Computer Vision HW2 Report

Student ID: R10522811

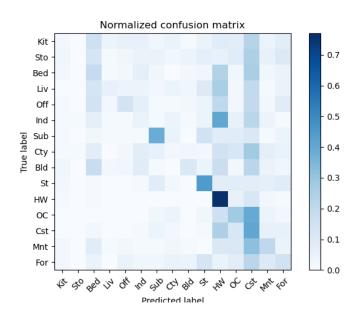
Name: 簡永展

Part 1. (10%)

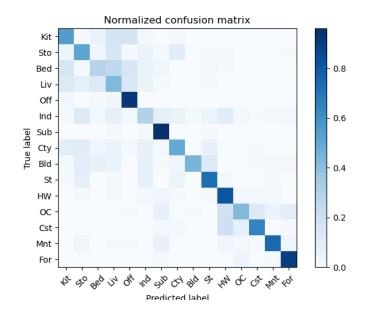
• Plot confusion matrix of two settings. (i.e. Bag of sift and tiny image representation) (5%)

Ans:

Tiny image:



Bag of sift:



• Compare the results/accuracy of both settings and explain the result. (5%) Ans:

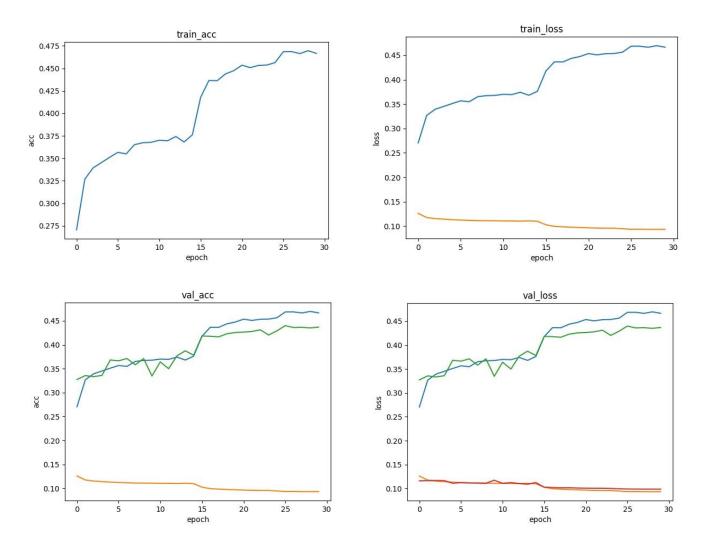
The difference between tiny images and bag of sift is the feature extraction. Tiny images use normal data augmentation (transform) as feature; however, bag of sift do as the convolution as the feature. Therefore, the result of bag of sift is better.

Part 2. (35%)

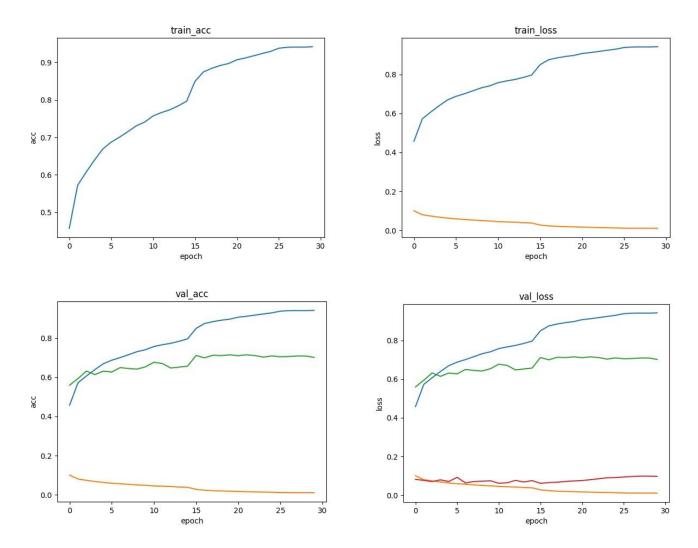
• Compare the performance on residual networks and LeNet. Plot the learning curve (loss and accuracy) on both training and validation sets for both 2 schemes. 8 plots in total. (20%)

Ans:

LeNet:



Resnet:



• Attach basic information of the model you use including model architecture and the number of the parameters. Besides, report the accuracy you performed on the public test set. (5%)

Ans:

```
(resnet): 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, 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=(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, 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, 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)
(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=(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)
(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=(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)
```

```
(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=(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)
 (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=(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)
     (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=(1, 1), 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)
   (avgpool): AdaptiveAvgPool2d(output size=(1, 1))
   (fc): Linear(in features=2048, out features=1000, bias=True)
 (fc1): Linear(in features=1000, out features=512, bias=True)
 (fc2): Linear(in_features=512, out features=10, bias=True)
Loading model parameters from ./save_dir/pretrain/best_model.pt...
End of loading !!!
Number of images is 500
```

• Briefly describe what method do you apply? (e.g. data augmentation, model architecture, loss function, semi-supervised etc.) (10%)

| 1250/1250 [01:15<00:00, 16.66it/s]

Ans:

100%

accuracy: 0.8182

Data augmentation: random horizontal flip

Model architecture: pretrained resnet

Loss function: cross-entropy loss