1112 Deep Learning – Homework 2

Due: 11:59 pm, 12/12, 2023

For the following questions, please upload the source code to Moodle and explain the results in your report. Please submit your homework using the IPython Notebook (.ipynb), Python script (.py), and/or a PDF file (code needs to be turned in by .py or .ipynb files).
If your computer doesn't have a GPU, you can work on Google Colab.

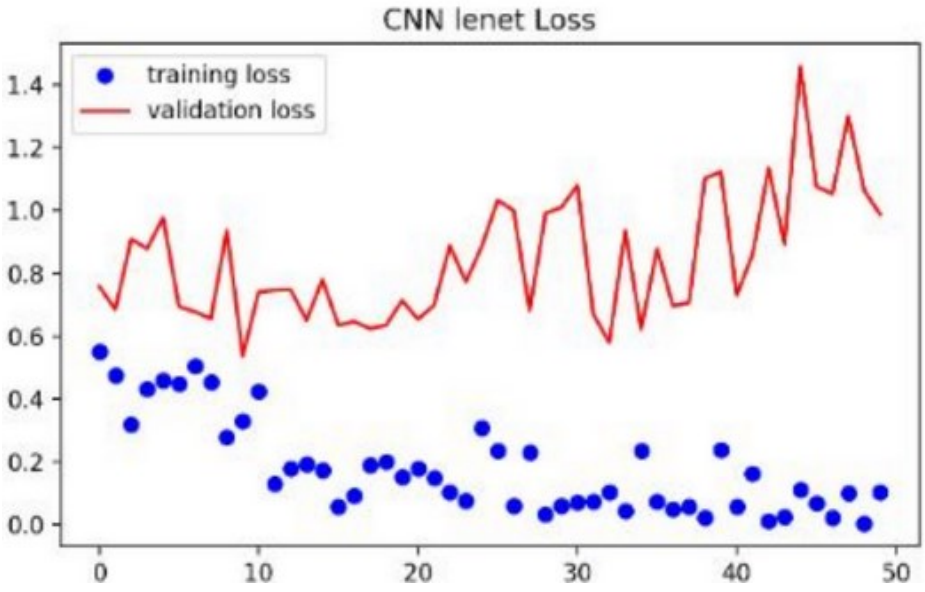
1. **Classification task (Cat and Dog):**
   Please download the dataset using the following link: https://drive.google.com/drive/folders/12J0JtSrqrHAjt2_olcB3tLVL6WIKlq5l?usp=sharing. The dataset includes two files: train.zip and test.zip.
   Utilize the training data to train two models, **AlexNet** and **ResNet**. After training, assess the performance of both models using the test data. Report the **accuracy** of the results obtained from testing. Additionally, please provide visualizations of the **training loss** changes for both models.

   For example:

LeNet5

```
[ ] class LeNet5(nn.Module):
        def __init__(self, num_classes=2):
            super(LeNet5, self).__init__()
            self.features = nn.Sequential(
                nn.Conv2d(3, 32, kernel_size=5),
                nn.ReLU(inplace=True),
                nn.MaxPool2d(kernel_size=2),
                nn.Conv2d(32, 64, kernel_size=5),
                nn.ReLU(inplace=True),
                nn.MaxPool2d(kernel_size=2),
                nn.Conv2d(64, 128, kernel_size=5),
                nn.ReLU(inplace=True),
                nn.MaxPool2d(kernel_size=2)
            )
            self.classifier = nn.Sequential(
                nn.Linear(128 * 12 * 12, 1024),
                nn.ReLU(inplace=True),
                nn.Dropout(0.5),
                nn.Linear(1024, 512),
                nn.ReLU(inplace=True),
                nn.Dropout(0.5),
                nn.Linear(512, num_classes)
            )

        def forward(self, x):
            x = self.features(x)
            # print(x.shape)
            x = torch.flatten(x, 1)
            x = self.classifier(x)
            return x
```



```
model = torch.load('/content/drive/MyDrive/CNN/dog_cat/checkpoints/best_LeNet.pth')
model.eval()
accuracy = []
correct = 0
total = 0
with torch.no_grad():
        for images, labels in test_dataloader:
                images = images.to(device)
                labels = labels.to(device)
                outputs = model(images)
                _, predicted = torch.max(outputs.data, 1)
                total += labels.size(0)
                correct += (predicted == labels).sum().item()

accuracy = 100 * correct / total
print('Accuracy: {:.2f}%'.format(accuracy))
```

```
Accuracy: 82.78%
```

2. **Classification Task (MNIST - Multiple Classes):**

   Please download the dataset using the following code:

```
[ ] import torch
    import torch.nn as nn
    from torch.utils.data import Dataset, DataLoader
    from torchvision import datasets
    import gzip
    import numpy as np

    # 下載 MNIST dataset
    full_dataset = datasets.MNIST(root='data', train=True, download=True)

    class MNISTDataset(Dataset):
        def __init__(self, data_file, label_file, transform=None):
            self.transform = transform

            # 讀取圖像資料
            with gzip.open(data_file, 'rb') as f:
                self.images = np.frombuffer(f.read(), np.uint8, offset=16).reshape(-1, 28, 28)

            # 讀取標籤資料
            with gzip.open(label_file, 'rb') as f:
                self.labels = np.frombuffer(f.read(), np.uint8, offset=8)

        # 查看照片總張數
        def __len__(self):
            return len(self.images)

        def __getitem__(self, idx):
            image = self.images[idx]
            label = self.labels[idx]

            image = np.reshape(image, (28, 28)) # 原始圖像大小為 28x28
            image = np.resize(image, (224, 224)) # 將圖像大小變成 224x224

            if self.transform:
                image = self.transform(image)

            return image, label
```

Follow the format of Question 1 and utilize the training data to train two different models, including **VGG** and a **CNN model you designed**. After training, assess the performance of both models using the test data. Report the **accuracy** of the results obtained from testing. Additionally, please provide visualizations of the **training loss** changes for both models.