System Analysis & Design

Final Project

Due: 18:00 on Tuesday, January 7, 2025

Image Classification: In this project, you will need to build a baseline Convolutional Neural Network (CNN) model to carry out image classification tasks.

You will also need to adopt some basic deep learning techniques, such as Dropout and Batch normalization, to try to improve performance of baseline model. You are asked to classify CIFAR-10 dataset.

The CIFAR-10 dataset consists of 60,000 32x32 colour images in 10 classes with 6,000 images per class. There are 50,000 training images and 10,000 test images.

The details of the CIFAR-10 dataset can be found at following website: https://www.cs.toronto.edu/~kriz/cifar.html.

You need to use **PyTorch** to build a baseline model. You can download the provided sample code files from MS Teams' File → Class material:

nutc_113-1_System Analysis & Design_final_data.zip

You may use sample code (ex10_cifar10_train.ipynb and ex10_cifar10_test.ipynb) files to finish this project (more details can be found in the code file).

You may use sample code (ex10_cifar10_train.ipynb) to train your baseline models only using training images and evaluate the validation accuracy to adjust your model.

You may also use sample code (ex10_cifar10_test.ipynb) to test model accuracy on the CIFAR-10 10,000 test images, and report the best accuracies for **each model**.

For a sample baseline model (cifar10_model.py), it requires following layers in your built model. You may add more layers in the sample baseline model. (Note that you are not allowed to delete any layers.)

cifar10_model.py is displayed below:

```
from torch import nn
import torch.nn.functional as F

class CNN(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1=nn.Conv2d(3,10,3,1,1)
        self.conv2=nn.Conv2d(10,20,3,1,1)
        self.conv3=nn.Conv2d(20,40,3,1,1)
        self.conv4=nn.Conv2d(40,80,3,1,1)
        self.pool=nn.MaxPool2d(2,2)
        self.linear1=nn.Linear(80*4*4,100)
        self.linear2=nn.Linear(100,100)
        self.linear3=nn.Linear(100,10)
        self.dropout=nn.Dropout(0.2)

    def forward(self,x):
        x=self.pool(F.relu(self.conv1(x))) # 10*16*16
        x=self.pool(F.relu(self.conv2(x))) # 20*8*8
        x=self.pool(F.relu(self.conv3(x))) # 40*4*4
        x=F.relu(self.conv4(x)) # 80*4*4
        x=x.view(-1,80*4*4)
        x=self.dropout(x)
        x=F.relu(self.linear1(x))
        x=F.relu(self.linear2(x))
        x=self.dropout(x)
        x=F.log_softmax(self.linear3(x),dim=1)
        return x
```

You need to add Dropout and Batch Normalization layers to improve accuracy by completing following experiments at least. You are also welcome to include more deep learning techniques such as attention, residual dense block, or transformer, or pre-trained models (cifar10_resnet.py is provided) to do more ablation experiments and add more results to the Table 1.

Table 1. Ablation Study on Cifar-10

Method	Accuracy
Baseline model (Need to modify cifar10_model.py)	
Baseline model + dropout (cifar10_model_drop.py)	
Baseline model + batch normalization (cifar10_model_bn.py)	
Baseline model + dropout & batch normalization (cifar10_model_drop_bn.py)	

For adding Batch Normalization, you may use following sample code:

```
class CNN(nn.Module):
  def __init__(self):
    super().__init__()
    self.conv1=nn.Conv2d(3,10,3,1,1)
    self.batch1= nn.NatchNorm2d(10)
    self.conv2=nn.Conv2d(10,20,3,1,1)
    self.batch2= nn.NatchNorm2d(20)
    self.conv3=nn.Conv2d(20,40,3,1,1)
    self.batch3= nn.NatchNorm2d(40)
    self.conv4=nn.Conv2d(40,80,3,1,1)
    self.pool=nn.MaxPool2d(2,2)
    self.linear1=nn.Linear(80*4*4,100)
    self.linear2=nn.Linear(100,100)
    self.linear3=nn.Linear(100,10)
    self.dropout=nn.Dropout(0.2)

  def forward(self,x):
    x=self.conv1(x)
    x=self.batch1(x)
    x=F.relu(x)
    x=self.pool(x) # 10*16*16
    x=self.conv2(x)
    x=self.batch2(x)
    x=F.relu(x)
    x=self.pool(x) # 20*8*8
    x=self.conv3(x)
```

```
x=self.batch3(x)
x=F.relu(x)
x=self.pool(x) # 40*4*4
x=F.relu(self.conv4(x)) # 80*4*4
x=x.view(-1,80*4*4)
x=self.dropout(x)
x=F.relu(self.linear1(x))
x=F.relu(self.linear2(x))
x=self.dropout(x)
x=F.log_softmax(self.linear3(x),dim=1)
return x
```

Submission guideline:

1. Your codes (at least 6 files: 4 models (.py), 1 train (.ipynb) and 1 test (.ipynb))
   s1410932033_王大明_cifar10_model.py
   s1410932033_王大明_cifar10_model_drop.py
   s1410932033_王大明_cifar10_model_bn.py
   s1410932033_王大明_cifar10_model_drop_bn.py
   s1410932033_王大明_cifar10_train.ipynb
   s1410932033_王大明_cifar10_test.ipynb
2. Your writing report (sid_name.docx ex. s1410932033_王大明.docx). Please remember to describe your methods and report their best accuracy in your writing report in order to get better grades.
3. Zip above files (.py, .ipynb, .docx) into ONE file such as s1410932033_王大明.zip and upload it to MS Teams Assignment.