Computer Vision HW2 Report

Student ID: B11901164
Name: 陳秉緯

Part 1. (10%)

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

Ans:

- bag of sift

- tiny image

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

Ans:

- bag_of_sift: Accuracy = 0.6126666666666667
- tiny_image: Accuracy = 0.22466666666666665

The bag of SIFT method works a lot better than the tiny image approach for scene recognition. It got an accuracy of 0.61, while the tiny image method only reached 0.22. This big difference is mainly because SIFT can catch important local features in an image, even when there are changes like rotation or scale. Then, using the bag-of-words model helps organize and represent these features effectively. Overall, this shows that using local features gives much better results than just relying on basic global image representations, especially when scenes look similar in layout but have different objects and textures.

Part 2. (25%)

• Report accuracy of both models on the validation set. (2%)

Ans:

	A	B

accuracy	0.85120	0.90000

• Print the network architecture & number of parameters of both models. What is the main difference between ResNet and other CNN architectures? (5%)

Ans:

I use torchsummary to get the following result.

Architecture of model A (MyNet):

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1            [-1, 64, 32, 32]           1,792
       BatchNorm2d-2            [-1, 64, 32, 32]             128
              ReLU-3            [-1, 64, 32, 32]               0
            Conv2d-4            [-1, 64, 32, 32]          36,928
       BatchNorm2d-5            [-1, 64, 32, 32]             128
              ReLU-6            [-1, 64, 32, 32]               0
         MaxPool2d-7            [-1, 64, 16, 16]               0
            Conv2d-8           [-1, 128, 16, 16]          73,856
       BatchNorm2d-9           [-1, 128, 16, 16]             256
             ReLU-10           [-1, 128, 16, 16]               0
           Conv2d-11           [-1, 128, 16, 16]         147,584
      BatchNorm2d-12           [-1, 128, 16, 16]             256
             ReLU-13           [-1, 128, 16, 16]               0
        MaxPool2d-14             [-1, 128, 8, 8]               0
           Conv2d-15             [-1, 256, 8, 8]         295,168
      BatchNorm2d-16             [-1, 256, 8, 8]             512
             ReLU-17             [-1, 256, 8, 8]               0
           Conv2d-18             [-1, 256, 8, 8]         590,080
      BatchNorm2d-19             [-1, 256, 8, 8]             512
             ReLU-20             [-1, 256, 8, 8]               0
        MaxPool2d-21             [-1, 256, 4, 4]               0
           Conv2d-22             [-1, 512, 4, 4]       1,180,160
      BatchNorm2d-23             [-1, 512, 4, 4]           1,024
             ReLU-24             [-1, 512, 4, 4]               0
           Conv2d-25             [-1, 512, 4, 4]       2,359,808
      BatchNorm2d-26             [-1, 512, 4, 4]           1,024
             ReLU-27             [-1, 512, 4, 4]               0
        MaxPool2d-28             [-1, 512, 2, 2]               0
          Dropout-29                  [-1, 2048]               0
           Linear-30                  [-1, 1024]       2,098,176
             ReLU-31                  [-1, 1024]               0
          Dropout-32                  [-1, 1024]               0
           Linear-33                    [-1, 10]          10,250
================================================================
Total params: 6,797,642
Trainable params: 6,797,642
Non-trainable params: 0
```

Architecture of model B (ResNet18):

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1            [-1, 64, 32, 32]           1,728
       BatchNorm2d-2            [-1, 64, 32, 32]             128
```

ReLU-3	[-1, 64, 32, 32]	0
Identity-4	[-1, 64, 32, 32]	0
Conv2d-5	[-1, 64, 32, 32]	36,864
BatchNorm2d-6	[-1, 64, 32, 32]	128
ReLU-7	[-1, 64, 32, 32]	0
Conv2d-8	[-1, 64, 32, 32]	36,864
BatchNorm2d-9	[-1, 64, 32, 32]	128
ReLU-10	[-1, 64, 32, 32]	0
BasicBlock-11	[-1, 64, 32, 32]	0
Conv2d-12	[-1, 64, 32, 32]	36,864
BatchNorm2d-13	[-1, 64, 32, 32]	128
ReLU-14	[-1, 64, 32, 32]	0
Conv2d-15	[-1, 64, 32, 32]	36,864
BatchNorm2d-16	[-1, 64, 32, 32]	128
ReLU-17	[-1, 64, 32, 32]	0
BasicBlock-18	[-1, 64, 32, 32]	0
Conv2d-19	[-1, 128, 16, 16]	73,728
BatchNorm2d-20	[-1, 128, 16, 16]	256
ReLU-21	[-1, 128, 16, 16]	0
Conv2d-22	[-1, 128, 16, 16]	147,456
BatchNorm2d-23	[-1, 128, 16, 16]	256
Conv2d-24	[-1, 128, 16, 16]	8,192
BatchNorm2d-25	[-1, 128, 16, 16]	256
ReLU-26	[-1, 128, 16, 16]	0
BasicBlock-27	[-1, 128, 16, 16]	0
Conv2d-28	[-1, 128, 16, 16]	147,456
BatchNorm2d-29	[-1, 128, 16, 16]	256
ReLU-30	[-1, 128, 16, 16]	0
Conv2d-31	[-1, 128, 16, 16]	147,456
BatchNorm2d-32	[-1, 128, 16, 16]	256
ReLU-33	[-1, 128, 16, 16]	0
BasicBlock-34	[-1, 128, 16, 16]	0
Conv2d-35	[-1, 256, 8, 8]	294,912
BatchNorm2d-36	[-1, 256, 8, 8]	512
ReLU-37	[-1, 256, 8, 8]	0
Conv2d-38	[-1, 256, 8, 8]	589,824
BatchNorm2d-39	[-1, 256, 8, 8]	512
Conv2d-40	[-1, 256, 8, 8]	32,768
BatchNorm2d-41	[-1, 256, 8, 8]	512
ReLU-42	[-1, 256, 8, 8]	0
BasicBlock-43	[-1, 256, 8, 8]	0
Conv2d-44	[-1, 256, 8, 8]	589,824
BatchNorm2d-45	[-1, 256, 8, 8]	512
ReLU-46	[-1, 256, 8, 8]	0
Conv2d-47	[-1, 256, 8, 8]	589,824
BatchNorm2d-48	[-1, 256, 8, 8]	512
ReLU-49	[-1, 256, 8, 8]	0
BasicBlock-50	[-1, 256, 8, 8]	0
Conv2d-51	[-1, 512, 4, 4]	1,179,648
BatchNorm2d-52	[-1, 512, 4, 4]	1,024
ReLU-53	[-1, 512, 4, 4]	0
Conv2d-54	[-1, 512, 4, 4]	2,359,296
BatchNorm2d-55	[-1, 512, 4, 4]	1,024
Conv2d-56	[-1, 512, 4, 4]	131,072
BatchNorm2d-57	[-1, 512, 4, 4]	1,024

```
                ReLU-58               [-1, 512, 4, 4]                   0
           BasicBlock-59              [-1, 512, 4, 4]                   0
               Conv2d-60              [-1, 512, 4, 4]           2,359,296
          BatchNorm2d-61              [-1, 512, 4, 4]               1,024
                ReLU-62              [-1, 512, 4, 4]                   0
               Conv2d-63              [-1, 512, 4, 4]           2,359,296
          BatchNorm2d-64              [-1, 512, 4, 4]               1,024
                ReLU-65              [-1, 512, 4, 4]                   0
           BasicBlock-66              [-1, 512, 4, 4]                   0
AdaptiveAvgPool2d-67              [-1, 512, 1, 1]                   0
               Linear-68                    [-1, 10]               5,130
               ResNet-69                    [-1, 10]                   0
================================================================================
Total params: 11,173,962
Trainable params: 11,173,962
Non-trainable params: 0
```

The ResNet18 model contains multiple BasicBlock modules (lines 11, 18, 27, 34, 43, 50, 59, and 66), each implementing a residual connection. These blocks allow the network to learn more complex features while maintaining stable gradient flow during training, which is a major reason why ResNet18 achieved better performance compared to MyNet, despite the additional computational cost of having approximately 11.2M parameters versus MyNet's 6.8M parameters.

• Plot four learning curves (loss & accuracy) of the training process (train/validation) for both models. Total 8 plots. (8%)

Ans:

model A (MyNet):

model B (ResNet18):

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

Ans:

data augmentation:

- Random horizontal flipping with 50% probability
- Random rotation up to 15 degrees
- Color jittering (brightness, contrast, and saturation adjustments of up to 20%)
- Random affine transformations with small translations (up to 10%)

model architecture:

- Modified the first convolutional layer from the standard 7×7 kernel with stride 2 to a 3×3 kernel with stride 1 and padding 1. This adjustment better handles the small 32×32 images of CIFAR-10, as the original design was optimized for 224×224 ImageNet images.
- Replaced the maxpool layer with an Identity layer to prevent excessive downsampling of the already small input images.
- Used pretrained weights from ImageNet for transfer learning, which provided a strong initialization of the convolutional filters for detecting low-level features.

loss function:

- Used CrossEntropyLoss which combines LogSoftmax and NLLLoss in a single class, making it numerically more stable for multi-class classification tasks.