Experiment 8 - MNIST Digit Classification using Keras

Problem Statement:

To build an image classifier with Keras and Convolutional Neural Networks for the Fashion MNIST dataset.

Objective:

Your task is to build an image classifier with Keras and Convolutional Neural Networks for the Fashion MNIST dataset. This data set includes 10 labels of different clothing types with 28 by 28 *grayscale* images. There is a training set of 60,000 images and 10,000 test images.

GitHub & Google Colab Link:

GitHub Link: https://github.com/piyush-gambhir/ncu-lab-manual-and-end-semester-projects/blob/main/NCU-CSL312%20-%20DL%20-%20Lab%20Manual/Experiment%208/Experiment%208.ipynb

Google Colab Link:

Open in Colab

Installing Dependencies:

```
In [ ]: ! pip install tabulate numpy pandas matplotlib seaborn torch torchvision
```

```
Requirement already satisfied: tabulate in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packa
ges (0.9.0)
Requirement already satisfied: numpy in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packages
(1.26.4)
Requirement already satisfied: pandas in c:\users\mainp\appdata\local\programs\python\python311\lib\site-package
s (2.2.2)
Requirement already satisfied: matplotlib in c:\users\mainp\appdata\local\programs\python\python311\lib\site-pac
kages (3.8.4)
Requirement already satisfied: seaborn in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packag
es (0.13.2)
Requirement already satisfied: torch in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packages
(2.3.0)
Requirement already satisfied: torchvision in c:\users\mainp\appdata\local\programs\python\python311\lib\site-pa
ckages (0.18.0)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\mainp\appdata\local\programs\python\python311\
lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\mainp\appdata\local\programs\python\python311\lib\site-p
ackages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\mainp\appdata\local\programs\python\python311\lib\site
-packages (from pandas) (2024.1)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\mainp\appdata\local\programs\python\python311\lib\si
te-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\mainp\appdata\local\programs\python\python311\lib\site-p
ackages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\mainp\appdata\local\programs\python\python311\lib\s
ite-packages (from matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\mainp\appdata\local\programs\python\python311\lib\s
ite-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in c:\users\mainp\appdata\local\programs\python\python311\lib\sit
e-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=8 in c:\users\mainp\appdata\local\programs\python\python311\lib\site-pack
ages (from matplotlib) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\mainp\appdata\local\programs\python\python311\lib\si
te-packages (from matplotlib) (3.1.2)
Requirement already satisfied: filelock in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packa
ges (from torch) (3.13.4)
Requirement already satisfied: typing-extensions>=4.8.0 in c:\users\mainp\appdata\local\programs\python\python31
1\lib\site-packages (from torch) (4.11.0)
Requirement already satisfied: sympy in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packages
(from torch) (1.12)
Requirement already satisfied: networkx in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packa
ges (from torch) (3.3)
Requirement already satisfied: jinja2 in c:\users\mainp\appdata\local\programs\python\python311\lib\site-package
s (from torch) (3.1.3)
Requirement already satisfied: fsspec in c:\users\mainp\appdata\local\programs\python\python311\lib\site-package
s (from torch) (2024.3.1)
Requirement already satisfied: mkl<=2021.4.0,>=2021.1.1 in c:\users\mainp\appdata\local\programs\python\python31
1\lib\site-packages (from torch) (2021.4.0)
Requirement already satisfied: intel-openmp==2021.* in c:\users\mainp\appdata\local\programs\python\python311\li
b\site-packages (from mkl<=2021.4.0,>=2021.1.1->torch) (2021.4.0)
Requirement already satisfied: tbb==2021.* in c:\users\mainp\appdata\local\programs\python\python311\lib\site-pa
ckages (from mkl<=2021.4.0,>=2021.1.1->torch) (2021.12.0)
Requirement already satisfied: six>=1.5 in c:\users\mainp\appdata\local\programs\python\python311\lib\site-packa
ges (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\mainp\appdata\local\programs\python\python311\lib\sit
e-packages (from jinja2->torch) (2.1.5)
Requirement already satisfied: mpmath>=0.19 in c:\users\mainp\appdata\local\programs\python\python311\lib\site-p
ackages (from sympy->torch) (1.3.0)
```

Code

In [ ]:

```
import numpy as np
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms
import matplotlib.pyplot as plt
from torch.utils.data.sampler import SubsetRandomSampler
from torch.utils.data import DataLoader
from collections import OrderedDict
```

In [ ]:

```
# Configuration
config = {
    'batch_size': 64,
    'n_epochs': 35,
    'lr': 0.0007,
    'dropout': 0.25,
    'input_size': 784,  # 28x28 images
    'hidden_sizes': [392, 196, 98, 49],
    'output_size': 10
}
```

In [ ]:
```
# Data Preparation
def load_data():
    transform = transforms.Compose([
        transforms.ToTensor(),
        transforms.Normalize((0.5,), (0.5,))
    ])
    train_ds = datasets.FashionMNIST('F_MNIST_data', download=True, train=True, transform=transform)
    test_ds = datasets.FashionMNIST('F_MNIST_data', download=True, train=False, transform=transform)

    # Split train set into training and validation set (80/20)
    num_train = len(train_ds)
    indices = list(range(num_train))
    np.random.shuffle(indices)
    split = int(np.floor(0.2 * num_train))
    train_idx, val_idx = indices[split:], indices[:split]

    # Creating data samplers and loaders
    train_sampler = SubsetRandomSampler(train_idx)
    val_sampler = SubsetRandomSampler(val_idx)

    train_dl = DataLoader(train_ds, batch_size=config['batch_size'], sampler=train_sampler)
    val_dl = DataLoader(train_ds, batch_size=config['batch_size'], sampler=val_sampler)
    test_dl = DataLoader(test_ds, batch_size=config['batch_size'], shuffle=True)

    return train_dl, val_dl, test_dl
```

In [ ]:
```
# Model Architecture
def build_network():
    layers = OrderedDict([
        ('fc1', nn.Linear(config['input_size'], config['hidden_sizes'][0])),
        ('relu1', nn.ReLU()),
        ('drop1', nn.Dropout(config['dropout'])),
        ('fc2', nn.Linear(config['hidden_sizes'][0], config['hidden_sizes'][1])),
        ('relu2', nn.ReLU()),
        ('drop2', nn.Dropout(config['dropout'])),
        ('fc3', nn.Linear(config['hidden_sizes'][1], config['hidden_sizes'][2])),
        ('relu3', nn.ReLU()),
        ('drop3', nn.Dropout(config['dropout'])),
        ('fc4', nn.Linear(config['hidden_sizes'][2], config['hidden_sizes'][3])),
        ('relu4', nn.ReLU()),
        ('output', nn.Linear(config['hidden_sizes'][3], config['output_size'])),
        ('logsoftmax', nn.LogSoftmax(dim=1))
    ])
    model = nn.Sequential(layers)
    device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
    model.to(device)
    return model, device
```

In [ ]:
```
# Training and Validation
def train_validate(model, device, train_dl, val_dl, n_epochs):
    loss_fn = nn.NLLLoss()
    optimizer = optim.Adam(model.parameters(), lr=config['lr'])
    train_losses, val_losses = [], []

    for epoch in range(n_epochs):
        model.train()
        total_train_loss = 0
        for images, labels in train_dl:
            images, labels = images.to(device), labels.to(device)
            images = images.view(images.shape[0], -1)
            optimizer.zero_grad()
            outputs = model(images)
            loss = loss_fn(outputs, labels)
            loss.backward()
            optimizer.step()
            total_train_loss += loss.item()

        avg_train_loss = total_train_loss / len(train_dl)
        train_losses.append(avg_train_loss)
        val_loss, val_acc = validate(model, device, val_dl, loss_fn)
        val_losses.append(val_loss)
        print(f'Epoch {epoch}: Train Loss: {avg_train_loss:.4f}, Val Loss: {val_loss:.4f}, Val Acc: {val_acc:.2

    plot_losses(train_losses, val_losses)
```

In [ ]:
```
def validate(model, device, loader, loss_fn):
    total_loss, total_correct = 0, 0
    model.eval()
    with torch.no_grad():
        for images, labels in loader:
            images, labels = images.to(device), labels.to(device)
            images = images.view(images.shape[0], -1)
            outputs = model(images)
            loss = loss_fn(outputs, labels)
            total_loss += loss.item()
            total_correct += (outputs.argmax(1) == labels).type(torch.float).sum().item()

    avg_loss = total_loss / len(loader)
    accuracy = 100 * total_correct / (len(loader) * config['batch_size'])
    return avg_loss, accuracy
```

In [ ]:

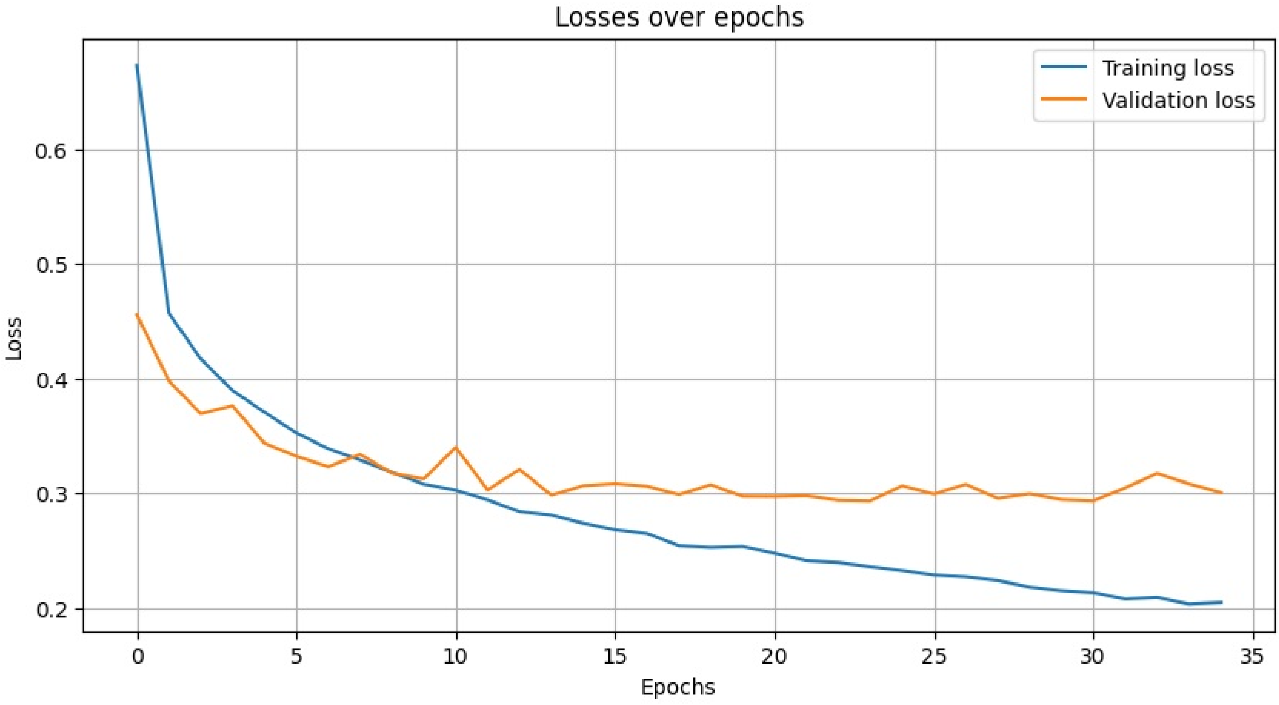
```
def plot_losses(train_losses, val_losses):
    plt.figure(figsize=(10, 5))
    plt.plot(train_losses, label='Training loss')
    plt.plot(val_losses, label='Validation loss')
    plt.title('Losses over epochs')
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend()
    plt.grid(True)
    plt.show()
```

In [ ]:

```
# Main
def main():
    train_dl, val_dl, test_dl = load_data()
    model, device = build_network()
    train_validate(model, device, train_dl, val_dl, config['n_epochs'])

if __name__ == '__main__':
    main()
```

```
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz to F_MNIST_dat
a\FashionMNIST\raw\train-images-idx3-ubyte.gz
100.0%
Extracting F_MNIST_data\FashionMNIST\raw\train-images-idx3-ubyte.gz to F_MNIST_data\FashionMNIST\raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz to F_MNIST_dat
a\FashionMNIST\raw\train-labels-idx1-ubyte.gz
100.0%
Extracting F_MNIST_data\FashionMNIST\raw\train-labels-idx1-ubyte.gz to F_MNIST_data\FashionMNIST\raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz to F_MNIST_data
\FashionMNIST\raw\t10k-images-idx3-ubyte.gz
100.0%
Extracting F_MNIST_data\FashionMNIST\raw\t10k-images-idx3-ubyte.gz to F_MNIST_data\FashionMNIST\raw

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz to F_MNIST_data
\FashionMNIST\raw\t10k-labels-idx1-ubyte.gz
100.0%
```

```
Extracting F_MNIST_data\FashionMNIST\raw\t10k-labels-idx1-ubyte.gz to F_MNIST_data\FashionMNIST\raw

Epoch 0: Train Loss: 0.6731, Val Loss: 0.4556, Val Acc: 83.71%
Epoch 1: Train Loss: 0.4570, Val Loss: 0.3980, Val Acc: 85.53%
Epoch 2: Train Loss: 0.4171, Val Loss: 0.3694, Val Acc: 86.74%
Epoch 3: Train Loss: 0.3895, Val Loss: 0.3762, Val Acc: 85.88%
Epoch 4: Train Loss: 0.3706, Val Loss: 0.3435, Val Acc: 87.28%
Epoch 5: Train Loss: 0.3525, Val Loss: 0.3322, Val Acc: 87.66%
Epoch 6: Train Loss: 0.3388, Val Loss: 0.3231, Val Acc: 88.18%
Epoch 7: Train Loss: 0.3290, Val Loss: 0.3338, Val Acc: 87.33%
Epoch 8: Train Loss: 0.3182, Val Loss: 0.3175, Val Acc: 88.33%
Epoch 9: Train Loss: 0.3079, Val Loss: 0.3125, Val Acc: 88.60%
Epoch 10: Train Loss: 0.3026, Val Loss: 0.3400, Val Acc: 87.74%
Epoch 11: Train Loss: 0.2941, Val Loss: 0.3029, Val Acc: 89.05%
Epoch 12: Train Loss: 0.2840, Val Loss: 0.3207, Val Acc: 88.74%
Epoch 13: Train Loss: 0.2808, Val Loss: 0.2983, Val Acc: 88.92%
Epoch 14: Train Loss: 0.2738, Val Loss: 0.3065, Val Acc: 89.10%
Epoch 15: Train Loss: 0.2682, Val Loss: 0.3083, Val Acc: 89.01%
Epoch 16: Train Loss: 0.2648, Val Loss: 0.3060, Val Acc: 89.06%
Epoch 17: Train Loss: 0.2543, Val Loss: 0.2988, Val Acc: 89.34%
Epoch 18: Train Loss: 0.2529, Val Loss: 0.3073, Val Acc: 89.47%
Epoch 19: Train Loss: 0.2536, Val Loss: 0.2972, Val Acc: 89.54%
Epoch 20: Train Loss: 0.2477, Val Loss: 0.2971, Val Acc: 89.51%
Epoch 21: Train Loss: 0.2412, Val Loss: 0.2977, Val Acc: 89.81%
Epoch 22: Train Loss: 0.2395, Val Loss: 0.2938, Val Acc: 89.40%
Epoch 23: Train Loss: 0.2359, Val Loss: 0.2933, Val Acc: 89.70%
Epoch 24: Train Loss: 0.2326, Val Loss: 0.3064, Val Acc: 89.49%
Epoch 25: Train Loss: 0.2287, Val Loss: 0.2993, Val Acc: 89.39%
Epoch 26: Train Loss: 0.2271, Val Loss: 0.3078, Val Acc: 89.43%
Epoch 27: Train Loss: 0.2238, Val Loss: 0.2955, Val Acc: 89.46%
Epoch 28: Train Loss: 0.2181, Val Loss: 0.2995, Val Acc: 89.36%

Epoch 29: Train Loss: 0.2149, Val Loss: 0.2944, Val Acc: 89.84%
Epoch 30: Train Loss: 0.2131, Val Loss: 0.2933, Val Acc: 90.28%
Epoch 31: Train Loss: 0.2077, Val Loss: 0.3046, Val Acc: 89.83%
Epoch 32: Train Loss: 0.2091, Val Loss: 0.3172, Val Acc: 89.39%
Epoch 33: Train Loss: 0.2033, Val Loss: 0.3081, Val Acc: 89.68%
Epoch 34: Train Loss: 0.2047, Val Loss: 0.3006, Val Acc: 89.79%
```


Losses over epochs
Training loss
Validation loss
Loss
Epochs