Homework #1

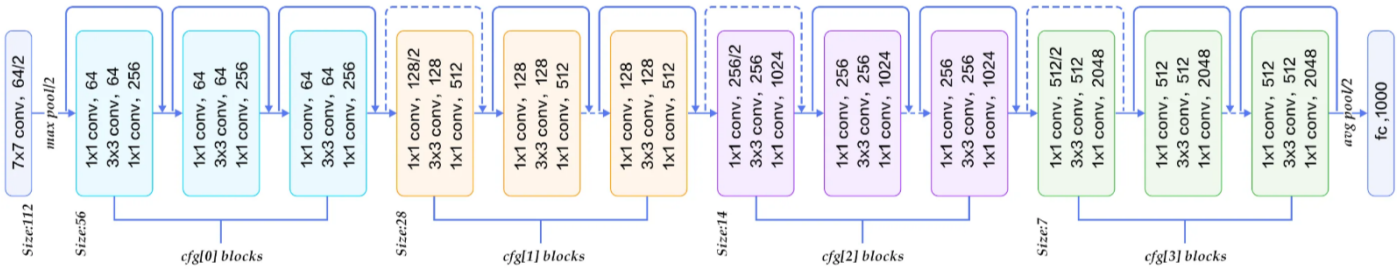
Deep Learning for Computer Vision

NTU, Fall 2023

周奕節
r10943131

Problem 1: Image Classification

1. **(2%)** Draw the network architecture of method A or B.
    a. Model A backbone: Resnet50, cfg=[3, 4, 6, 3].
    b. Modify the FC layer's output to 50.



ref: https://blog.devgenius.io/resnet50-6b42934db431

2. **(1%)** Report accuracy of your models (both A, B) on the validation set.
    a. model A: validation accuracy = 65.36%

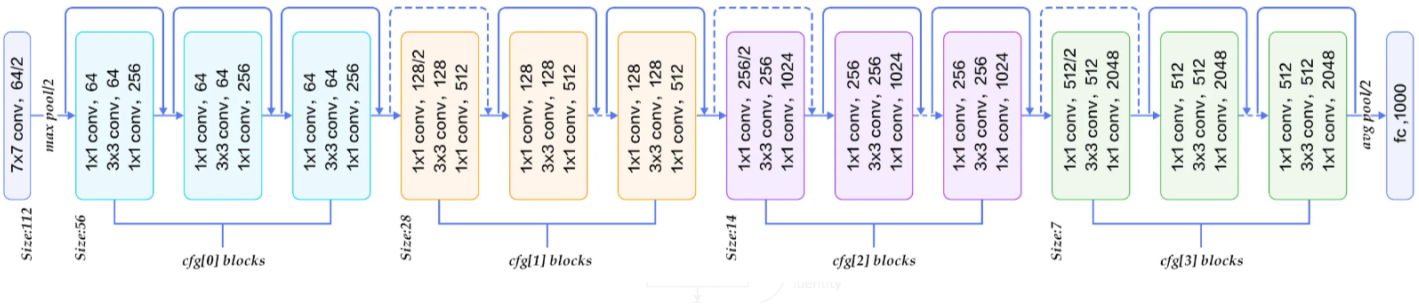
```
============================
epoch = 25
training loss : 0.011541891297698021  train acc = 0.8820888888888889
val loss : 0.043918472719192504  val acc = 0.6536
============================
```

    b. model B: validation accuracy = 87.44%

```
============================
epoch = 99
training loss : 0.001226466033121364  train acc = 0.9903555555555555
val loss : 0.01485545292943716  val acc = 0.8744
============================
```

3. **(2%)** Report your implementation details of model A.
    a. Loss function: Cross entropy loss, often used when dealing with classification tasks
    b. Optimizer: SGD, with initial learning rate = 0.01, momentum = 0.9, weight decay = 1e-6, and Nesterov momentum, which adjusts the update direction slightly ahead of the current parameter value to improve convergence.
    c. Scheduler: 'MultiStepLR' could dynamically adjust the learning rate during training. With milestone=[12, 24, 32] and gamma=0.1, which means the LR will reduced by 90% in 12, 24, and 32 epochs.

4. **(3%)** Report your alternative model or method in B, and describe its difference from model A.
    a. Model B backbone: ResNet152, cfg=[3, 8, 36, 3].
    b. Modify the FC layer's output to 50.
    c. Difference: ResNet152 is much deeper than ResNet50, there are 36 cfg[2] blocks. Thus, it has a larger capacity to capture more complex patterns and less prone to overfitting.

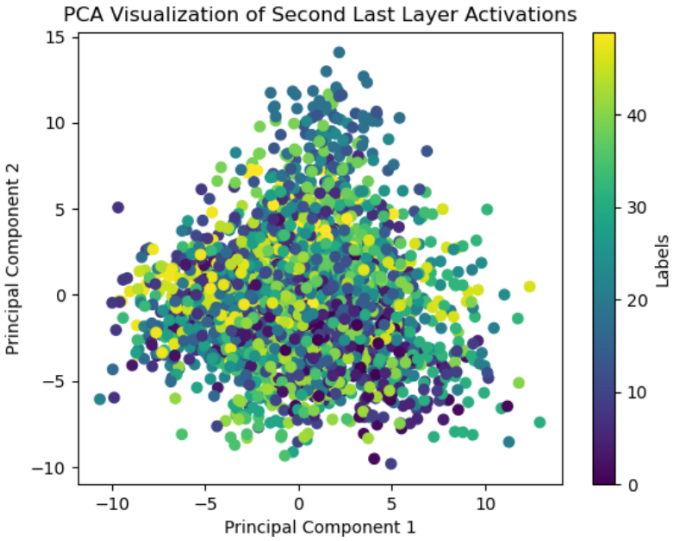


ref: https://blog.devgenius.io/resnet50-6b42934db431

5. **(3%)** Visualize the learned visual representations of **model A** on the **validation set** by implementing **PCA** on the output of **the second last layer**. Briefly explain your result of the PCA visualization.

    Since the validation accuracy of model A is 65.36%, the visualization of PCA isn't that well.

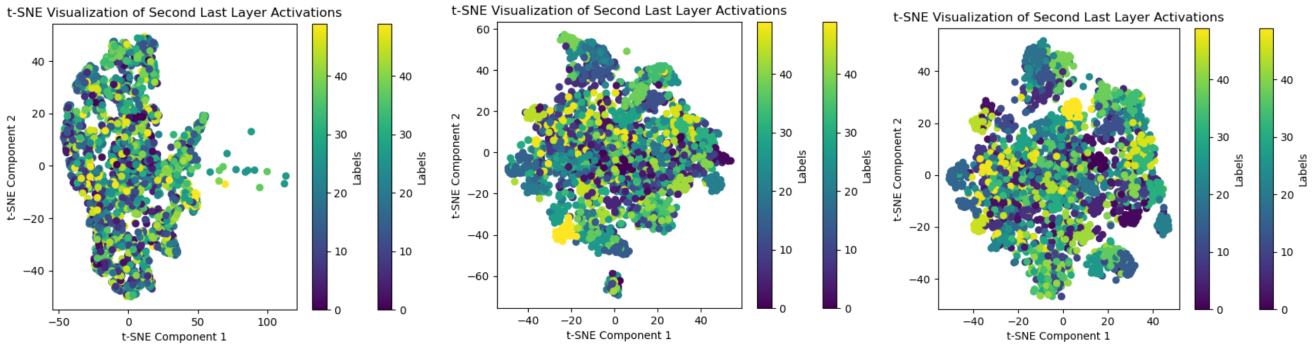
    There is no obvious color of cluster and different classes overlap each other. We can tell that the model didn't capture different features effectively.



PCA on the output of the second last layer

6. **(4%)** Visualize the learned visual representation of **model A**, again on the output of the second last layer, but using **t-SNE** instead. Depict your visualization from **three different epochs** including the first one and the last one. Briefly explain the above results.

The figure below from left to right is the visualization from the first to the last epoch. At the first epoch, the points are in high density, meaning that the model had not yet classify features and different classes aren't separated. Then we can find the the points starts to spread out in the middle progress, indicating that some features were learned. In the last epoch, the points separated the widest and is relatively stable. However, different color clusters aren't separated so well since validation accuracy of model A is 65.36%.



t-SNE of the first epoch

t-SNE of the middle epoch

t-SNE of the last epoch

Problem 2: Self-Supervised Pre-training for Image Classification

1. (5%) Describe the implementation details of your SSL method for pre-training the ResNet50 backbone.

   a. SSL method: Bootstrap Your Own Latent (BYOL)
   b. Data augmentation: resize and center to 128x128, however, we don't need centercrop if we had resized it to 128x128
   c. Loss function: Cross entropy loss.
   d. Optimizer: 'AdamW', an improved version of Adam that includes weight decay. Learning rate = 1e-3, weight decay = 1e-5.
   e. Scheduler: 'CosineAnnealingWarmRestarts', adjusts the LR according to a cosine annealing schedule. 'T_0=2', means the LR is reset to its maximum value after 2 epochs. 'T_mult=2', means T_0 will multiply by 2 after each restart.
   f. Batch size = 64 for the limited memory size of the GPU.

2. (20%) Please conduct the Image classification on **Office-Home** dataset as the downstream task. Also, please complete the following Table, which contains different image classification setting, and **discuss/analyze** the results.

Discuss/Analyze:

1. Discuss Setting A v.s. B and C:

   From the results we can know that pre-training(B and C) performs better than training from scratch(A). It seems Setting A wasn't trained effectively, it may need more epochs, datas or fine-tuning of hyperparameters.

2. Discuss Setting B v.s. C:

   In this case SSL pre-trained backbone(C) performs slightly better than SL pre-trained backbone(B). We can say that using SSL backbone on Mini-ImageNet can help improve performance on Office-Home dataset. As for the SL backbone, there may be some room for improvement.

3. Discuss Setting B v.s. D and C vs. E:

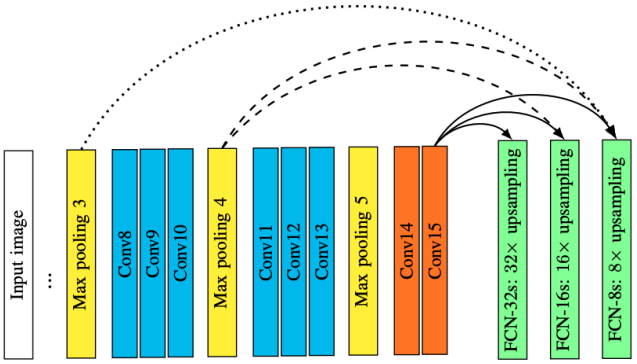
   It is obvious that fine-tuning both the backbone and classifier(B and C) will have better results than fixing the backbone and training only the classifier (D and E). Consequently, Setting B has higher validation accuracy than setting D, similar results for Setting C and E.

Setting	Pre-training (Mini-ImageNet)	Fine-tuning (Office-Home dataset)	Validation accuracy (Office-Home dataset)
A	-	Train full model (backbone + classifier)	*0.281*
B	w/ label (TAs have provided this backbone)	Train full model (backbone + classifier)	*0.544*
C	w/o label (Your SSL pre-trained backbone)	Train full model (backbone + classifier)	*0.608*
D	w/ label (TAs have provided this backbone)	Fix the backbone. Train classifier only	*0.249*
E	w/o label (Your SSL pre-trained backbone)	Fix the backbone. Train classifier only	*0.318*

Problem 3: Semantic Segmentation

1. **(3%)** Draw the network architecture of your VGG16-FCN32s model (model A).

VGG-16 architecture

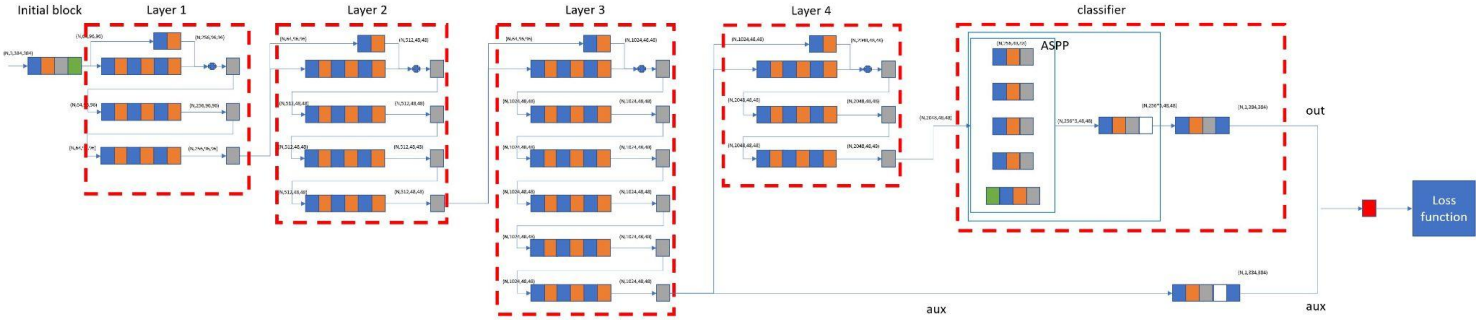


VGG16-FCN32s (ignore FCN-16s/8s on the right)

ref:https://www.researchgate.net/publication/322413149_Everything_You_Wanted_to_Know_about_Deep_Learning_for_Computer_Vision_but_Were_Afraid_to_Ask

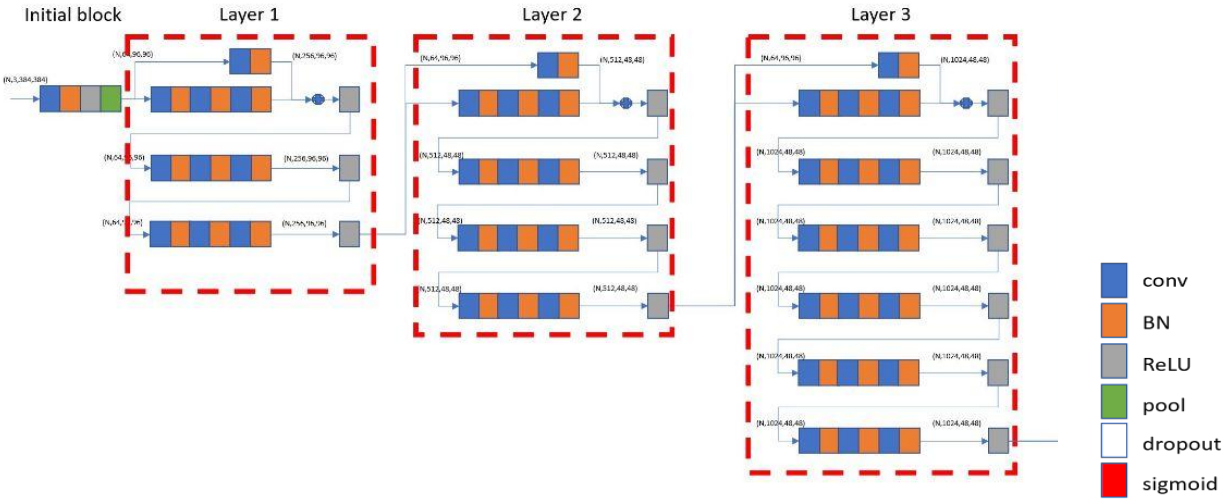
2. **(3%)** Draw the network architecture of the improved model (model B)
and explain it differs from your VGG16-FCN32s model.
    a. Model B: DeepLabv3_resnet50
    b. Backbone: Resnet is a newer backbone that can train deeper network, while VGG is a simpler one.
    c. Segmentation Architecture: Different from fully convolutional network(FCN), DeepLabv3 uses atrous convolutions to capture multi-scale context. Atrous convolutions and ASPP module make DeepLabv3 outperforms FCN32.

Full Scale

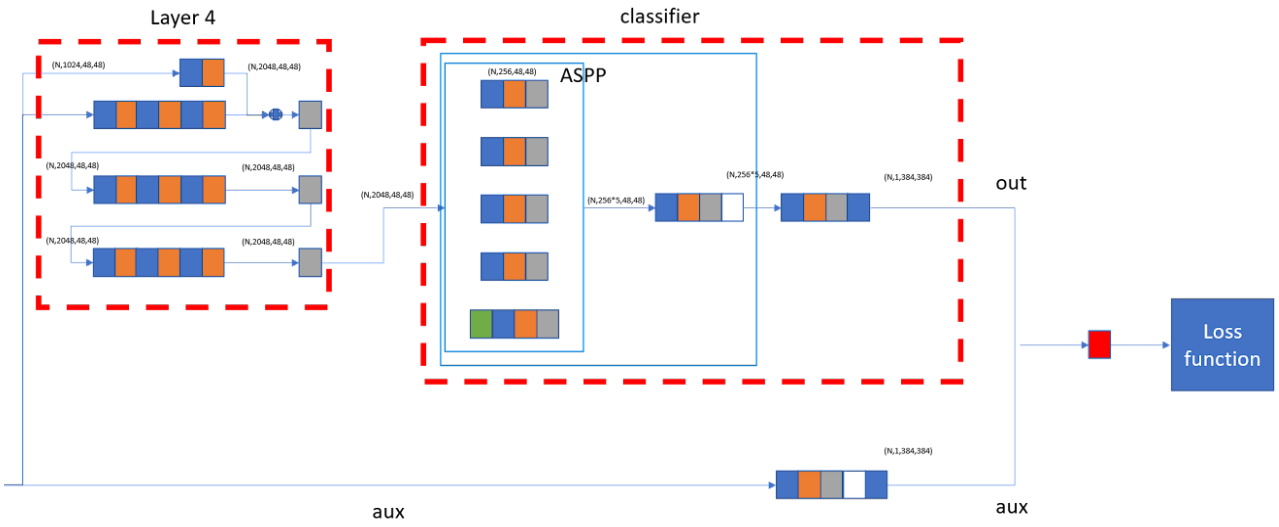


DeepLabv3_resnet50

Left-half part



Right-half part



ref: https://blog.csdn.net/weixin_44816589/article/details/115266935

3. **(1%)** Report mIoUs of two models on the validation set.

   a. model A: validation mIoU = 48.97%

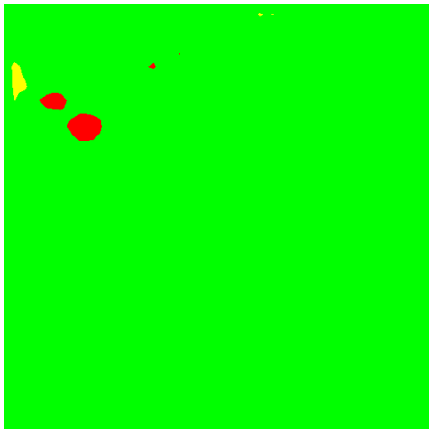
```
============================
epoch = 11
training loss : 0.0765215776860714  train iou = 0.44322762612355054
val loss : 0.07061218797928628  val iou = 0.48971773479750624
============================
```

   b. model B: validation mIoU = 73.98%

```
============================
epoch = 11
training loss : 0.017414316449314356  train iou = 0.8607634907804809
val loss : 0.04077544999725624  val iou = 0.7398336101613023
============================
```

4. **(3%)** Show the predicted segmentation mask of “validation/0013_sat.jpg”, “validation/0062_sat.jpg”, “validation/0104_sat.jpg” during the early, middle, and the final stage during the training process of the improved model.

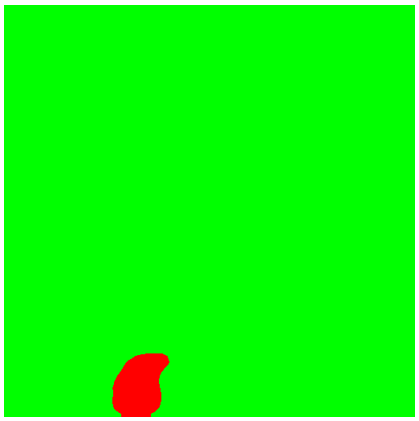
We can find that in the early stages of segmentation, The images contain mostly one color and not much places being segmented. But when in the middle of segmentation, the progress is almost stable and could crop most of the objects. In the final of the segmentation, it only fine-tunes some details from the middle stages.



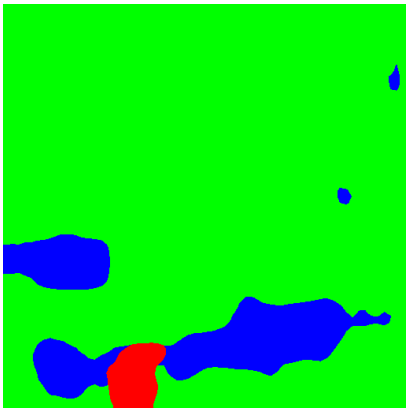
0013_early_mask

0013_middle_mask

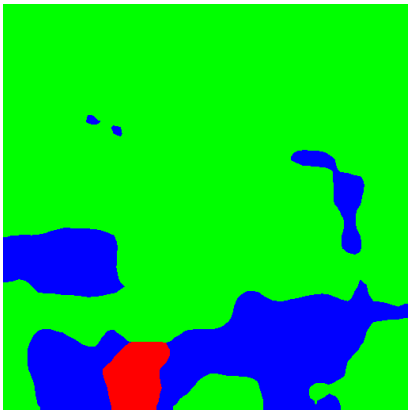
0013_final_mask



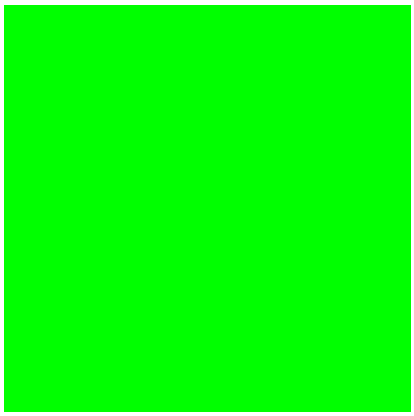
0062_early_mask



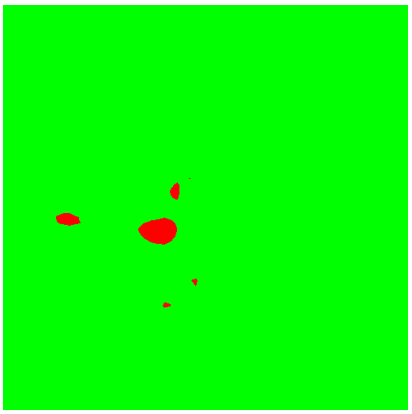
0062_middle_mask



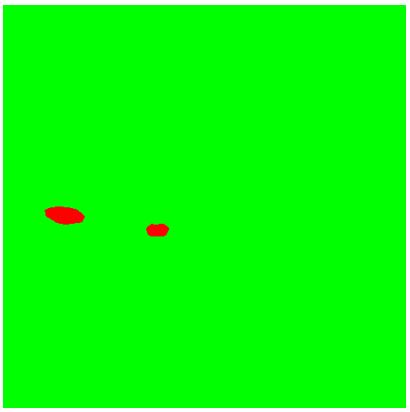
0062_final_mask



0104_early_mask



0104_middle_mask



0104_final_mask