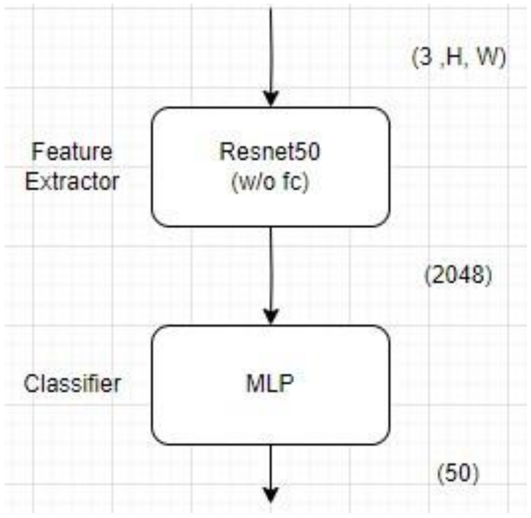
DLCV HW1

R12943010 林孟平

Problem 1: Image Classification

(1) Draw the network architecture of method A or B.

Model A



(2) Report accuracy of your models (both A, B) on the validation set.

Accuracy of model A: 0.5552

Accuracy of model B: 0.8788

(3) Report your implementation details of model A.

In the Training phase of model A, I select the best model with the highest validation accuracy in the training progress.

Hyperparameters:

Loss Function: CrossEntropyLoss

Total Training Epoch: 50

Batch Size: 16

Optimizer: Adam with learning rate = 1e-3

Scheduler: CosineAnnealingLR with T_max = 50, last_epoch = -1 Data Augmentation:

Resize image to (224, 224),

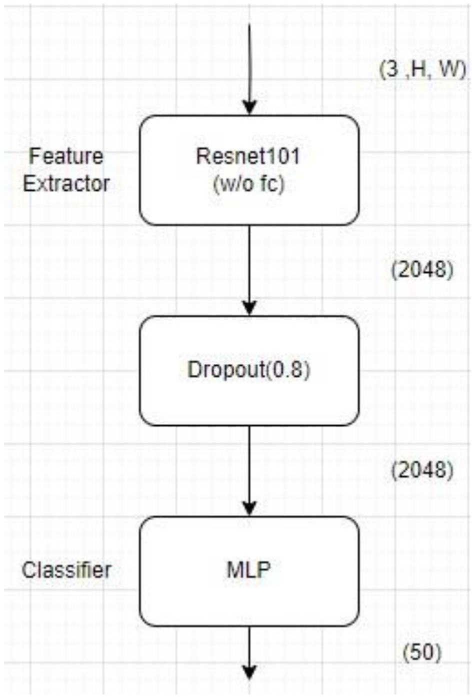
Normalizaiton with mean = [0.508, 0.481, 0.431], std = [0.200, 0.199, 0.203],

RandomCrop(224, 224, padding = 4)

HorizontalFlip with p = 0.5
RandomRotation(-30, 30)

(4) Report your alternative model or method in B, and describe its difference from model A

Model B



We use pre-trained weights in Resnet101 (Model B)
In the Training phase of model B, I also select the best model with the highest validation accuracy in the training progress.

Hyperparameters:
Loss Function: CrossEntropyLoss
Total Training Epoch: 15
Batch Size: 32
Optimizer: SGD with learning rate = 7e-4, weight decay = 3e-4, momentum = 0.9
Scheduler: CosineAnnealingLR with T_max = 15, last_epoch = -1 Data Augmentation:
Resize image to (224, 224),
Normalizaiton with mean = [0.508, 0.481, 0.431], std = [0.200, 0.199, 0.203],
RandomCrop(224, 224, padding = 4)
HorizontalFlip with p = 0.5
RandomRotation((-30, 30))

Difference between A and B:

(a) Batch size and Total epochs

I guess that it takes less time for model B to converge in training phase, so I select smaller batch size and training epoch compared with model A.

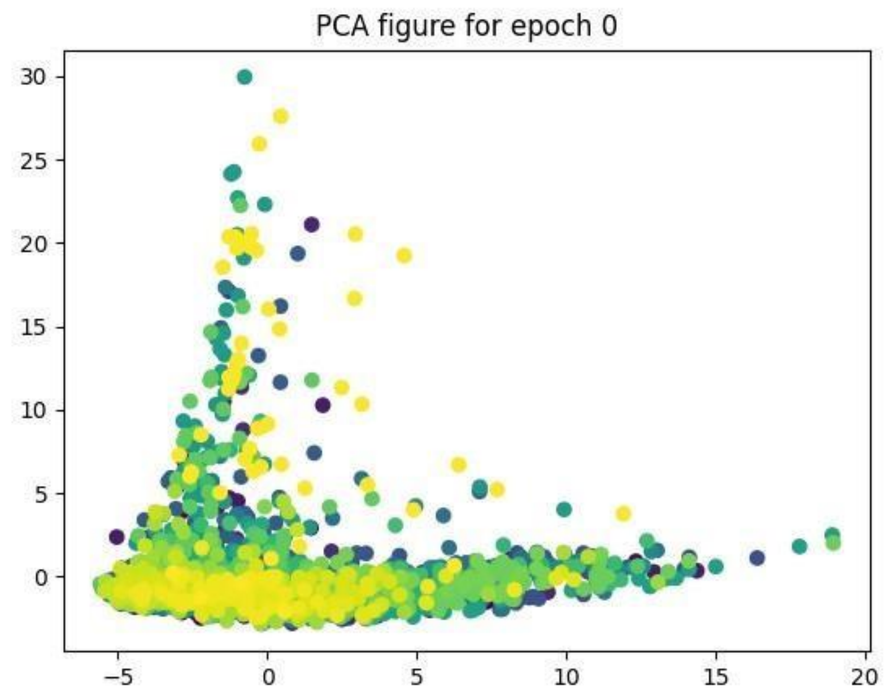
(b) Optimizer

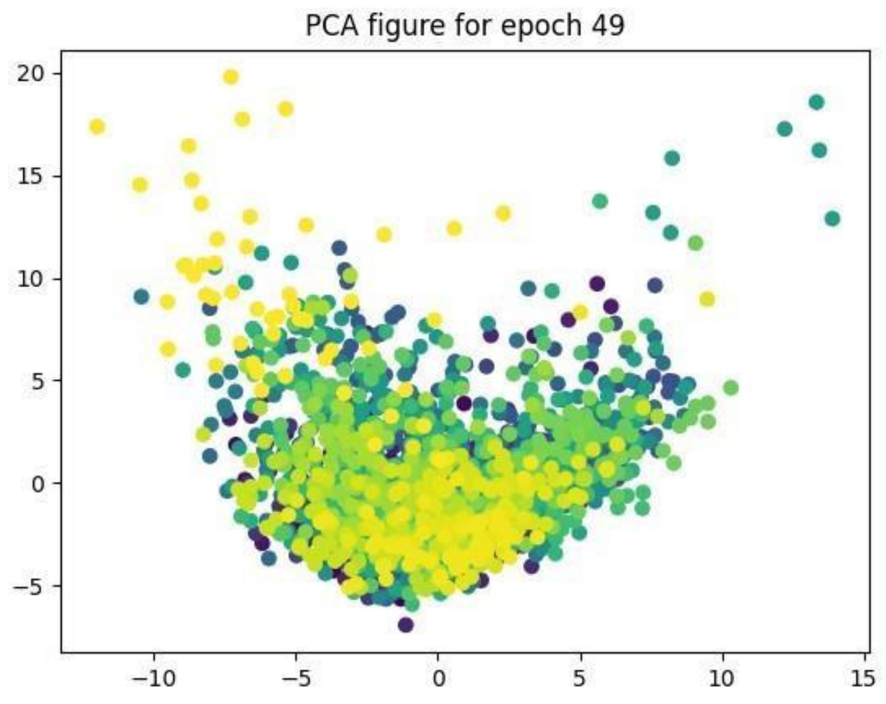
The optimizer of model A is Adam with default learning rate, while the optimizer of model B is SGD with parameters given in the paper of Resnet.

(c) Dropout Layer

It seems that model A encounter overfitting issue. To avoid this, we add a dropout layer at the end of the feature extractor in model B.

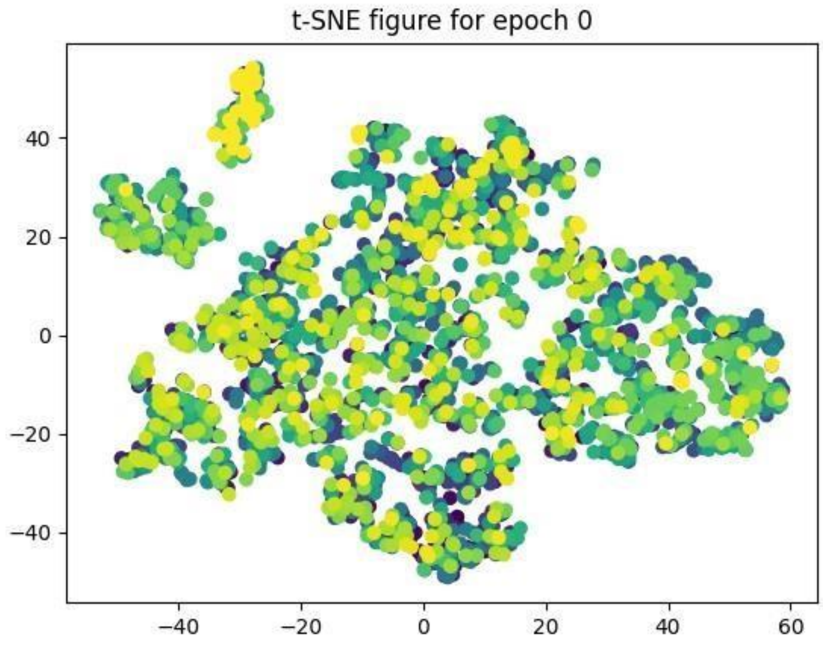
(5) Visualize the learned visual representations of model A on the validation set by implementing PCA (Principal Component Analysis) on the output of the second last layer. Briefly explain your result of the PCA visualization.

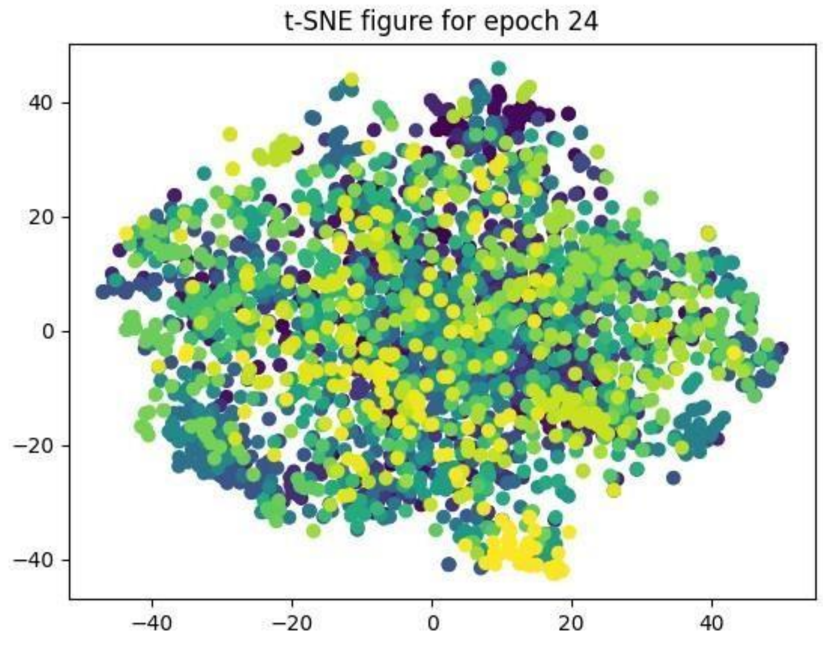


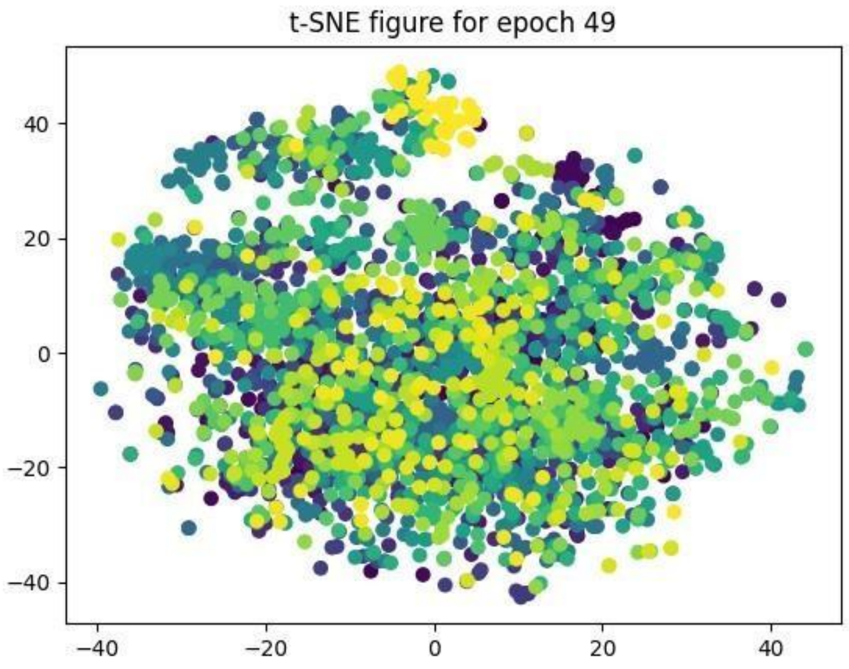


From PCA visualization, we can see that the model gain ability to separate different classes after training. For example, lots of yellow points are on the left side of the graph, while green points are on the right side of the graph. However, there is a clustering with different color of points at the center. It may indicate that some images are not classified correctly.

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







From t-SNE visualization, we can see that the model gain ability to separate different classes after training. Although due to the low accuracy, the model still can’t separate different classes very well, we can see that the problem of clustering is not as serious as PCA.

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

(1) Describe the implementation details of your SSL method for pretraining the ResNet50 backbone.

In the Pre-training phase, I use BYOL (https://github.com/lucidrains/byol-pytorch) as pre-training method for this problem, with use_momentum = False. Furthermore, I split the MiniDataset into training set and validation set (0.9:0.1) and select the best model with the lowest validation loss in the training progress.

Model: Resnet50

Hyperparameters:
Total Training Epoch: 500
Batch Size: 128
Optimizer: Adam with learning rate = 1e-4

Scheduler: CosineAnnealingLR with T_max = 500, last_epoch = -1 Data
Augmentation:

Resize image to (128, 128),

Normalizaiton with mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]

(2) 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.

In the Finetuning phase, I select the best model with the highest validation accuracy in the training progress.

Model:

Backbone: Resnet50

Classifier: Linear (1000, 512) + Batch Norm + ReLU + Linear (512, 65)

Hyperparameters:

Total Training Epoch: 200

Batch Size: 128

Optimizer: Adam with learning rate = 3e-4 Scheduler:
StepLR with step size = 20, gamma = 0.75 Data
Augmentation:

Resize image to (128, 128),

Normalizaiton with mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225],

RandomCrop(128, 128)

GaussianBlur((3, 3), (1.0, 2.0)) with p = 0.2

ColorJitter((0.7, 0.7, 0.7, 0.2)) with p = 0.1

HorizontalFlip with p = 0.5

RandomRotation((-30, 30))

Setting	Pre-training	Finetuning	Accuracy
A	x	Train full model	0.3424
B	w/ label (SL)	Train full model	0.4409
C	w/o label (SSL)	Train full model	0.4803
D	w/ label (SL)	Train classifier	0.2291

E	w/o label (SSL)	Train classifier	0.3768

⇨ C is the best among all the setting.

(a)

Under the same setting for finetuning the models, different pre-training methods give us different accuracy:

SSL Pre-Training> SL Pre-Training > No- Pre-Training (C>B>A, E>D)

Pre-Training methods help the Resnet backbone learn how to extract good features in the image, and I guess that compared with SL Pre-Training, SSL Pre-Training did a better job. The result also justifies the statement that professor gave in the lecture: "The result of SSL may be superior to SL in lots of cases."

(b)

Under the same setting for Pre-Training, different settings for finetuning give us different accuracy:

Train full model > Train classifier (B>D, C>E)

I guess that the reason is "the way to extract good features in MiniDataset" is different from "the way to extract good features in OfficeDataset". If we fix the weights of the backbone, we also limit the possibility that the backbone gain ability to extract good features in the classification task. Therefore, we need to finetune the backbone to get better performance.

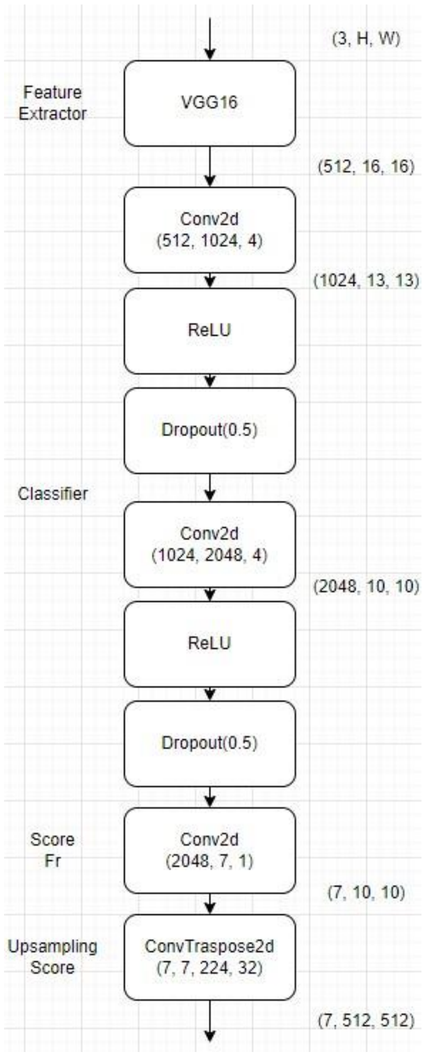
(c)

The time required for convergence is also worth mentioning

No- Pre-Training (~150 ep) > SSL Pre-Training (~70 ep) > SL Pre-Training (~40ep) It is intuitive that No-Pre-Trained model needs more time to learn how to extract good features. For different Pre-Training methods, I guess that it is because SSL PreTrained model have more flexibility than SL- Pre-Trained model to learn how to extract feature in classification task. Therefore, SSL requires more time to converge and thus has better accuracy.

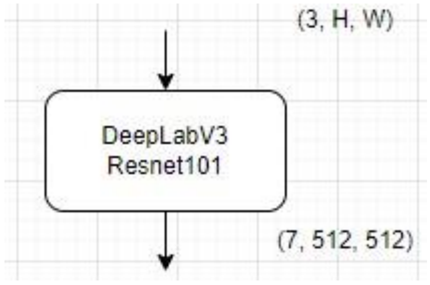
Problem 3: Semantic Segmentation

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



We use pre-trained weights in VGG16

(2) Draw the network architecture of the improved model (model B) and explain it differs from your VGG16-FCN32s model.



We use pre-trained weights (on MSCOCO dataset) in DeepLabV3-Resnet101
In the Training phase of model A, I select the best model with the highest validation mIOU in the training progress.

Hyperparameters (Model A):
Loss Function: CrossEntropyLoss
Total Training Epoch: 100
Batch Size: 8
Optimizer: Adam with learning rate = 1e-4
Scheduler: CosineAnnealingLR with T_max = 100, last_epoch = -1 Data Augmentation:
  Normalizaiton with mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]
 RandomHorizontalFlip(0.5)
 RandomVerticalFlip(0.5)

In the Training phase of model B, I select the best model with the highest validation mIOU in the training progress.

Hyperparameters (Model B):
Loss Function: CrossEntropyLoss
Total Training Epoch: 30
Batch Size: 4
Optimizer: SGD with learning rate = 0.002, weight decay = 1e-5, momentum = 0.9
Scheduler: CosineAnnealingLR with T_max = 30, last_epoch = -1 Data Augmentation:
  Normalizaiton with mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]
 RandomHorizontalFlip(0.5)
 RandomVerticalFlip(0.5)

Difference between A and B:
In VGG16-FCN32s (model A), we only up-sample one feature map (1/32) to get segmentation result. To achieve better mIOU score, we use relative larger model "DeepLabV3-Resnet101" (model B). In this model, besides the low resolution feature maps, we also use dilated convolutions to obtain multi-scale pyramid. We merge these feature maps together to get final segmentation result, and the accuracy reported by two models also shows that model B surpass model A.

Furthermore, the settings of training also differ in two models:

(a) Batch size and Total epochs

I guess that it takes less time for model B to converge in training phase, so I select smaller batch size and training epoch compared with model A.

(b) Optimizer

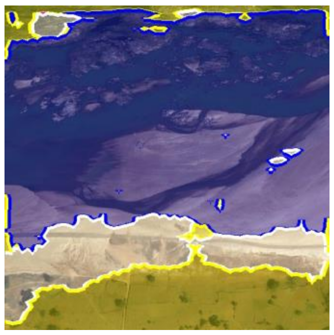
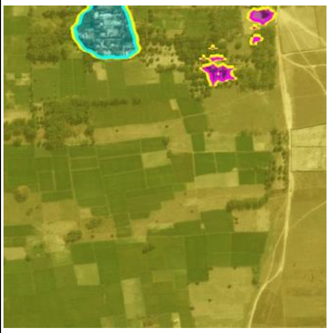
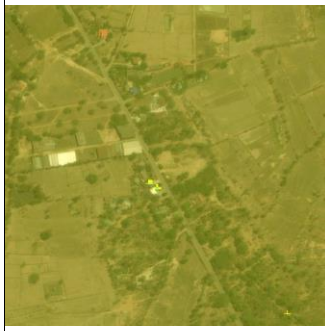
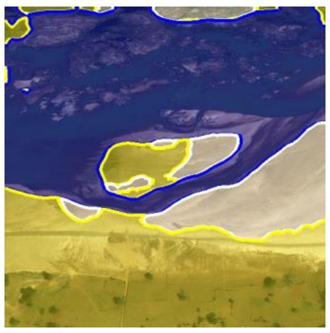
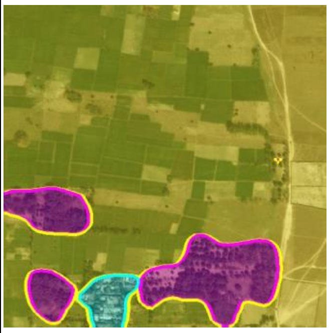
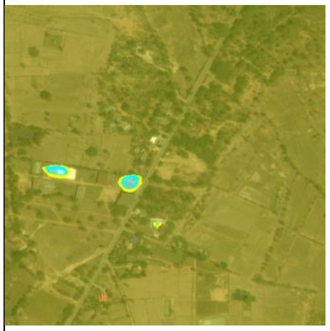
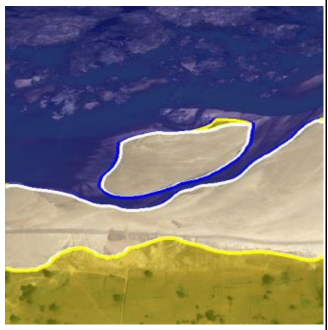
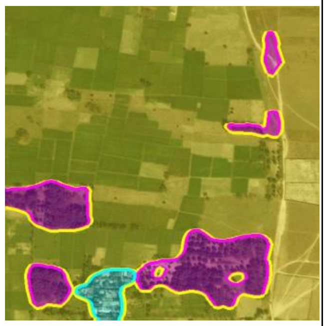
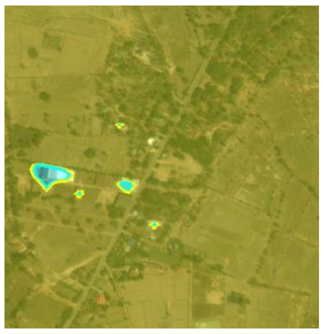
The optimizer of model A is Adam, while the optimizer of model B is SGD.

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

mIOU of model A: 0.6259

mIOU of model B: 0.7501

(4) **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.**

	0013_sat.jpg	**0062_sat.jpg**	**0104_sat.jpg**
Early			
Middle			
Last			
Ground Truth