[EEEE30026] Data Science

Model Compression

Po-Chuan, Chen
Student ID: 311511052
present90308.ee11@nycu.edu.tw

National Yang Ming Chiao Tung University
March 20, 2023

Contents

1 Methods

In this homework, I use **Knowledge Distillation** to compress the model.

Before training the model, I do the data augmentation to increase the number of training data. Such as random horizontal flip, rotation, translation, scaling.

And using scheduler to adjust the learning rate. The scheduler will reduce the learning rate by a factor of `factor` every `patience` epochs if the validation loss plateaus.

The student model is a simple CNN model with 3 convolutional layers and 2 fully connected layer.

The teacher model is provided by the TA. The student model is trained by both the teacher model and the ground truth.

After doing knowledge distillation, we can at most get the accuracy of `93.93%` and the number of parameters is less than `100,000`, which is `95,386`.

```
==========================================================================================
Layer (type:depth-idx)                   Output Shape              Param #
==========================================================================================
├─Conv2d: 1-1                            [-1, 32, 28, 28]          896
├─BatchNorm2d: 1-2                       [-1, 32, 28, 28]          64
├─Conv2d: 1-3                            [-1, 32, 28, 28]          9,248
├─BatchNorm2d: 1-4                       [-1, 32, 28, 28]          64
├─AvgPool2d: 1-5                         [-1, 32, 14, 14]          --
├─Conv2d: 1-6                            [-1, 32, 14, 14]          9,248
├─BatchNorm2d: 1-7                       [-1, 32, 14, 14]          64
├─AvgPool2d: 1-8                         [-1, 32, 7, 7]            --
├─Dropout: 1-9                           [-1, 1568]                --
├─Linear: 1-10                           [-1, 48]                  75,312
├─Linear: 1-11                           [-1, 10]                  490
==========================================================================================
Total params: 95,386
Trainable params: 95,386
Non-trainable params: 0
Total mult-adds (M): 9.78
==========================================================================================
Input size (MB): 0.01
Forward/backward pass size (MB): 0.86
Params size (MB): 0.36
Estimated Total Size (MB): 1.23
==========================================================================================
```

Figure 1: Student model summary

```
class StudentNet(nn.Module):
    def __init__(self):
        super(StudentNet, self).__init__()
       # 3 layers of convolutions, less than 100k parameters
        self.conv1 = nn.Conv2d(3, 32, 3, padding=1)
        self.conv2 = nn.Conv2d(32, 32, 3, padding=1)
        self.conv3 = nn.Conv2d(32, 32, 3, padding=1)
        self.bn1 = nn.BatchNorm2d(32)
        self.bn2 = nn.BatchNorm2d(32)
        self.bn3 = nn.BatchNorm2d(32)
        self.pool = nn.AvgPool2d(2, 2)
        self.fc1 = nn.Linear(32 * 7 * 7, 48)
        self.fc2 = nn.Linear(48, 10)
        self.dropout = nn.Dropout(0.35)

    def forward(self, x):
        x = self.conv1(x)
        x = self.bn1(x)
        x = F.relu(x)
        x = self.conv2(x)
        x = self.bn2(x)
        x = F.relu(x)
        x = self.pool(x)
        x = self.conv3(x)
        x = self.bn3(x)
        x = F.relu(x)
        x = self.pool(x)
        x = x.view(-1, 32 * 7 * 7)
        x = self.dropout(x)
        x = self.fc1(x)
        #x = self.dropout(x)
        x = self.fc2(x)
        return x
```

Figure 2: Student model

2 Reference

1. Fashion-MNIST
2. Resnet50 with fashion MNIST
3. ResNet50 knowledge distillation