DLCV HW1 Report

B07502071 陳志臻

Image classification

1. model architecture - wide\_resnet\_50

Net(

(load\_model): ResNet(

(conv1): Conv2d(3, 64, kernel\_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)

(bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(maxpool): MaxPool2d(kernel\_size=3, stride=2, padding=1, dilation=1, ceil\_mode=False)

(layer1): Sequential(

(0): Bottleneck(

(conv1): Conv2d(64, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(64, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(256, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(256, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(layer2): Sequential(

(0): Bottleneck(

(conv1): Conv2d(256, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(256, 512, kernel\_size=(1, 1), stride=(2, 2), bias=False)

(1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(512, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(512, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(3): Bottleneck(

(conv1): Conv2d(512, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(layer3): Sequential(

(0): Bottleneck(

(conv1): Conv2d(512, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(2, 2), bias=False)

(1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(1024, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(1024, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(3): Bottleneck(

(conv1): Conv2d(1024, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(4): Bottleneck(

(conv1): Conv2d(1024, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(5): Bottleneck(

(conv1): Conv2d(1024, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(layer4): Sequential(

(0): Bottleneck(

(conv1): Conv2d(1024, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(1024, 1024, kernel\_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(1024, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(1024, 2048, kernel\_size=(1, 1), stride=(2, 2), bias=False)

(1): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(2048, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(1024, 1024, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(1024, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(2048, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(1024, 1024, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(1024, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(avgpool): AdaptiveAvgPool2d(output\_size=(1, 1))

(fc): Linear(in\_features=2048, out\_features=1000, bias=True)

)

(Layer): Linear(in\_features=1000, out\_features=50, bias=True)

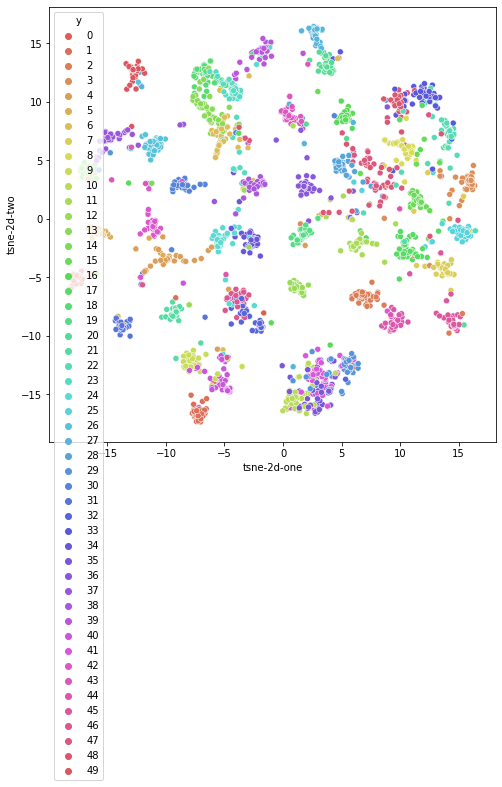
)

1. Accuracy of validation set

Test set: Average loss: 0.0036, Accuracy: 2170/2500 (87%)

1. Result of t-sne

This is the second last layer of my output. In the second last layer, there are 1000 dimensions, so we should find 1000 points in this picture. Points with same color mean that they have same label output. We can find that points in the same color are divided into groups.



Semantic Segmentation

1. model architecture of vgg16-fcn32

FCN32s(

(vgg): VGG(

(features): Sequential(

(0): Conv2d(3, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(1): ReLU(inplace=True)

(2): Conv2d(64, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(3): ReLU(inplace=True)

(4): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(5): Conv2d(64, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(6): ReLU(inplace=True)

(7): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(8): ReLU(inplace=True)

(9): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(10): Conv2d(128, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(11): ReLU(inplace=True)

(12): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(13): ReLU(inplace=True)

(14): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(15): ReLU(inplace=True)

(16): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(17): Conv2d(256, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(18): ReLU(inplace=True)

(19): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(20): ReLU(inplace=True)

(21): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(22): ReLU(inplace=True)

(23): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

(24): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(25): ReLU(inplace=True)

(26): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(27): ReLU(inplace=True)

(28): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1))

(29): ReLU(inplace=True)

(30): MaxPool2d(kernel\_size=2, stride=2, padding=0, dilation=1, ceil\_mode=False)

)

(avgpool): AdaptiveAvgPool2d(output\_size=(7, 7))

(classifier): Sequential(

(0): Linear(in\_features=25088, out\_features=4096, bias=True)

(1): ReLU(inplace=True)

(2): Dropout(p=0.5, inplace=False)

(3): Linear(in\_features=4096, out\_features=4096, bias=True)

(4): ReLU(inplace=True)

(5): Dropout(p=0.5, inplace=False)

(6): Linear(in\_features=4096, out\_features=1000, bias=True)

)

)

(conv6): Conv2d(512, 4096, kernel\_size=(1, 1), stride=(1, 1))

(relu6): ReLU(inplace=True)

(drop6): Dropout2d(p=0.5, inplace=False)

(conv7): Conv2d(4096, 4096, kernel\_size=(1, 1), stride=(1, 1))

(relu7): ReLU(inplace=True)

(drop7): Dropout2d(p=0.5, inplace=False)

(score): Conv2d(4096, 7, kernel\_size=(1, 1), stride=(1, 1))

(upsample32): ConvTranspose2d(7, 7, kernel\_size=(64, 64), stride=(32, 32), bias=False)

)

1. model architecture of improvement model – Deeplabv3-resnet50

DeepLabv3\_ResNet50(

(model): DeepLabV3(

(backbone): IntermediateLayerGetter(

(conv1): Conv2d(3, 64, kernel\_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)

(bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(maxpool): MaxPool2d(kernel\_size=3, stride=2, padding=1, dilation=1, ceil\_mode=False)

(layer1): Sequential(

(0): Bottleneck(

(conv1): Conv2d(64, 64, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(64, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(64, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(64, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(256, 64, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(64, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(64, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(256, 64, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(64, 64, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(64, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(layer2): Sequential(

(0): Bottleneck(

(conv1): Conv2d(256, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(256, 512, kernel\_size=(1, 1), stride=(2, 2), bias=False)

(1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(512, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(512, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(3): Bottleneck(

(conv1): Conv2d(512, 128, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(128, 128, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(128, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(layer3): Sequential(

(0): Bottleneck(

(conv1): Conv2d(512, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(512, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(1024, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(1024, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(3): Bottleneck(

(conv1): Conv2d(1024, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(4): Bottleneck(

(conv1): Conv2d(1024, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(5): Bottleneck(

(conv1): Conv2d(1024, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2), bias=False)

(bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(256, 1024, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

(layer4): Sequential(

(0): Bottleneck(

(conv1): Conv2d(1024, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

(downsample): Sequential(

(0): Conv2d(1024, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(1): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

)

)

(1): Bottleneck(

(conv1): Conv2d(2048, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(4, 4), dilation=(4, 4), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

(2): Bottleneck(

(conv1): Conv2d(2048, 512, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv2): Conv2d(512, 512, kernel\_size=(3, 3), stride=(1, 1), padding=(4, 4), dilation=(4, 4), bias=False)

(bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(conv3): Conv2d(512, 2048, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(bn3): BatchNorm2d(2048, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(relu): ReLU(inplace=True)

)

)

)

(classifier): DeepLabHead(

(0): ASPP(

(convs): ModuleList(

(0): Sequential(

(0): Conv2d(2048, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(2): ReLU()

)

(1): ASPPConv(

(0): Conv2d(2048, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(12, 12), dilation=(12, 12), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(2): ReLU()

)

(2): ASPPConv(

(0): Conv2d(2048, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(24, 24), dilation=(24, 24), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(2): ReLU()

)

(3): ASPPConv(

(0): Conv2d(2048, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(36, 36), dilation=(36, 36), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(2): ReLU()

)

(4): ASPPPooling(

(0): AdaptiveAvgPool2d(output\_size=1)

(1): Conv2d(2048, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(3): ReLU()

)

)

(project): Sequential(

(0): Conv2d(1280, 256, kernel\_size=(1, 1), stride=(1, 1), bias=False)

(1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(2): ReLU()

(3): Dropout(p=0.5, inplace=False)

)

)

(1): Conv2d(256, 256, kernel\_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)

(2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track\_running\_stats=True)

(3): ReLU()

(4): Conv2d(256, 7, kernel\_size=(1, 1), stride=(1, 1))

)

)

)

1. mIoU of the improved model on the validation set

class #0 : 0.77608

class #1 : 0.90939

class #2 : 0.44211

class #3 : 0.84126

class #4 : 0.78139

class #5 : 0.72226

mean\_iou: 0.745417

1. 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.

|  |  |  |  |
| --- | --- | --- | --- |
| Stage\.jpg | 0010\_sat.jpg | 0097\_sat.jpg | 0107\_sat.jpg |
| Early |  |  |  |
| Middle |  |  |  |
| Final |  |  |  |

References:

<https://github.com/kai860115/DLCV2020-FALL>

<https://towardsdatascience.com/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b>

<https://debuggercafe.com/semantic-segmentation-using-pytorch-deeplabv3-resnet50/>

Collaborators:

B08502141 石旻翰

B08902134 曾揚哲