Name: 李漪莛　Dep.: 物理四　Student ID: B03202017

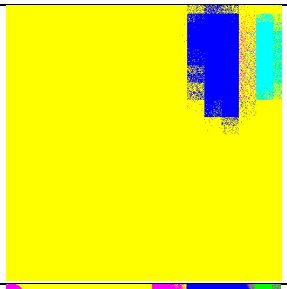
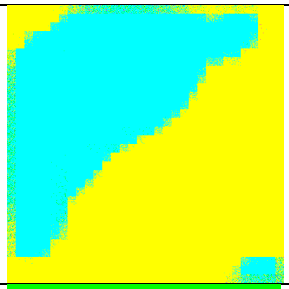
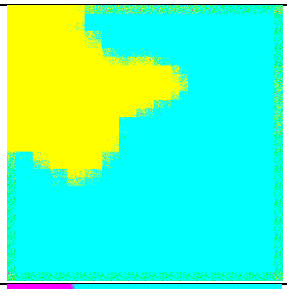
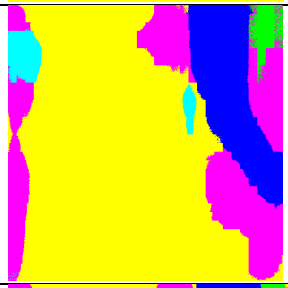
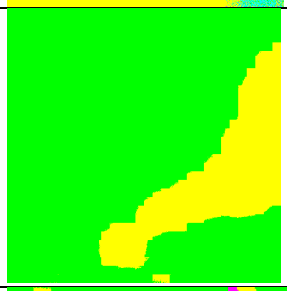
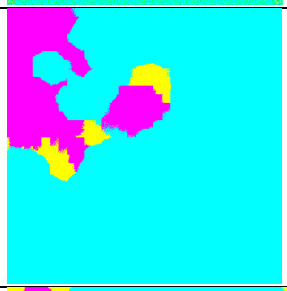
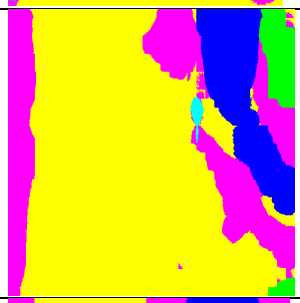
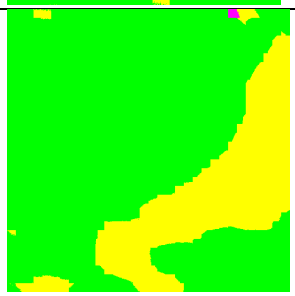
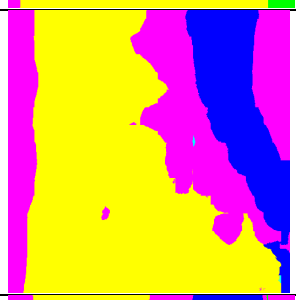
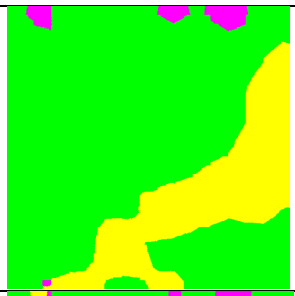
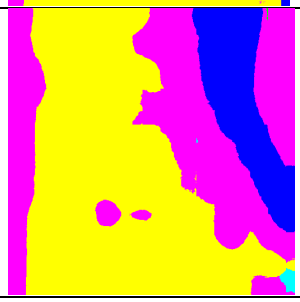
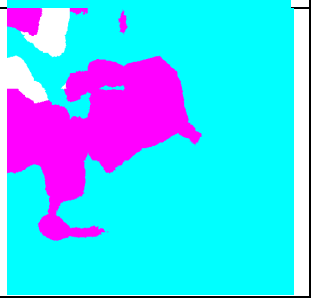
1. (5%) Print the network architecture of your VGG16-FCN32s model.

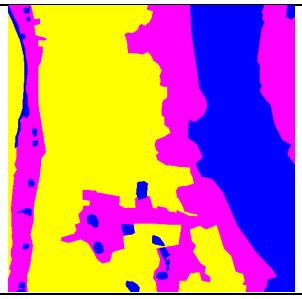
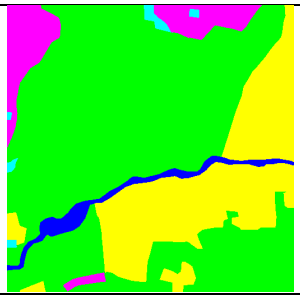
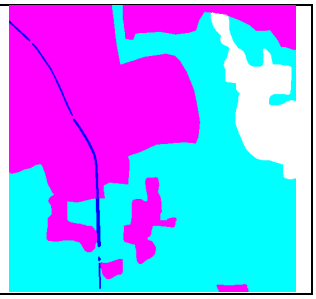
| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_1 (InputLayer) | (None, 512, 512, 3) | 0 |
| zero_padding2d_1 (ZeroPaddin | (None, 712, 712, 3) | 0 |
| block1_conv1 (Conv2D) | (None, 710, 710, 64) | 1792 |
| block1_conv2 (Conv2D) | (None, 710, 710, 64) | 36928 |
| block1_pool (MaxPooling2D) | (None, 355, 355, 64) | 0 |
| block2_conv1 (Conv2D) | (None, 355, 355, 128) | 73856 |
| block2_conv2 (Conv2D) | (None, 355, 355, 128) | 147584 |
| block2_pool (MaxPooling2D) | (None, 177, 177, 128) | 0 |
| block3_conv1 (Conv2D) | (None, 177, 177, 256) | 295168 |
| block3_conv2 (Conv2D) | (None, 177, 177, 256) | 590080 |
| block3_conv3 (Conv2D) | (None, 177, 177, 256) | 590080 |
| block3_pool (MaxPooling2D) | (None, 88, 88, 256) | 0 |
| block4_conv1 (Conv2D) | (None, 88, 88, 512) | 1180160 |
| block4_conv2 (Conv2D) | (None, 88, 88, 512) | 2359808 |
| block4_conv3 (Conv2D) | (None, 88, 88, 512) | 2359808 |
| block4_pool (MaxPooling2D) | (None, 44, 44, 512) | 0 |
| block5_conv1 (Conv2D) | (None, 44, 44, 512) | 2359808 |
| block5_conv2 (Conv2D) | (None, 44, 44, 512) | 2359808 |
| block5_conv3 (Conv2D) | (None, 44, 44, 512) | 2359808 |
| block5_pool (MaxPooling2D) | (None, 22, 22, 512) | 0 |
| FCN_block6_conv1 (Conv2D) | (None, 16, 16, 2048) | 51382272 |
| dropout_1 (Dropout) | (None, 16, 16, 2048) | 0 |
| FCN_block6_conv2 (Conv2D) | (None, 16, 16, 2048) | 4196352 |
| dropout_2 (Dropout) | (None, 16, 16, 2048) | 0 |
| FCN_block6_conv3 (Conv2D) | (None, 16, 16, 7) | 14343 |
| FCN_block7 (Conv2DTranspose) | (None, 544, 544, 7) | 200704 |
| cropping2d_1 (Cropping2D) | (None, 512, 512, 7) | 0 |
| reshape_1 (Reshape) | (None, 262144, 7) | 0 |
| activation_1 (Activation) | (None, 262144, 7) | 0 |

Total params: 70,508,359
Trainable params: 70,508,359
Non-trainable params: 0

After training, there is a layer to reshape the output into (512, 512, 7), get argmax and change the label to rgb color. The shape of output = (None, 512, 512, 3)

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)
備註：我跑了 40 個 epochs，放了不同的五個階段的圖片。

| VGG16_FCN32 | Valid/0008 | Valid/0097 | Valid/0107 |
|---|---|---|---|
| Epoch=1<br>early |  |  |  |
| Epoch=10 |  |  |  |
| Epoch=20<br>middle |  |  |  |
| Epoch=30 |  |  |  |
| Epoch=40<br>final |  |  |  |

| Ground Truth |  |  |  |
|---|---|---|---|

3. (15%) Implement an improved model which performs better than your baseline model. Print the network architecture of this model.
   我 implement VGG16-FCN16。

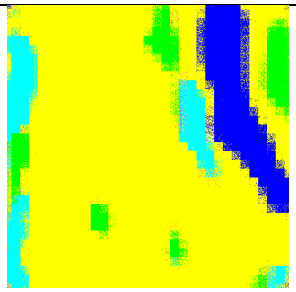
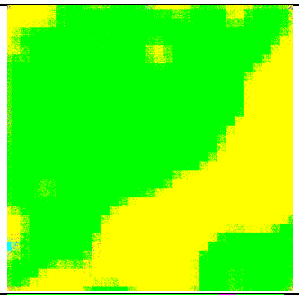
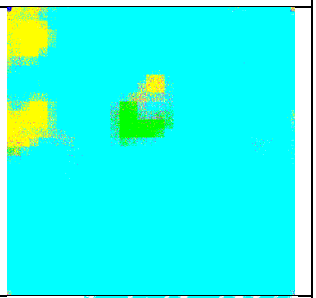
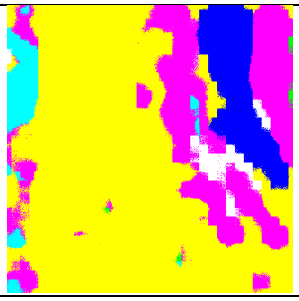
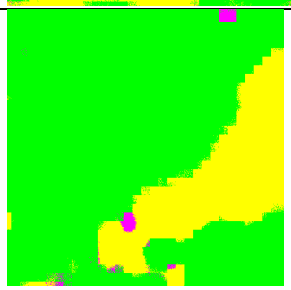
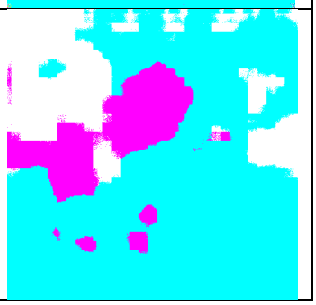
| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_1 (InputLayer) | (None, 512, 512, 3) | 0 | |
| zero_padding2d_1 (ZeroPadding2D | (None, 712, 712, 3) | 0 | input_1[0][0] |
| block1_conv1 (Conv2D) | (None, 710, 710, 64) | 1792 | zero_padding2d_1[0][0] |
| block1_conv2 (Conv2D) | (None, 710, 710, 64) | 36928 | block1_conv1[0][0] |
| block1_pool (MaxPooling2D) | (None, 355, 355, 64) | 0 | block1_conv2[0][0] |
| block2_conv1 (Conv2D) | (None, 355, 355, 128 | 73856 | block1_pool[0][0] |
| block2_conv2 (Conv2D) | (None, 355, 355, 128 | 147584 | block2_conv1[0][0] |
| block2_pool (MaxPooling2D) | (None, 177, 177, 128 | 0 | block2_conv2[0][0] |
| block3_conv1 (Conv2D) | (None, 177, 177, 256 | 295168 | block2_pool[0][0] |
| block3_conv2 (Conv2D) | (None, 177, 177, 256 | 590080 | block3_conv1[0][0] |
| block3_conv3 (Conv2D) | (None, 177, 177, 256 | 590080 | block3_conv2[0][0] |
| block3_pool (MaxPooling2D) | (None, 88, 88, 256) | 0 | block3_conv3[0][0] |
| block4_conv1 (Conv2D) | (None, 88, 88, 512) | 1180160 | block3_pool[0][0] |
| block4_conv2 (Conv2D) | (None, 88, 88, 512) | 2359808 | block4_conv1[0][0] |
| block4_conv3 (Conv2D) | (None, 88, 88, 512) | 2359808 | block4_conv2[0][0] |
| block4_pool (MaxPooling2D) | (None, 44, 44, 512) | 0 | block4_conv3[0][0] |
| block5_conv1 (Conv2D) | (None, 44, 44, 512) | 2359808 | block4_pool[0][0] |
| block5_conv2 (Conv2D) | (None, 44, 44, 512) | 2359808 | block5_conv1[0][0] |
| block5_conv3 (Conv2D) | (None, 44, 44, 512) | 2359808 | block5_conv2[0][0] |
| block5_pool (MaxPooling2D) | (None, 22, 22, 512) | 0 | block5_conv3[0][0] |

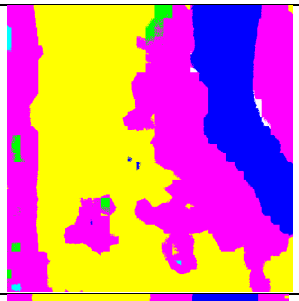
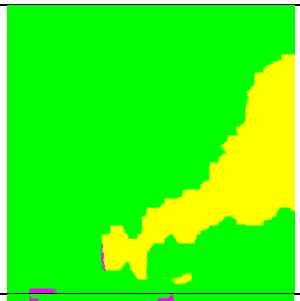
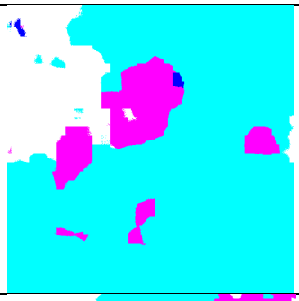
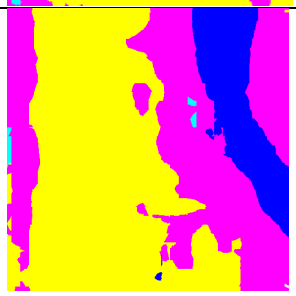
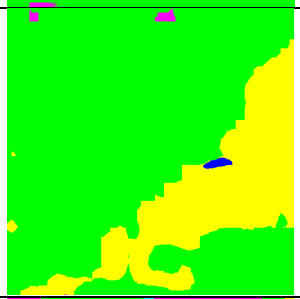
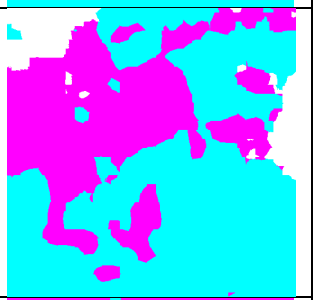
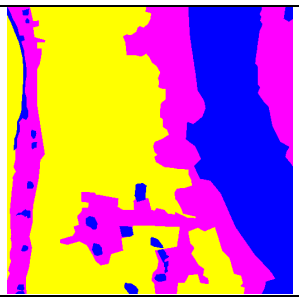
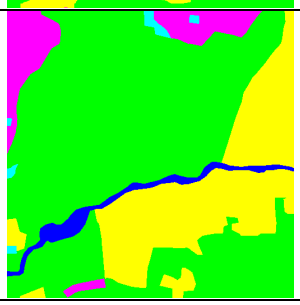
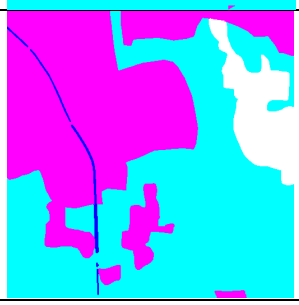
| | | | |
|---|---|---|---|
| FCN_block6_conv1 (Conv2D) | (None, 16, 16, 2048) | 51382272 | block5_pool[0][0] |
| dropout_1 (Dropout) | (None, 16, 16, 2048) | 0 | FCN_block6_conv1[0][0] |
| FCN_block6_conv2 (Conv2D) | (None, 16, 16, 2048) | 4196352 | dropout_1[0][0] |
| dropout_2 (Dropout) | (None, 16, 16, 2048) | 0 | FCN_block6_conv2[0][0] |
| FCN_block6_conv3 (Conv2D) | (None, 16, 16, 7) | 14343 | dropout_2[0][0] |
| conv2d_1 (Conv2D) | (None, 44, 44, 7) | 3591 | block4_pool[0][0] |
| FCN_block7 (Conv2DTranspose) | (None, 34, 34, 7) | 784 | FCN_block6_conv3[0][0] |
| cropping2d_1 (Cropping2D) | (None, 34, 34, 7) | 0 | conv2d_1[0][0] |
| add_1 (Add) | (None, 34, 34, 7) | 0 | FCN_block7[0][0]<br>cropping2d_1[0][0] |
| conv2d_transpose_1 (Conv2DTrans | (None, 560, 560, 7) | 50176 | add_1[0][0] |
| cropping2d_2 (Cropping2D) | (None, 512, 512, 7) | 0 | conv2d_transpose_1[0][0] |
| reshape_1 (Reshape) | (None, 262144, 7) | 0 | cropping2d_2[0][0] |
| activation_1 (Activation) | (None, 262144, 7) | 0 | reshape_1[0][0] |

Total params: 70,362,206
Trainable params: 70,362,206
Non-trainable params: 0

After training, there is a layer to reshape the output into (512, 512, 7), get argmax and change the label to rgb color. The shape of output = (None, 512, 512, 3)

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.

| VGG16_FCN16 | Valid/0008 | Valid/0097 | Valid/0107 |
|---|---|---|---|
| Epoch=1<br>early |  |  |  |
| Epoch=10 |  |  |  |

| | | | |
|---|---|---|---|
| Epoch=20<br>middle |  |  |  |
| Epoch=40<br>final |  |  |  |
| Ground Truth |  |  |  |

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.

| Score of VGG16-FCN32: | Score of VGG16-FCN16: |
|---|---|
| class #0 : 0.74465<br>class #1 : 0.87416<br>class #2 : 0.32958<br>class #3 : 0.78323<br>class #4 : 0.73437<br>class #5 : 0.66033<br>mean_iou: 0.687720 | class #0 : 0.75916<br>class #1 : 0.88346<br>class #2 : 0.35816<br>class #3 : 0.80161<br>class #4 : 0.76096<br>class #5 : 0.66990<br>mean_iou: 0.705542 |

- Reason:

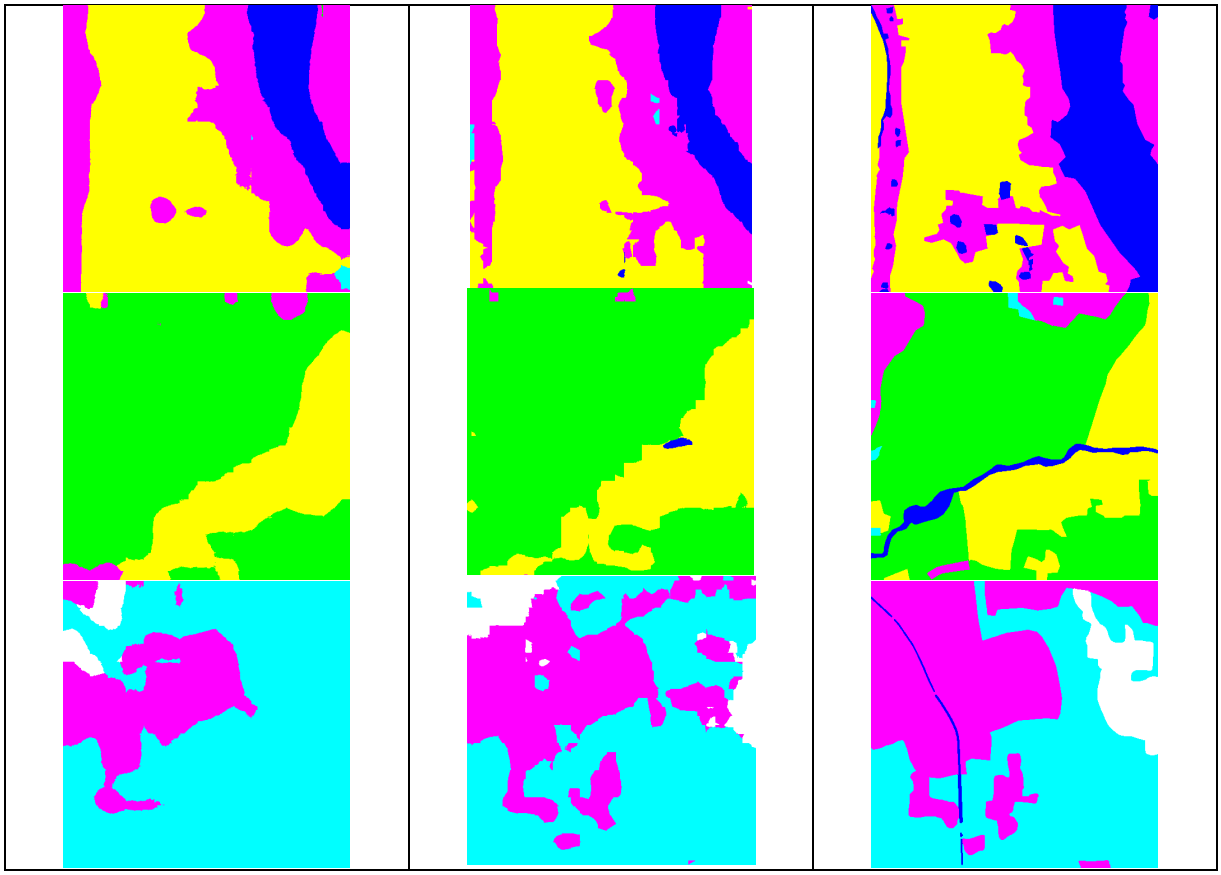
FCN16 相較於 FCN32，多了一層對 pool4（第四個 max pooling 輸出）做分類，再把這個結果和原本從 pool5 後兩層再 de-convolutional 兩倍的結果做 elementwise-add，這會有以下差別：

1. pool4 和 pool5 相比，精細度較高，可以多拿到比較細部的資訊，兩者合併後可同時保留 coarse 和 fine 的資訊；

2. pool5 的圖片在做 de-convolutional 時一次放大 32 倍，突然要多生出很多參數，可能效果較差。

可明顯由這份報告的第二、第四題的圖片看出（複製在下表），中間欄的圖片，有比較多細節（雖然不一定都正確），左欄的切割都比較接近一坨一坨圓圓的。

| FCN32, 40 epochs | FCN16, 40 epochs | Ground Truth |
|---|---|---|



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

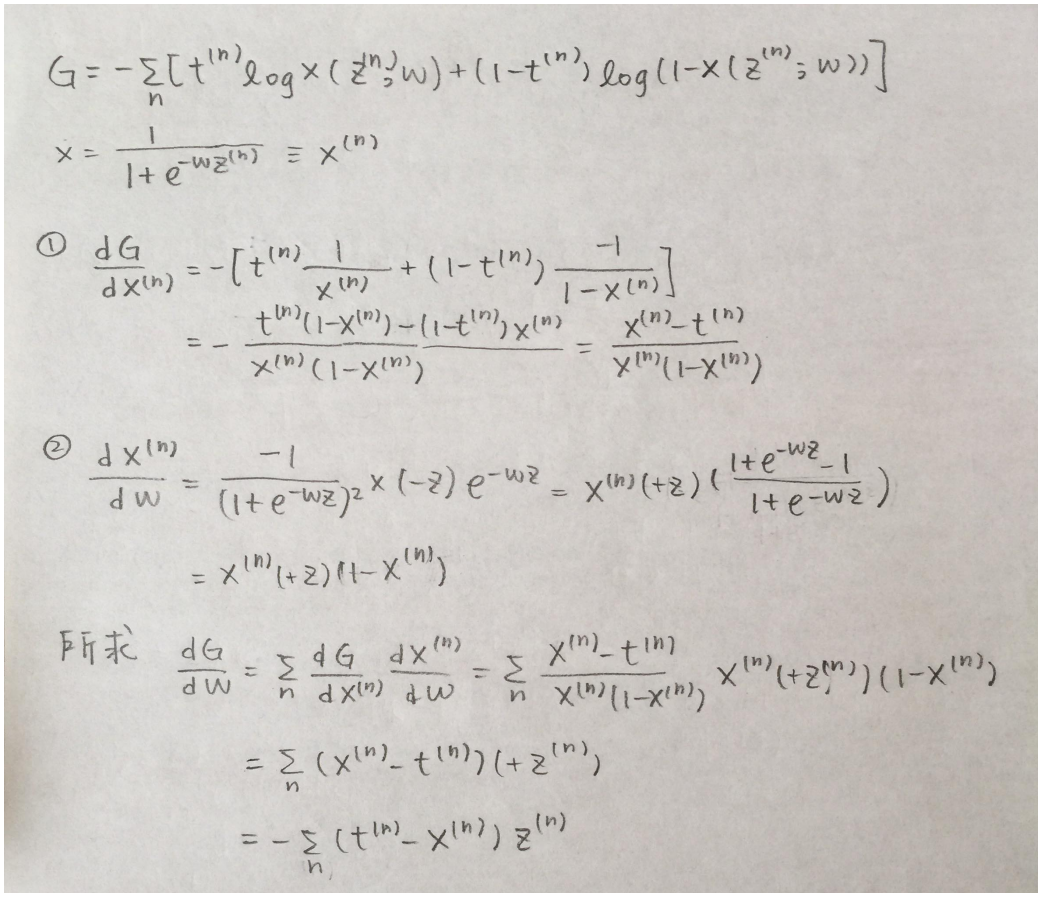
**objective function:**

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

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} G(\boldsymbol{w})$$ 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)



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

$$x = \frac{1}{1+e^{-wz^{(n)}}} \equiv x^{(n)}$$

① $$\frac{dG}{dx^{(n)}} = -\left[ t^{(n)} \frac{1}{x^{(n)}} + (1-t^{(n)}) \frac{-1}{1-x^{(n)}} \right]$$

$$= -\frac{t^{(n)}(1-x^{(n)}) - (1-t^{(n)})x^{(n)}}{x^{(n)}(1-x^{(n)})} = \frac{x^{(n)} - t^{(n)}}{x^{(n)}(1-x^{(n)})}$$

② $$\frac{dx^{(n)}}{dw} = \frac{-1}{(1+e^{-wz})^2} \times (-z) e^{-wz} = x^{(n)}(+z)\left(\frac{1+e^{-wz}-1}{1+e^{-wz}}\right)$$

$$= x^{(n)}(+z)(1-x^{(n)})$$

所求 $$\frac{dG}{dw} = \sum_n \frac{dG}{dx^{(n)}} \frac{dx^{(n)}}{dw} = \sum_n \frac{x^{(n)} - t^{(n)}}{x^{(n)}(1-x^{(n)})} x^{(n)}(+z^{(n)})(1-x^{(n)})$$

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

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

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**).