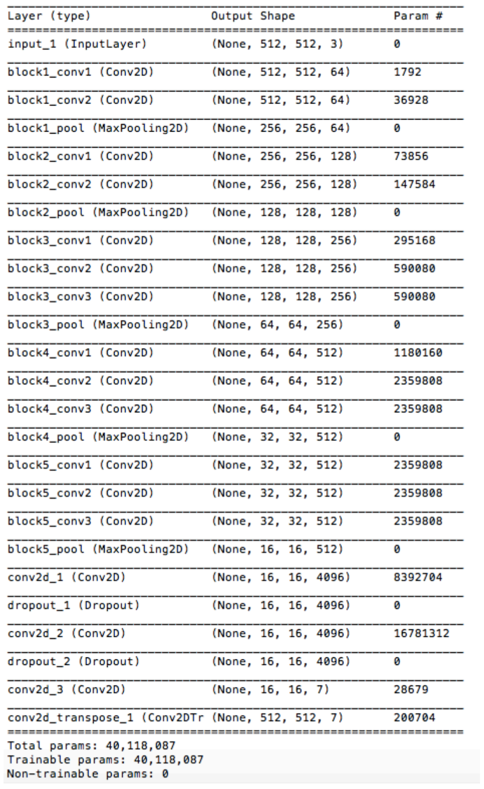
Please use this report template, and upload it in the PDF format. Reports in other forms/formats will result in ZERO point. Reports written in either Chinese or English is acceptable. The length of your report should NOT exceed 6 pages (**excluding bonus**).

Name: 黃聖喻　Dep.:電機三　Student ID:B04901073

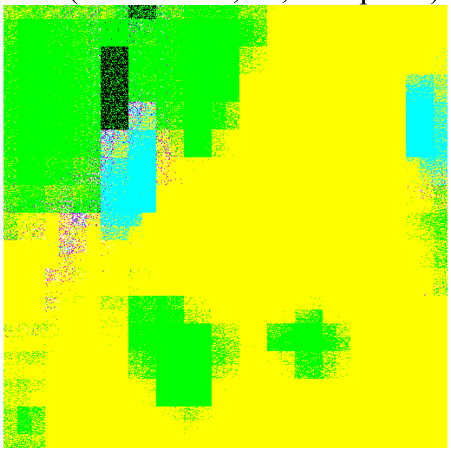
1. (5%) Print the network architecture of your VGG16-FCN32s model.

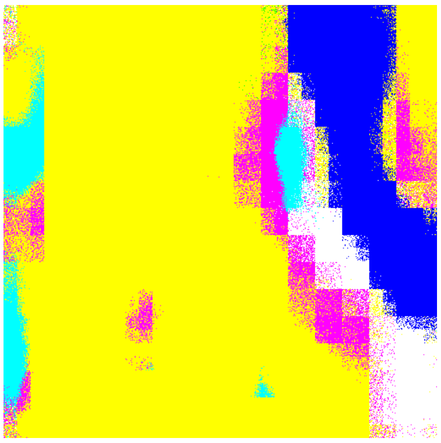


```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 512, 512, 3)       0
_________________________________________________________________
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
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 16, 4096)      8392704
_________________________________________________________________
dropout_1 (Dropout)          (None, 16, 16, 4096)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 4096)      16781312
_________________________________________________________________
dropout_2 (Dropout)          (None, 16, 16, 4096)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 7)         28679
_________________________________________________________________
conv2d_transpose_1 (Conv2DTr (None, 512, 512, 7)       200704
=================================================================
Total params: 40,118,087
Trainable params: 40,118,087
Non-trainable params: 0
```

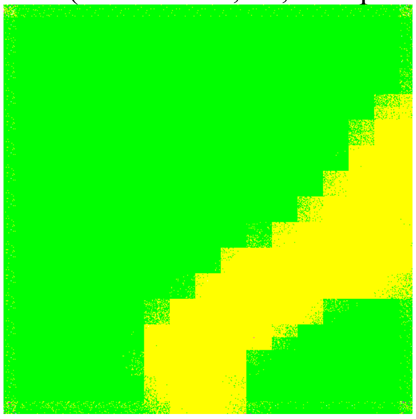
2. (10%) Show the predicted segmentation mask of validation/0008_sat.jpg, validation/0097_sat.jpg, validation/0107_sat.jpg during the early, middle, and the final stage during the training stage. (For example, results of 1st, 10th, 20th epoch)

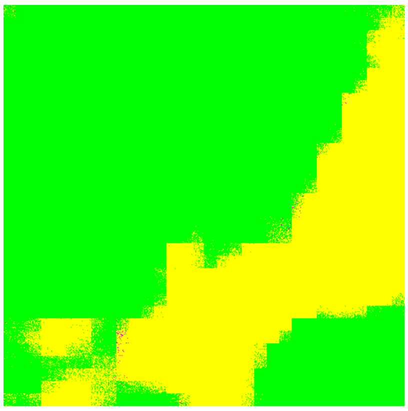
0008(in order 2nd ,6th ,12th epoch)

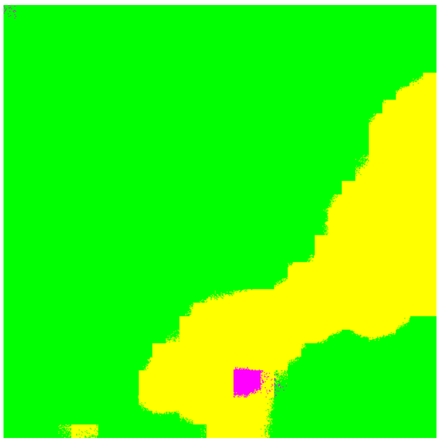




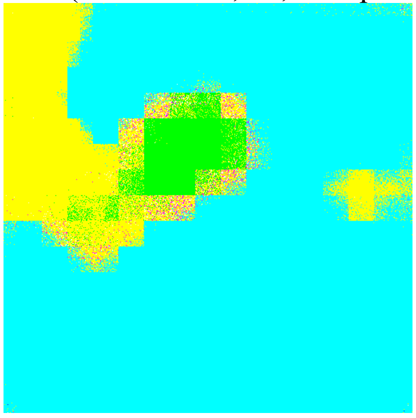
0097(in order 2nd ,6th ,12th epoch)

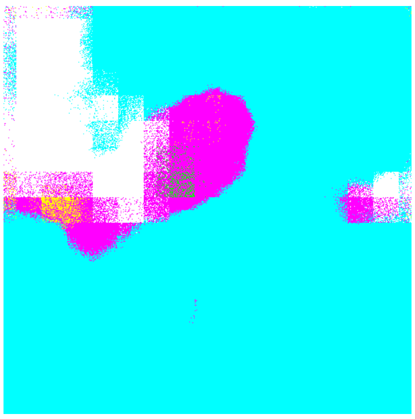


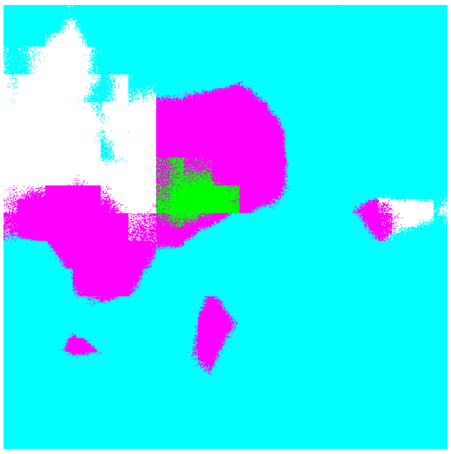




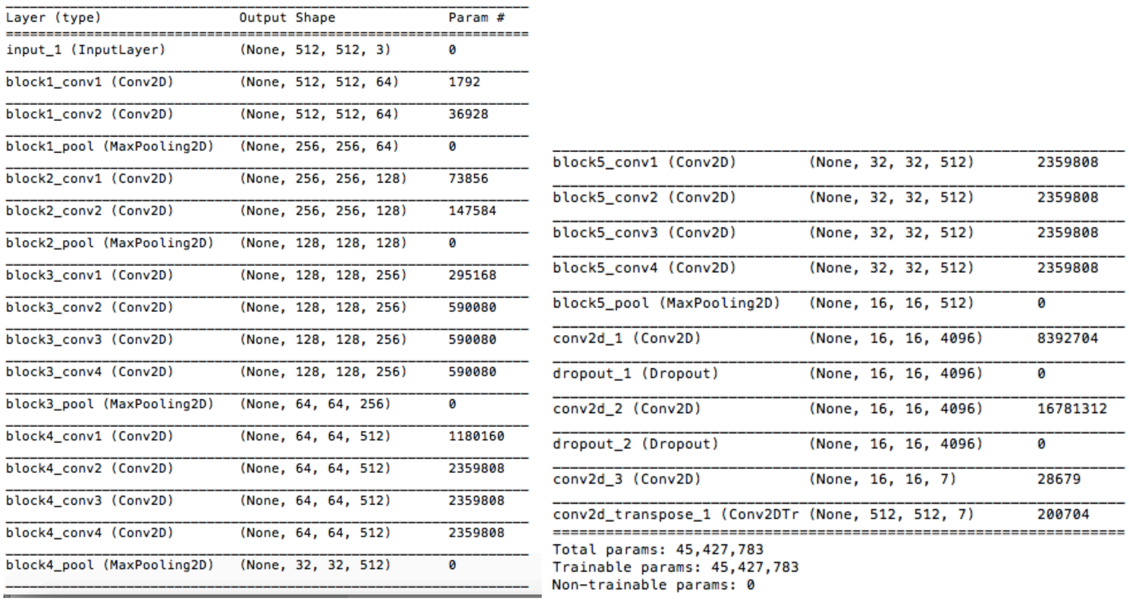
0107(in order 2nd ,6th ,12th epoch)







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

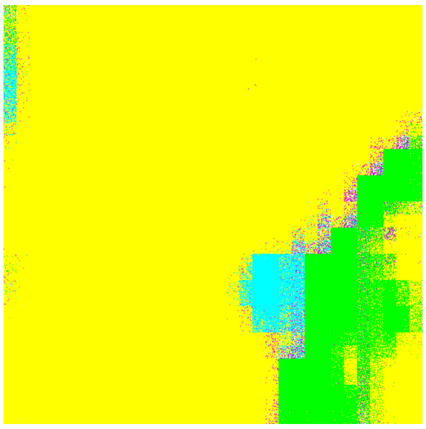
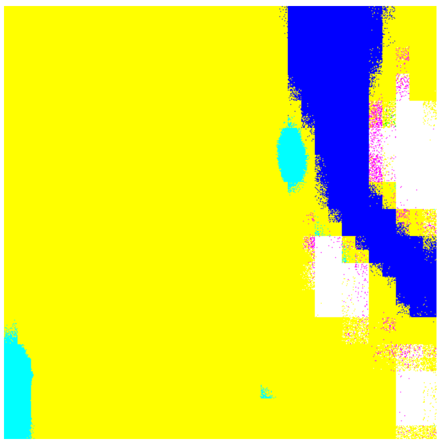


```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 512, 512, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 512, 512, 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_conv4 (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_conv4 (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_conv4 (Conv2D)        (None, 32, 32, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 16, 16, 512)       0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 16, 16, 4096)      8392704
_________________________________________________________________
dropout_1 (Dropout)          (None, 16, 16, 4096)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 16, 16, 4096)      16781312
_________________________________________________________________
dropout_2 (Dropout)          (None, 16, 16, 4096)      0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 7)         28679
_________________________________________________________________
conv2d_transpose_1 (Conv2DTr (None, 512, 512, 7)       200704
=================================================================
Total params: 45,427,783
Trainable params: 45,427,783
Non-trainable params: 0
```

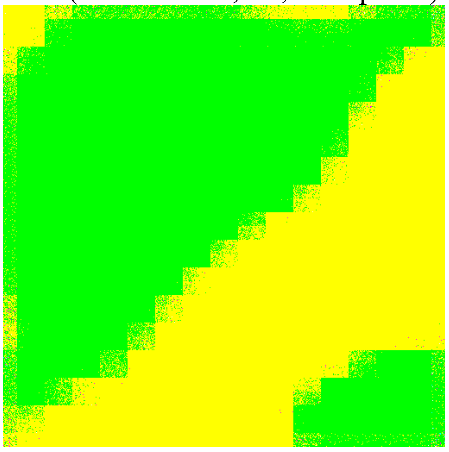
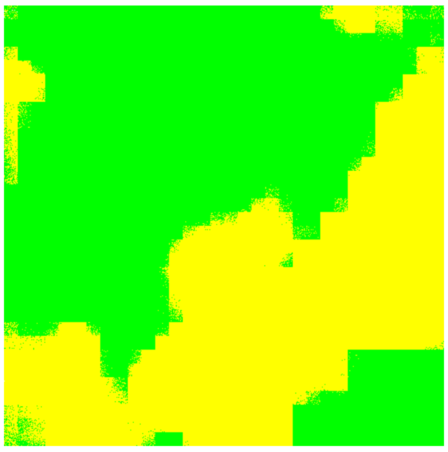
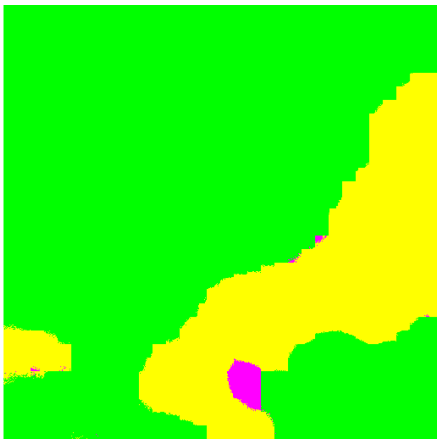
what I used was a vgg19-fcn32 structure, which is three more layer than vgg16

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.

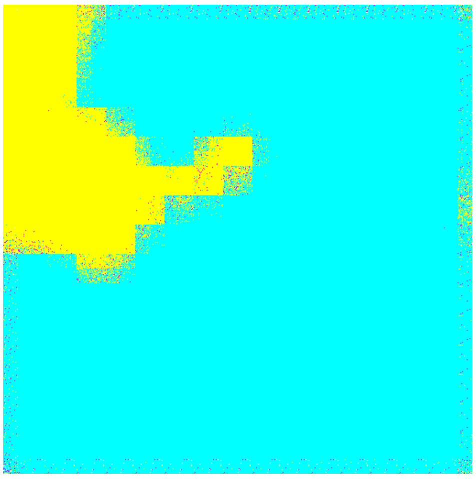
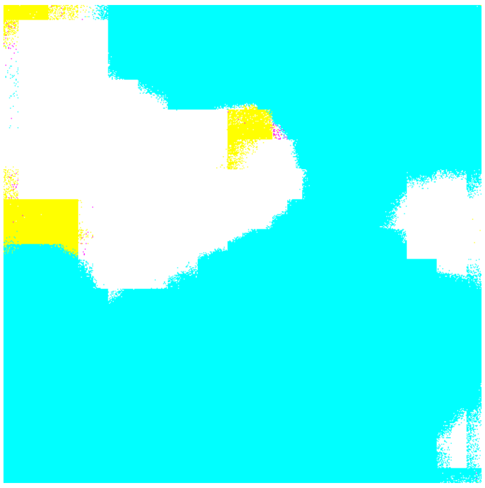
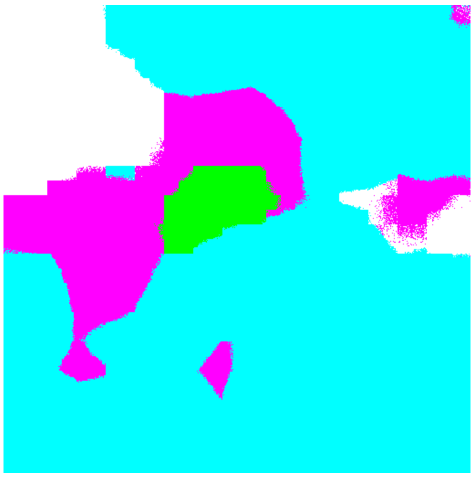
0008(in order 2nd ,8th ,18th epoch)

0097(in order 2nd ,8th ,18th epoch)

0107(in order 2nd ,8th ,18th epoch)

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.

Base:0.679854
Improved:0.680081

我使用 vgg19 的模型做分類，但是實際上表現並沒有好很多，而且我試著調整 batch size 還有 regularizer 的大小做過各種嘗試，最好仍舊只停留在 0.68，因此就結果論，或許選擇 vgg19 並不是一個太恰當的改進，因為雖然多了三層 conv2D 可能可以多學到一點特徵，然而時間的運行上比 vgg16 多了約 20%(我自己 vgg16 一次要四個小時左右，而 vgg19 則快要五個小時)，就效益上來說就不是那麼好了，我有問過其他同學，有的人使用 vgg16-fcn16 或 vgg16-fcn8 就有更大幅的增進。即便如此，在多次的 training 當中還是有看出一些結論，譬如說一開始會想要使用較大的 batch size(16,20,24)結果反而出現嚴重的 overfitting，mIOU 也只有 0.60~0.65 之間，最後的最佳結果是使用 batch size=12 的模型，因此在有限的 training data 下不宜使用太高的 batch size。

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)