

Deep Learning for Computer Vision

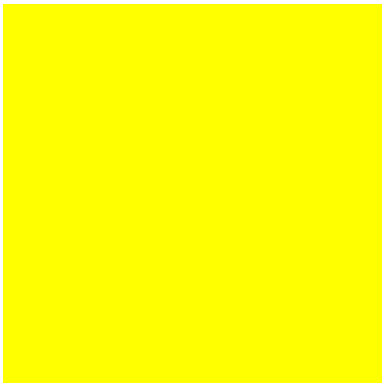
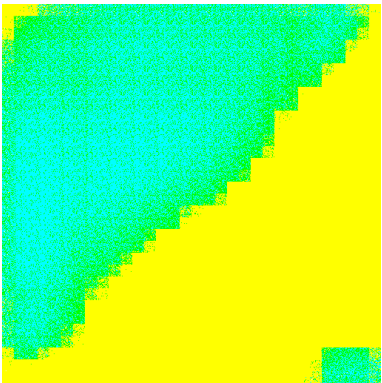
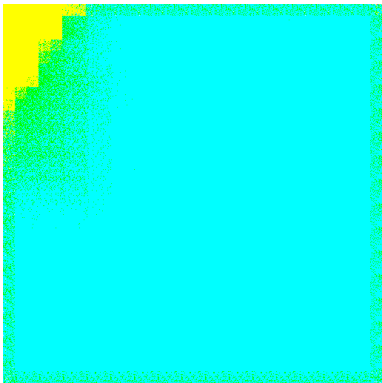

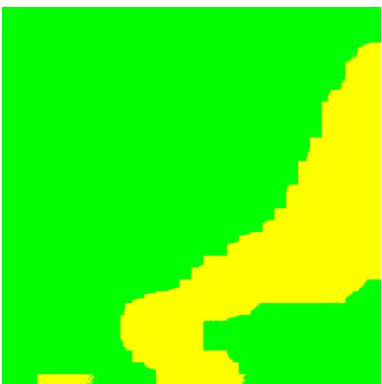




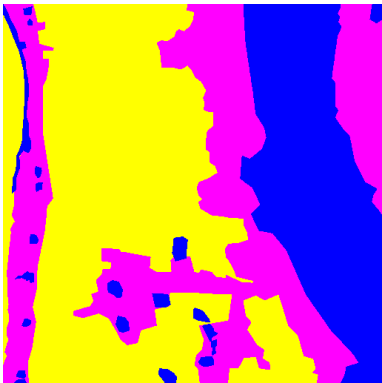
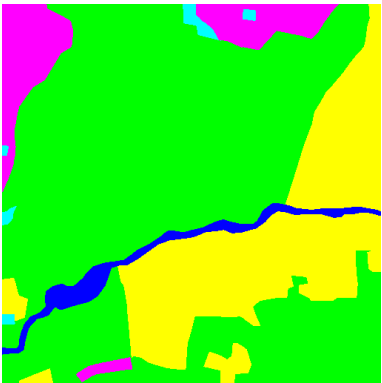
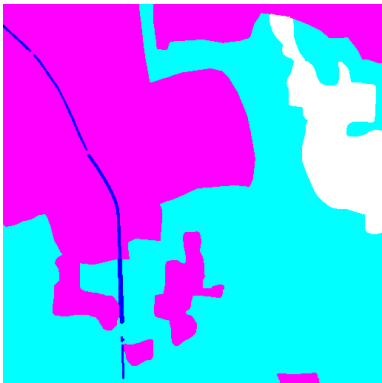
Homework 3

R06922072 黃宇謙

Problem 1: The network architecture of 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, 512)	12845568
dropout_1 (Dropout)	(None, 16, 16, 512)	0
block7_conv1 (Conv2D)	(None, 16, 16, 512)	262656
dropout_2 (Dropout)	(None, 16, 16, 512)	0
block8_conv1 (Conv2D)	(None, 16, 16, 7)	3591
block9_transpose (Conv2DTran	(None, 544, 544, 7)	200704
cropping2d_1 (Cropping2D)	(None, 512, 512, 7)	0
activation_1 (Activation)	(None, 512, 512, 7)	0
Total params: 28,027,207		
Trainable params: 28,027,207		
Non-trainable params: 0		

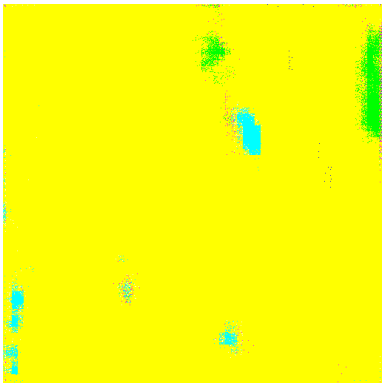
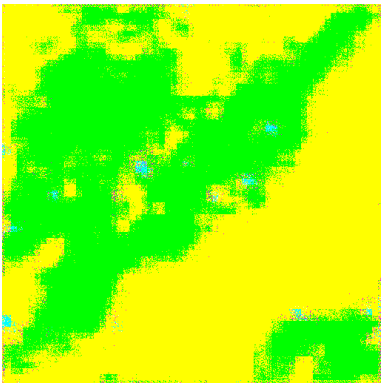
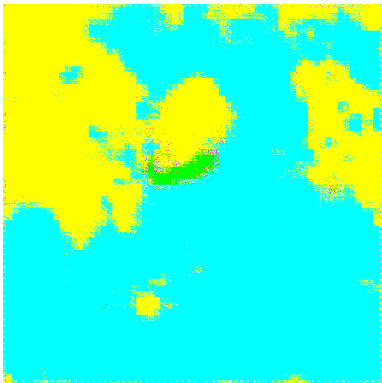
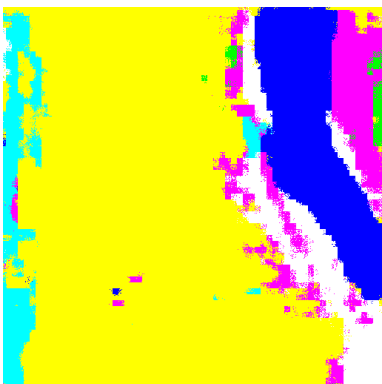
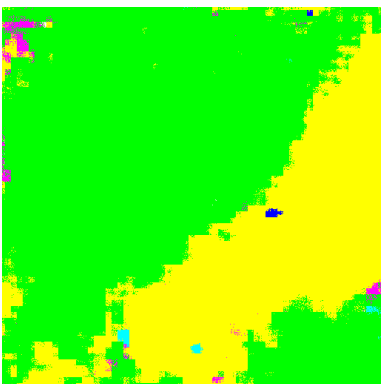
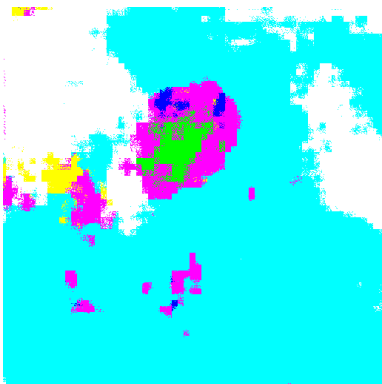
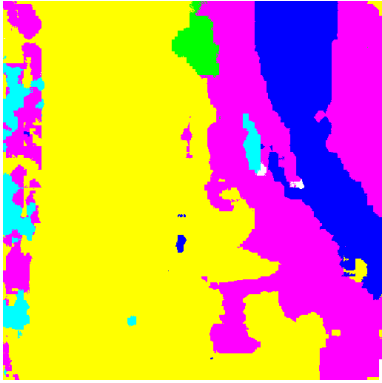
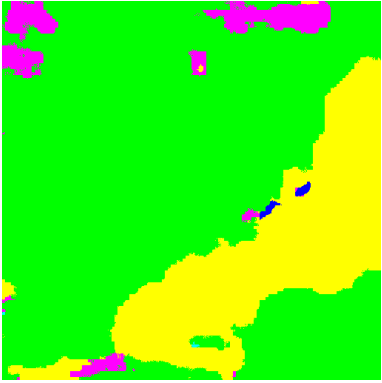
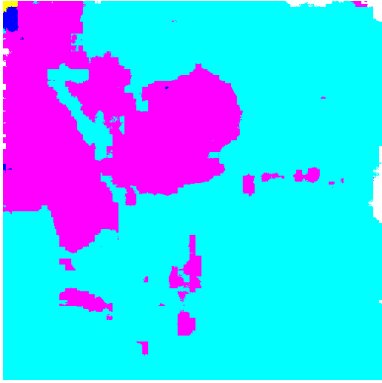
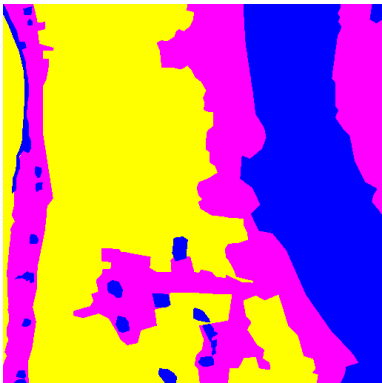
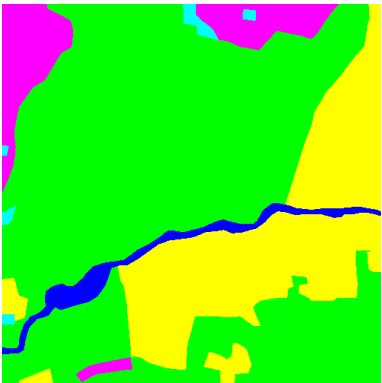
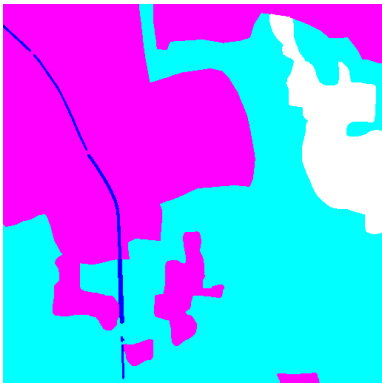
Problem 2: Show the predicted segmentation mask of VGG16-FCN32s model

	0008_mask.png	0097_mask.png	0107_mask.png
Early			
Middle			
Final			
Ground Truth			







Problem 3: The network architecture of improved model (VGG16-FCN8s 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]
block6_conv1 (Conv2D)	(None, 16, 16, 512)	12845568	block5_pool[0][0]
dropout_1 (Dropout)	(None, 16, 16, 512)	0	block6_conv1[0][0]
block7_conv1 (Conv2D)	(None, 16, 16, 512)	262656	dropout_1[0][0]
dropout_2 (Dropout)	(None, 16, 16, 512)	0	block7_conv1[0][0]
score (Conv2D)	(None, 16, 16, 7)	3591	dropout_2[0][0]
upscore_1 (Conv2DTranspose)	(None, 34, 34, 7)	784	score[0][0]
fcn4_score (Conv2D)	(None, 32, 32, 7)	3591	block4_pool[0][0]
cropping2d_1 (Cropping2D)	(None, 32, 32, 7)	0	upscore_1[0][0]
add_1 (Add)	(None, 32, 32, 7)	0	fcn4_score[0][0] cropping2d_1[0][0]
upscore_2 (Conv2DTranspose)	(None, 66, 66, 7)	784	add_1[0][0]
fcn3_score (Conv2D)	(None, 64, 64, 7)	1799	block3_pool[0][0]
cropping2d_2 (Cropping2D)	(None, 64, 64, 7)	0	upscore_2[0][0]
add_2 (Add)	(None, 64, 64, 7)	0	fcn3_score[0][0] cropping2d_2[0][0]
upscore_3 (Conv2DTranspose)	(None, 520, 520, 7)	12544	add_2[0][0]

Problem 4: Show the predicted segmentation mask of improved model



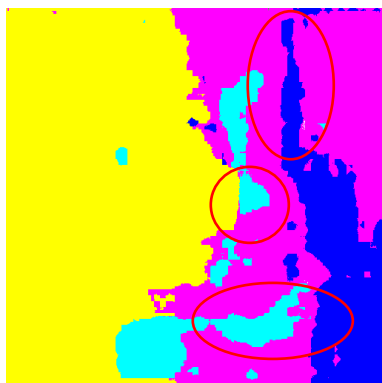
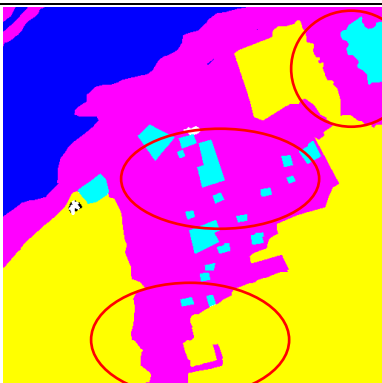
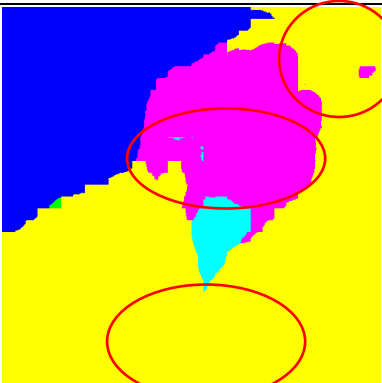
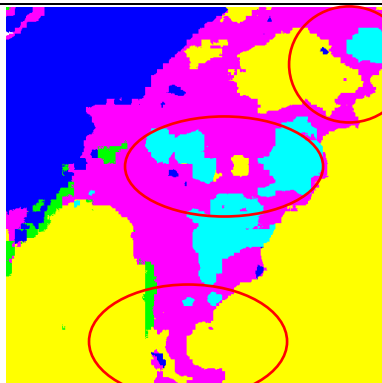
	0008_mask.png	0097_mask.png	0107_mask.png
Early			
Middle			
Final			
Ground Truth			

Problem 5:

	VGG16-FCN32s model	Improved model (VGG16-FCN8s model)
 Class #0 Urban	0.74924	0.75377
 Class #1 Agriculture	0.86632	0.88633
 Class #2 Rangeland	0.26339	0.34878
 Class #3 Forest	0.78939	0.81362
 Class #4 Water	0.66142	0.76085
 Class #5 Barren	0.66780	0.69878
Mean_iou	0.666260	0.710355

Reason:

在 FCN8s model 中，它並不是直接將最後一層的 score 直接放大 32 倍當成輸出，而是先放大兩倍，對第四層 maxpooling 輸出的 score 做 pixel wise 的相加，再放大兩倍，對第三層 maxpooling 輸出的 score 做 pixel wise 的加成，最後才放大回原圖的大小。藉由跟前面幾層的 score 做 pixel wise 加成，它能從中取得一些影像中較為細小的部分(小物件)，使得它的 image segmentation 在較為細小的部分有比較好的結果。(在有運用 Maxpooling 的卷積網路中，越後面的層數，特徵較為不明顯的部分會逐漸消失)

	Ground Truth	FCN32s model	FCN8s model
#1			
#2			

上面以兩個圖當作範例，紅色圓圈為我拿來比較的部分，在該區塊較為小的部分(顏色)，在 FCN32s model 中預測出來的結果裡，基本上這些相關的資訊(顏色)都消失不見，而在 FCN8s model 中還能看到部分的區塊，且有不錯的表現。這些因素，也導致最後 FCN8s 的 mean_iou 表現得比 FCN32s 還要好。