Deep Learning for Computer Vision (2021 Fall) HW1

R10942152 Ken Yu(游家權)

Problem 1 : Image Classification

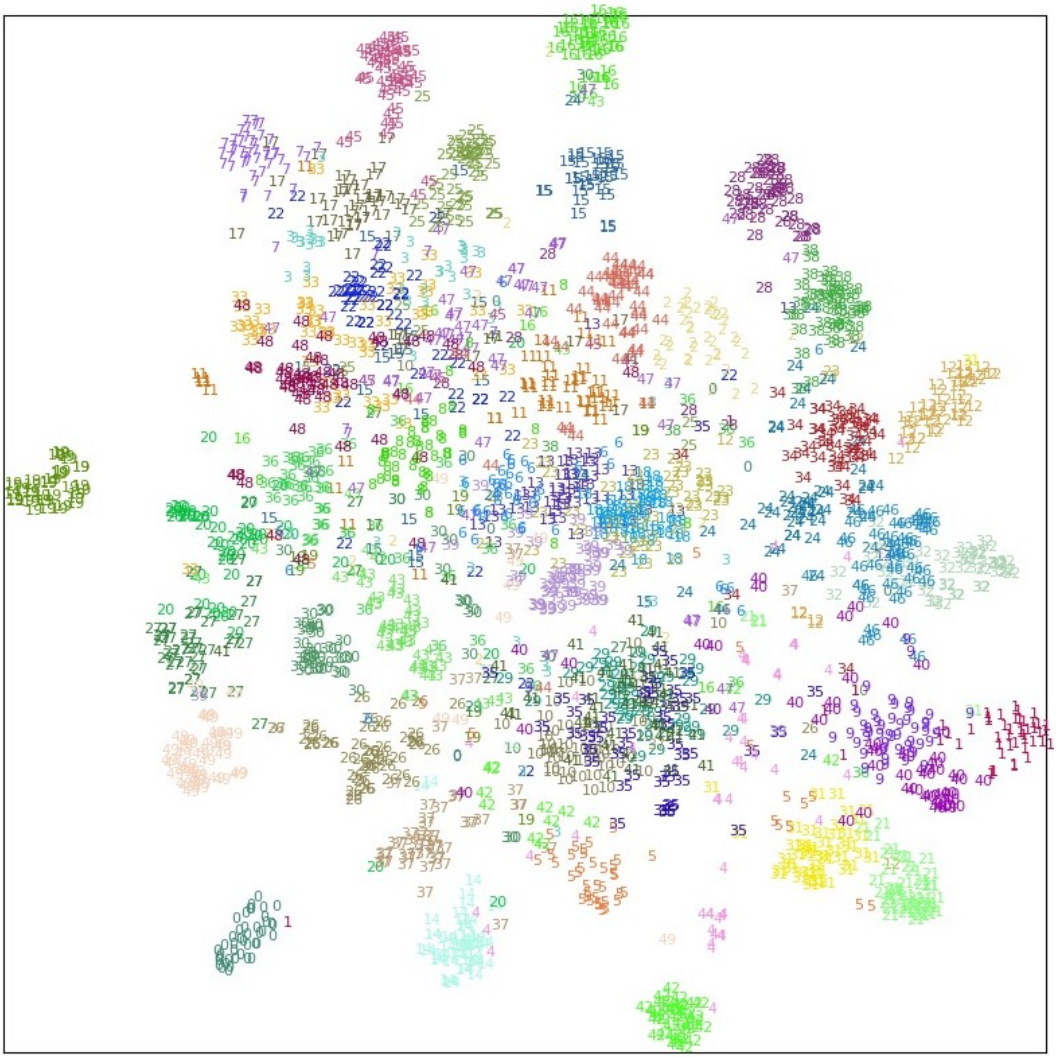
1. (2%) Print the network architecture of your model.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 224, 224]	1,792
BatchNorm2d-2	[-1, 64, 224, 224]	128
ReLU-3	[-1, 64, 224, 224]	0
Conv2d-4	[-1, 64, 224, 224]	36,928
BatchNorm2d-5	[-1, 64, 224, 224]	128
ReLU-6	[-1, 64, 224, 224]	0
MaxPool2d-7	[-1, 64, 112, 112]	0
Conv2d-8	[-1, 128, 112, 112]	73,856
BatchNorm2d-9	[-1, 128, 112, 112]	256
ReLU-10	[-1, 128, 112, 112]	0
Conv2d-11	[-1, 128, 112, 112]	147,584
BatchNorm2d-12	[-1, 128, 112, 112]	256
ReLU-13	[-1, 128, 112, 112]	0
MaxPool2d-14	[-1, 128, 56, 56]	0
Conv2d-15	[-1, 256, 56, 56]	295,168
BatchNorm2d-16	[-1, 256, 56, 56]	512
ReLU-17	[-1, 256, 56, 56]	0
Conv2d-18	[-1, 256, 56, 56]	590,080
BatchNorm2d-19	[-1, 256, 56, 56]	512
ReLU-20	[-1, 256, 56, 56]	0
Conv2d-21	[-1, 256, 56, 56]	590,080
BatchNorm2d-22	[-1, 256, 56, 56]	512
ReLU-23	[-1, 256, 56, 56]	0
MaxPool2d-24	[-1, 256, 28, 28]	0
Conv2d-25	[-1, 512, 28, 28]	1,180,160
BatchNorm2d-26	[-1, 512, 28, 28]	1,024
ReLU-27	[-1, 512, 28, 28]	0
Conv2d-28	[-1, 512, 28, 28]	2,359,808
BatchNorm2d-29	[-1, 512, 28, 28]	1,024
ReLU-30	[-1, 512, 28, 28]	0
Conv2d-31	[-1, 512, 28, 28]	2,359,808
BatchNorm2d-32	[-1, 512, 28, 28]	1,024
ReLU-33	[-1, 512, 28, 28]	0
MaxPool2d-34	[-1, 512, 14, 14]	0
Conv2d-35	[-1, 512, 14, 14]	2,359,808
BatchNorm2d-36	[-1, 512, 14, 14]	1,024
ReLU-37	[-1, 512, 14, 14]	0
Conv2d-38	[-1, 512, 14, 14]	2,359,808
BatchNorm2d-39	[-1, 512, 14, 14]	1,024
ReLU-40	[-1, 512, 14, 14]	0
Conv2d-41	[-1, 512, 14, 14]	2,359,808
BatchNorm2d-42	[-1, 512, 14, 14]	1,024
ReLU-43	[-1, 512, 14, 14]	0
MaxPool2d-44	[-1, 512, 7, 7]	0
AdaptiveAvgPool2d-45	[-1, 512, 7, 7]	0
Linear-46	[-1, 4096]	102,764,544
ReLU-47	[-1, 4096]	0
Dropout-48	[-1, 4096]	0
Linear-49	[-1, 4096]	16,781,312
ReLU-50	[-1, 4096]	0
Dropout-51	[-1, 4096]	0
Linear-52	[-1, 50]	204,850
VGG-53	[-1, 50]	0

2. (2%) Report accuracy of model on the validation set. (TA will reproduce your results, error ±0.5%)

Ans: Accuracy on validation set is **<u>0.8356</u>**

3. (6%) Visualize the classification result on validation set by implementing t-SNE on output features of the second last layer. Briefly explain your result of the tSNE visualization.



Ans: We can tell from tSNE plot that some classes are far from others, while some overlap with other classes. If the class is far from others, it means it has some distinct feature that make it easier to classify. If the class is overlapped

with others, it means their features can easily confuse detector, hence harder to classify.

For instance, class 0 is bicycle which has lots of features to recognize, making it stay far from other classes on tSNE plot. On the other hand, class 29 and class 35 are adult and child; due to the similarity of these features, these two classes are almost inseparable on tSNE plot.

<u>Problem 2: Segmentation</u>

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

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 512, 512]           1,792
              ReLU-2         [-1, 64, 512, 512]               0
            Conv2d-3         [-1, 64, 512, 512]          36,928
              ReLU-4         [-1, 64, 512, 512]               0
         MaxPool2d-5         [-1, 64, 256, 256]               0
            Conv2d-6        [-1, 128, 256, 256]          73,856
              ReLU-7        [-1, 128, 256, 256]               0
            Conv2d-8        [-1, 128, 256, 256]         147,584
              ReLU-9        [-1, 128, 256, 256]               0
        MaxPool2d-10        [-1, 128, 128, 128]               0
           Conv2d-11        [-1, 256, 128, 128]         295,168
             ReLU-12        [-1, 256, 128, 128]               0
           Conv2d-13        [-1, 256, 128, 128]         590,080
             ReLU-14        [-1, 256, 128, 128]               0
           Conv2d-15        [-1, 256, 128, 128]         590,080
             ReLU-16        [-1, 256, 128, 128]               0
        MaxPool2d-17          [-1, 256, 64, 64]               0
           Conv2d-18          [-1, 512, 64, 64]       1,180,160
             ReLU-19          [-1, 512, 64, 64]               0
           Conv2d-20          [-1, 512, 64, 64]       2,359,808
             ReLU-21          [-1, 512, 64, 64]               0
           Conv2d-22          [-1, 512, 64, 64]       2,359,808
             ReLU-23          [-1, 512, 64, 64]               0
        MaxPool2d-24          [-1, 512, 32, 32]               0
           Conv2d-25          [-1, 512, 32, 32]       2,359,808
             ReLU-26          [-1, 512, 32, 32]               0
           Conv2d-27          [-1, 512, 32, 32]       2,359,808
             ReLU-28          [-1, 512, 32, 32]               0
           Conv2d-29          [-1, 512, 32, 32]       2,359,808
             ReLU-30          [-1, 512, 32, 32]               0
        MaxPool2d-31          [-1, 512, 16, 16]               0
           Conv2d-32         [-1, 4096, 16, 16]       2,101,248
             ReLU-33         [-1, 4096, 16, 16]               0
        Dropout2d-34         [-1, 4096, 16, 16]               0
           Conv2d-35         [-1, 4096, 16, 16]      16,781,312
             ReLU-36         [-1, 4096, 16, 16]               0
        Dropout2d-37         [-1, 4096, 16, 16]               0
           Conv2d-38            [-1, 7, 16, 16]          28,679
  ConvTranspose2d-39          [-1, 7, 512, 512]         200,704
================================================================
Total params: 33,826,631
Trainable params: 33,826,631
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 3.00
Forward/backward pass size (MB): 1203.01
Params size (MB): 129.04
Estimated Total Size (MB): 1335.05
----------------------------------------------------------------
```

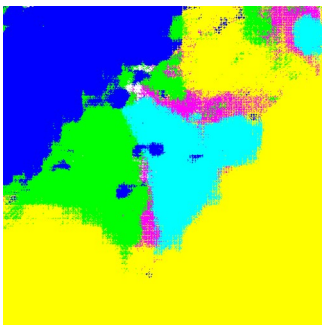
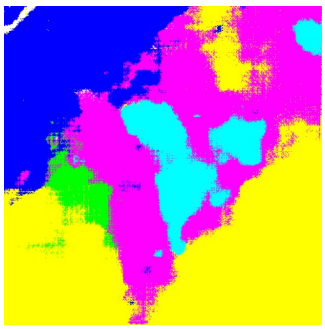
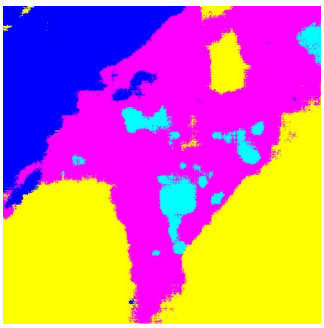
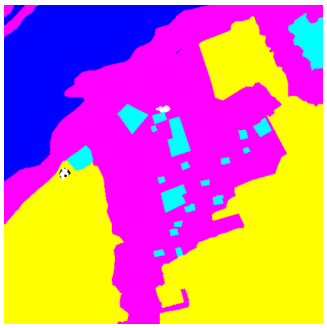
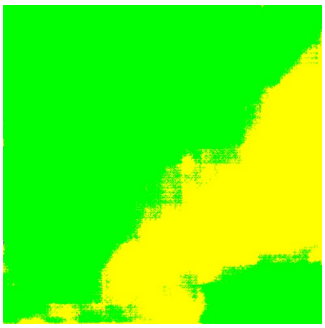
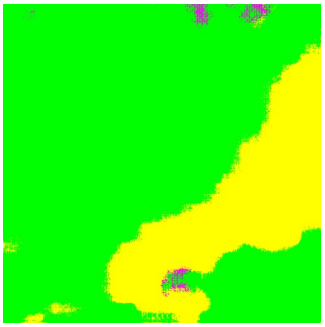
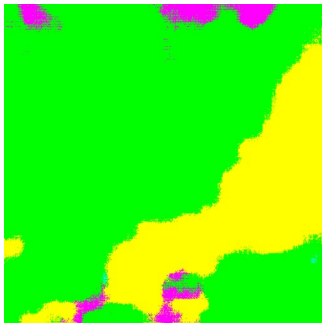
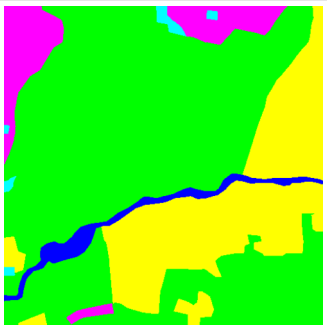
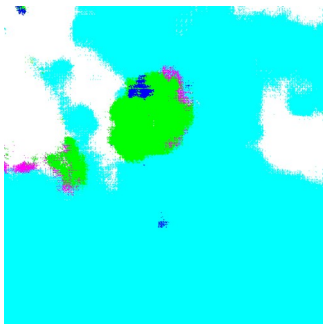
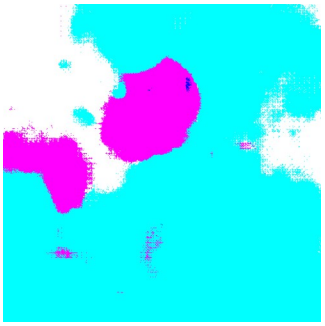
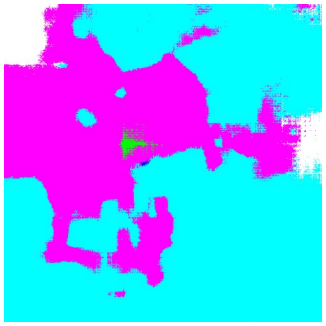
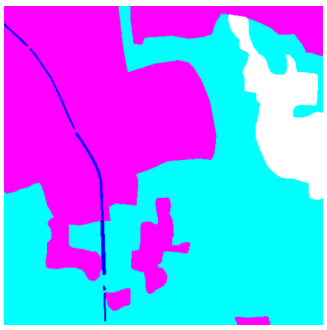
2. (5%) Implement an improved model which performs better than your baseline model. Print the network architecture of this model.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 512, 512]           1,792
              ReLU-2         [-1, 64, 512, 512]               0
            Conv2d-3         [-1, 64, 512, 512]          36,928
              ReLU-4         [-1, 64, 512, 512]               0
         MaxPool2d-5         [-1, 64, 256, 256]               0
            Conv2d-6        [-1, 128, 256, 256]          73,856
              ReLU-7        [-1, 128, 256, 256]               0
            Conv2d-8        [-1, 128, 256, 256]         147,584
              ReLU-9        [-1, 128, 256, 256]               0
        MaxPool2d-10        [-1, 128, 128, 128]               0
           Conv2d-11        [-1, 256, 128, 128]         295,168
             ReLU-12        [-1, 256, 128, 128]               0
           Conv2d-13        [-1, 256, 128, 128]         590,080
             ReLU-14        [-1, 256, 128, 128]               0
           Conv2d-15        [-1, 256, 128, 128]         590,080
             ReLU-16        [-1, 256, 128, 128]               0
        MaxPool2d-17          [-1, 256, 64, 64]               0
           Conv2d-18          [-1, 512, 64, 64]       1,180,160
             ReLU-19          [-1, 512, 64, 64]               0
           Conv2d-20          [-1, 512, 64, 64]       2,359,808
             ReLU-21          [-1, 512, 64, 64]               0
           Conv2d-22          [-1, 512, 64, 64]       2,359,808
             ReLU-23          [-1, 512, 64, 64]               0
        MaxPool2d-24          [-1, 512, 32, 32]               0
           Conv2d-25          [-1, 512, 32, 32]       2,359,808
             ReLU-26          [-1, 512, 32, 32]               0
           Conv2d-27          [-1, 512, 32, 32]       2,359,808
             ReLU-28          [-1, 512, 32, 32]               0
           Conv2d-29          [-1, 512, 32, 32]       2,359,808
             ReLU-30          [-1, 512, 32, 32]               0
        MaxPool2d-31          [-1, 512, 16, 16]               0
  ConvTranspose2d-32          [-1, 512, 32, 32]       2,359,808
             ReLU-33          [-1, 512, 32, 32]               0
      BatchNorm2d-34          [-1, 512, 32, 32]           1,024
  ConvTranspose2d-35          [-1, 256, 64, 64]       1,179,904
             ReLU-36          [-1, 256, 64, 64]               0
      BatchNorm2d-37          [-1, 256, 64, 64]             512
  ConvTranspose2d-38        [-1, 128, 128, 128]         295,040
             ReLU-39        [-1, 128, 128, 128]               0
      BatchNorm2d-40        [-1, 128, 128, 128]             256
  ConvTranspose2d-41         [-1, 64, 256, 256]          73,792
             ReLU-42         [-1, 64, 256, 256]               0
      BatchNorm2d-43         [-1, 64, 256, 256]             128
  ConvTranspose2d-44         [-1, 32, 512, 512]          18,464
             ReLU-45         [-1, 32, 512, 512]               0
      BatchNorm2d-46         [-1, 32, 512, 512]              64
           Conv2d-47          [-1, 7, 512, 512]             231
================================================================
Total params: 18,643,911
Trainable params: 18,643,911
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 3.00
Forward/backward pass size (MB): 1527.00
Params size (MB): 71.12
Estimated Total Size (MB): 1601.12
----------------------------------------------------------------
```

3. (5%) Report mIoU of the improved model on the validation set. (TA will reproduce your results, error ±0.5%)

Ans: My improved model is **FCN8s**, which produce **0.71** mean_iou on validation dataset. By the way, my baseline model only produce 0.68 mean_iou in the same experiment.

4. (5%) Show the predicted segmentation mask of “validation/0010_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 (5 epoch)	Middle Stage (20 epoch)	Final Stage (40 epoch)	Label
			
			
			

Reference

[1] How to get intermediate layer's output.

https://discuss.pytorch.org/t/how-can-l-load-my-best-model-as-a-feature-extractor-evaluator/17254/6

[2] FNC8s

https://zhuanlan.zhihu.com/p/73965733

[3] Adam Optimizer

https://zhuanlan.zhihu.com/p/32626442

[4] UNet

https://github.com/milesial/Pytorch-UNet