2 (Conv2D)	(None, 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
block6_conv1 (Conv2D)	(None, 16, 16, 4096)	102764544
block6_conv2 (Conv2D)	(None, 16, 16, 4096)	16781312
block6_conv3 (Conv2D)	(None, 16, 16, 7)	28679
conv2d_transpose_1 (Conv2DTr	(None, 512, 512, 7)	200704
activation_1 (Activation)	(None, 512, 512, 7)	0

Total params: 134,489,927
Trainable params: 119,775,239
Non-trainable params: 14,714,688

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)

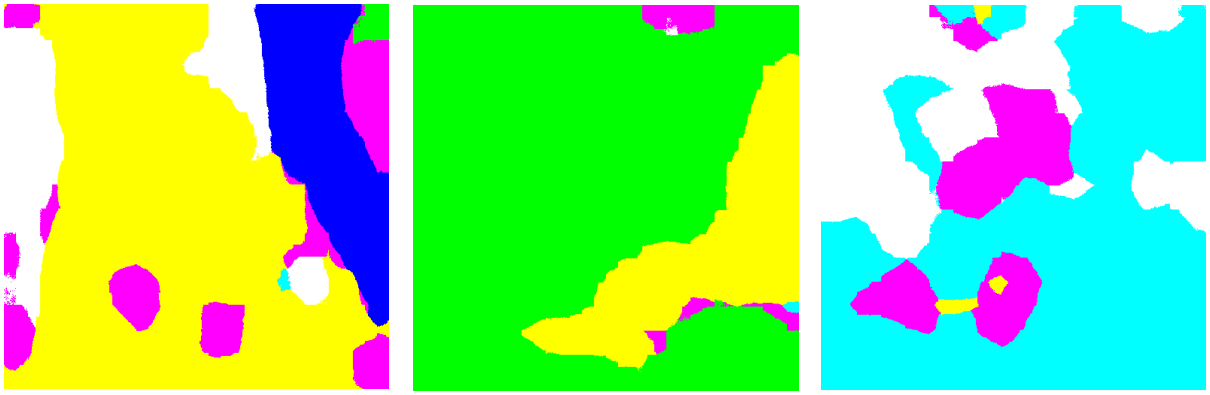
Early stage: epoch = 2

0008_sat.jpg 0097_sat.jpg 0107_sat.jpg



Middle stage: epoch = 10

0008_sat.jpg 0097_sat.jpg 0107_sat.jpg



Final stage: epoch = 21

0008_sat.jpg 0097_sat.jpg 0107_sat.jpg

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

FCN-8 Model:
https://github.com/divamgupta/image-segmentation-keras/blob/master/Models/FCN8.py

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]
conv2d_1 (Conv2D)	(None, 16, 16, 4096)	102764544	block5_pool[0][0]
dropout_1 (Dropout)	(None, 16, 16, 4096)	0	conv2d_1[0][0]
conv2d_2 (Conv2D)	(None, 16, 16, 4096)	16781312	dropout_1[0][0]
dropout_2 (Dropout)	(None, 16, 16, 4096)	0	conv2d_2[0][0]

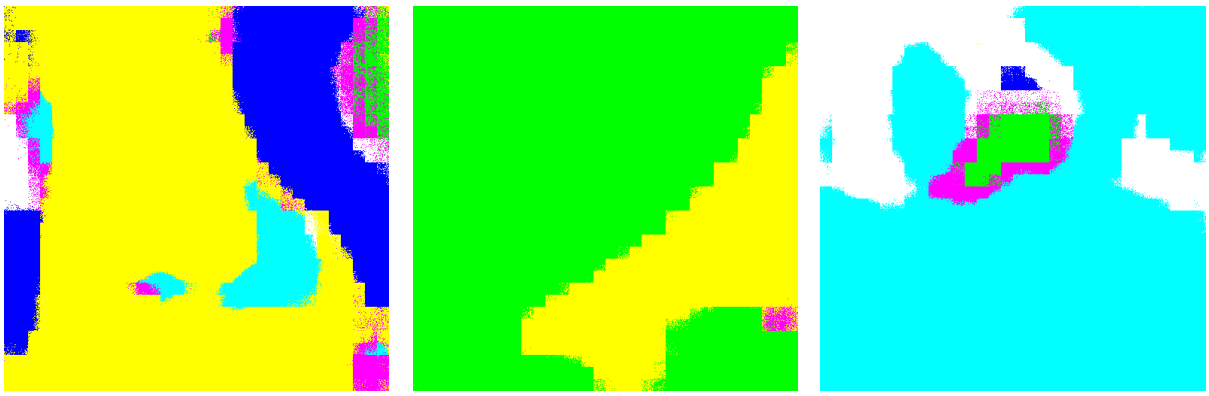
conv2d_4 (Conv2D)	(None, 32, 32, 7)	3591	block4_pool[0][0]
conv2d_3 (Conv2D)	(None, 16, 16, 7)	28679	dropout_2[0][0]
cropping2d_1 (Cropping2D)	(None, 32, 32, 7)	0	conv2d_4[0][0]
conv2d_transpose_1 (Conv2DTrans	(None, 32, 32, 7)	784	conv2d_3[0][0]
cropping2d_2 (Cropping2D)	(None, 32, 32, 7)	0	cropping2d_1[0][0]
add_1 (Add)	(None, 32, 32, 7)	0	conv2d_transpose_1[0][0] cropping2d_2[0][0]
conv2d_transpose_2 (Conv2DTrans	(None, 64, 64, 7)	784	add_1[0][0]
cropping2d_3 (Cropping2D)	(None, 64, 64, 7)	0	conv2d_transpose_2[0][0]
conv2d_5 (Conv2D)	(None, 64, 64, 7)	1799	block3_pool[0][0]
cropping2d_4 (Cropping2D)	(None, 64, 64, 7)	0	cropping2d_3[0][0]
add_2 (Add)	(None, 64, 64, 7)	0	conv2d_5[0][0] cropping2d_4[0][0]
conv2d_transpose_3 (Conv2DTrans	(None, 512, 512, 7)	12544	add_2[0][0]

Total params: 134,308,725
Trainable params: 119,594,037
Non-trainable params: 14,714,688

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.

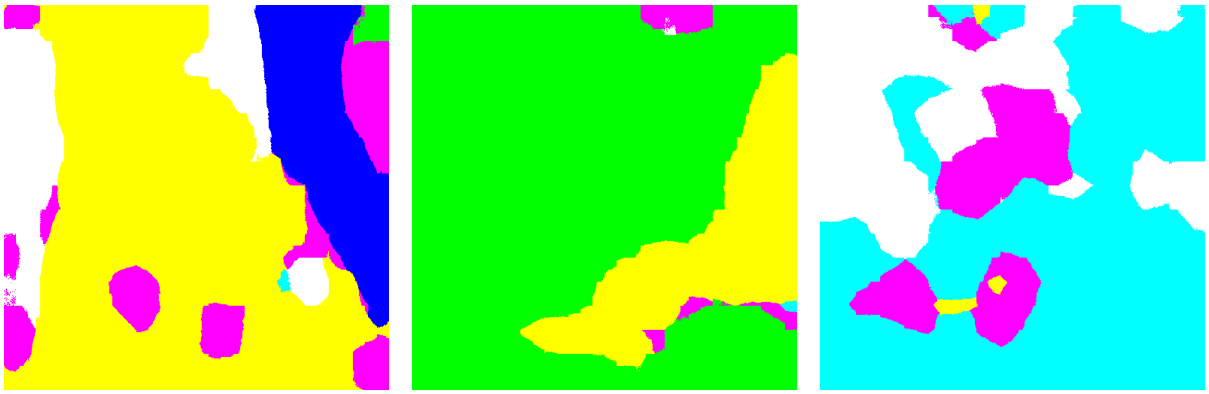
Early stage: epoch = 2

0008_sat.jpg 0097_sat.jpg 0107_sat.jpg



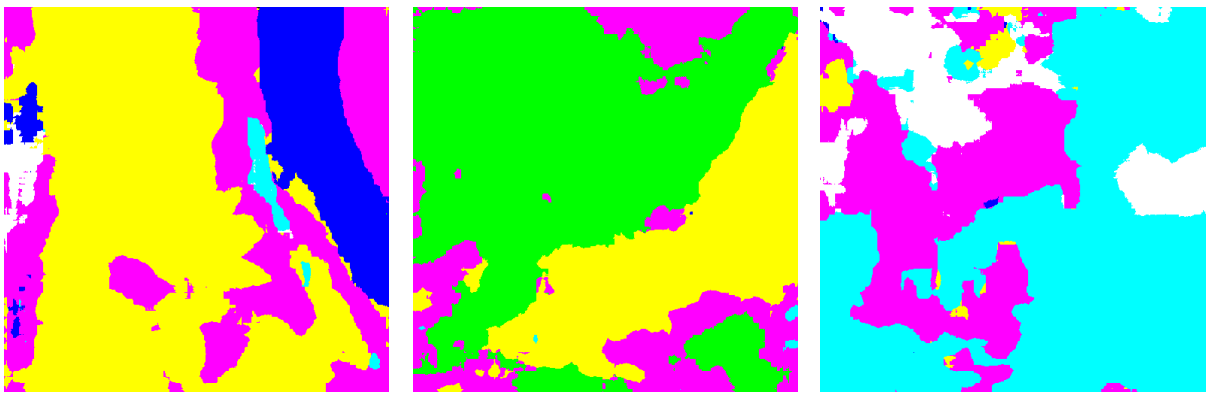
Middle stage: epoch = 10

0008_sat.jpg 0097_sat.jpg 0107_sat.jpz



Final stage: epoch = 45

0008_sat.jpg 0097_sat.jpg 0107_sat.jpg



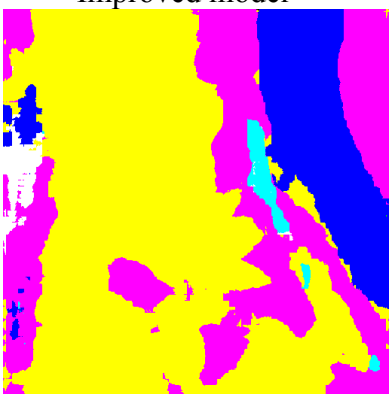
5. (15%) Report mIoU score of both models on the validation set. Discuss the reason why the improved model performs better than the baseline one. You may conduct some experiments and show some evidences to support your discussion.

   For baseline model, mIoU = 0.6436 on validation set (epoch = 21),
   While for improved model, mIoU = 0.6577 on validation set (epoch = 45)

   Fcn8 combines the information from pool3, pool4 and the final layer together, which may keep large scale imformation & details at the same time, so the model can predict finer details.
   Fcn32 only upsamples the final layer, whose size is only 16 * 16, therefore, lots of information is lost after passing many pooling layers.
   As you can see, the result of improved model 0008_sat.jpg got more details than the result of baseline model

Baseline model

Improved model



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

objective function:

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

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

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{w}} G(\boldsymbol{w}) = \sum_n \frac{\mathrm{d}G(\boldsymbol{w})}{\mathrm{d}x^{(n)}} \frac{\mathrm{d}x^{(n)}}{\mathrm{d}\boldsymbol{w}} = -\sum_n (t^{(n)} - x^{(n)}) \boldsymbol{z}^{(n)}$$ = prediction error x feature

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

$$\frac{d}{dw} G(w) = \sum_n \frac{dG(w)}{dx^{(n)}} \frac{dx^{(n)}}{dw} = -\sum_n (t^{(n)} - x^{(n)}) z^{(n)}$$

$$\frac{dG(w)}{dx^n} = \frac{t^{(n)}}{\chi(z^{(n)}, w)} - \frac{(1-t^{(n)})}{1-\chi(z^{(n)}, w)} = \frac{t^{(n)} - \chi(z^{(n)}, w)}{\chi(z^{(n)}, w) \cdot (1-\chi(z^{(n)}, w))}$$

$$\chi(z, w) = \frac{1}{1 + e^{-(z^{(n)}, w)}}$$

$$\frac{dx^{(n)}}{dw} = x^{(n)} \cdot (1-x^{(n)}) \cdot \frac{d(z^{(n)} w)}{dw} = x^{(n)} \cdot (1-x^{(n)}) \cdot z^{(n)}$$

$$\Rightarrow \frac{d}{dw} G(w) = \sum_n \frac{t^{(n)} - \chi(z^{(n)}, w)}{\chi(z^{(n)}, w) \cdot (1-\chi(z^{(n)}, w))} \cdot x^n \cdot (1-x^{(n)}) \cdot z^{(n)}$$

$$= -\sum_n (t^{(n)} - x^{(n)}) z^{(n)}$$