ELEC4542 Project Report

Name: Bao Xinyang

UID: 3035952989

Project: Image Classification

1. Introduction

   In this project, we aim to develop and evaluate an image classification model using the ResNet-18 architecture on the CIFAR-10 dataset. CIFAR-10 comprises 60,000 32x32 color images across 10 classes, making it a challenging benchmark for image classification tasks. ResNet-18, a deep convolutional neural network architecture, will serve as the backbone for our model, known for its effectiveness in various computer vision tasks.

2. Main content and discussion

   a) The aim of part a is using the ResNet-18 model in torchvision to train dataset CIFAR-10.

   By importing torchvision, I can directly create an instance of ResNet-18 by

```
ResNet18 = torchvision.models.resnet18()
```

   I use CrossEntropyLoss as loss function, which quantifies the difference between the predicted output and the actual target values. Stochastic Gradient Descent (SGD) is used as the optimizer, which is responsible for updating the model's parameters during the training process. The learning rate is set to 0.1, the momentum is set to 0.9, and weight decay is set to 5e-4.

```
# Define loss function and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.SGD(ResNet18.parameters(), lr=0.1, momentum=0.9, weight_decay=5e-4)
```

   I use learning rate scheduler with a step size of 50 and a gamma of 0.1. This will reduce the learning rate by a factor of 0.1 every 50 epochs, which is a common strategy to adapt the learning rate during training.

```
scheduler = optim.lr_scheduler.StepLR(optimizer, step_size=50, gamma=0.1)
```

   I train the network with 60 epochs. Although the more epochs are trained, the more accurate the result is, due to the limitation of time and GPU, I eventually choose 60 epochs. I also tried 30-50 epochs, but the accuracy did not achieve satisfied level (floating from 50% to 70%). The final accuracy on the test set is 79.44%.

   b) Part b is about using different data augmentation techniques to avoid overfitting. I have tried Random Flips and Rotations, Random Crop and Center Crop, as well as Normalization respectively.

   Random flips and rotations:

   In this transformation, I firstly randomly flip the input image horizontally with a default probability of 0.5, and then randomly rotated the image by a maximum angle of 30 degrees. These two transforms introduce variety into the training data, making the model more robust to different orientations of objects. By using these transforms, the final accuracy on the test set increased to 81.94%.

```
transform = transforms.Compose([
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(30),
    transforms.ToTensor(),
])
```

   Random Crop and Center Crop:

   In this transformation, I randomly crop on the input image to a size of 32x32 pixels. The padding=4 argument indicates that a padding of 4 pixels will be added to each side of the image before the crop. This is a form of data augmentation that helps the model become more robust by focusing on different parts of the input images during training. The final accuracy on the test set increased to 83.98%.

```
transform = transforms.Compose([
    transforms.RandomCrop(32, padding=4),
    transforms.ToTensor(),
])
```

Normalization:

In this transformation, I set the mean and std to (0.4914, 0.4822, 0.4465) and (0.247, 0.243, 0.261) respectively. They are corresponding to the mean and standard deviation of CIFAR-10 dataset.

```
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.4914, 0.4822, 0.4465],std=[0.247, 0.243, 0.261]),
])
```

However, after normalizing, the accuracy dramatically decreased to 30%, although the loss is keeping decrease. Normally, the accuracy should increase.

```
Validation Accuracy: 30.64%
Epoch 52/60: 100%|██████████| 391/391 [00:07<00:00, 55.37it/s]
Training Loss: 0.032054922957916544

Validation Accuracy: 32.38%
Epoch 53/60: 100%|██████████| 391/391 [00:07<00:00, 55.32it/s]
Training Loss: 0.015394367509857391

Validation Accuracy: 31.56%
Epoch 54/60: 100%|██████████| 391/391 [00:07<00:00, 55.44it/s]
Training Loss: 0.009395566143695732
```

I observed that the training lossis decreases dramatically. It may caused by overfitting. So I change the learning rate from 0.1 to 0.5. After that, the accuracy achieved to 76.37%

c) In this part, we are requested to use CAM to visualize the class activation maps of the trained models.

The label data is a list of 10,000 numbers ranging from 0 to 9, which corresponds to each of the 10 classes in CIFAR-10. (Airplane : 0, Automobile : 1, Bird : 2, Cat: 3, Deer : 4, Dog : 5, Frog : 6, Horse : 7, Ship : 8, Truck : 9)

The image we chosen should in the class we set for the targets.

```
targets = [ClassifierOutputTarget(0)]
```

```
image_path = test_dataset.data[23]
```

We can get different class activation maps by changing the layer we have chosen in the target_layer.

```
target_layers = [ResNet18.layer4]
```


(layer1)


(layer2)

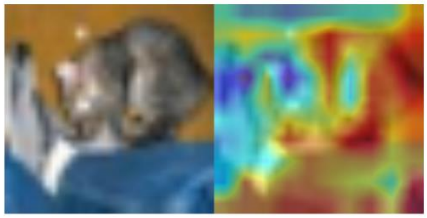

(layer3)

 (layer4)

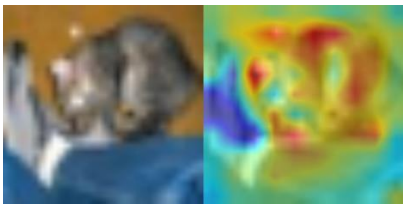
Different layers generate different CAM, that may because:

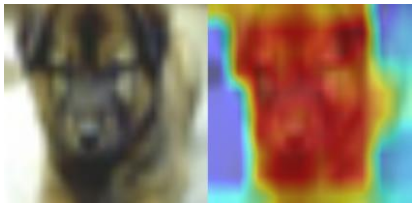
1. Deeper layers typically have lower spatial resolution but capture more abstract features, while shallower layers have higher spatial resolution but capture more fine-grained details. The choice of the target layer influences the spatial resolution of the CAM. Deeper layers may provide a more holistic view, while shallower layers may focus on finer details.
2. Different layers have different receptive fields, influencing the context in which information is captured. Deeper layers have larger receptive fields and capture more global context. The choice of target layer affects the contextual information considered when generating the CAM.

Initially, I used a ResNet-18 with 30 epochs training. But the result was not good. We can observe clearly that the red part on only focus on the cat, but also focus on the background.



Then I tried with more epochs, 50 and 60 respectively.
50 epochs:





In this case, we can observe a clearer region of an image that contribute the most to the prediction.

60 epochs:

 (accuracy=83.98%)

 (accuracy=79.44%)

These two output images are all generated by 60-epoch-trained ResNet-18. The first one is generated by ResNet-18 with rand crops, which has a higher accuracy (83.98%). The later one is generated by the modified model in part a, which has an accuracy equals to 79.44%. It can be easily observed that the first output, which is generated by the model with higher accuracy, has a clearer boundary of the object that we want.

3. Summary

This project aimed to develop, train, and evaluate an image classification model using the ResNet-18 on the CIFAR-10 dataset.

I achieved a test set accuracy of 79.44% of CIFAR-10 dataset after 60 epochs by using the ResNet-18 model.

Data augmentation techniques played a crucial role in improving the model's generalization ability. Random flips, rotations, and especially random crop with padding significantly enhanced the accuracy, reaching 83.98% on the test set. However, using data augmentation does not always reach a better accuracy. We observed a dramatically decrease accuracy after normalization hinted at potential overfitting. After adjusting the learning rate, the accuracy returned to a normal standard.

Class Activation Maps (CAM) were employed for visualizing regions of images contributing to predictions. When the target layer is different, the generated CAMs are different, which means they concentrate to different features. Models trained for 50 and 60 epochs, especially with data augmentation, provided clearer and more accurate CAM visualizations, emphasizing the importance of extended training duration.