```
# MLP Project by - Sainath Vaddi

# Github repository link
# https://github.com/sainathvaddi/MLP


!pip install torch
!pip install torchvision
import torch
import torch.nn as nn
import torch.optim as optim
import torchvision
import torchvision.transforms as transforms
from sklearn.model_selection import KFold
from sklearn.preprocessing import MinMaxScaler
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import seaborn as sns
import torch.nn.functional as F
from torch.utils.data import DataLoader, SubsetRandomSampler
from torchvision import datasets, transforms
import time
from torch.optim.lr_scheduler import StepLR
```

```
Requirement already satisfied: torch in /usr/local/lib/python3.10/dist-packages (2.2.1+cu121)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch) (3.14.0)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch) (4.11.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch) (3.1.3)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch) (2023.6.0)
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (from torch)
  Using cached nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (23.7 MB)
Collecting nvidia-cuda-runtime-cu12==12.1.105 (from torch)
  Using cached nvidia_cuda_runtime_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (823 kB)
Collecting nvidia-cuda-cupti-cu12==12.1.105 (from torch)
  Using cached nvidia_cuda_cupti_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (14.1 MB)
Collecting nvidia-cudnn-cu12==8.9.2.26 (from torch)
  Using cached nvidia_cudnn_cu12-8.9.2.26-py3-none-manylinux1_x86_64.whl (731.7 MB)
Collecting nvidia-cublas-cu12==12.1.3.1 (from torch)
  Using cached nvidia_cublas_cu12-12.1.3.1-py3-none-manylinux1_x86_64.whl (410.6 MB)
Collecting nvidia-cufft-cu12==11.0.2.54 (from torch)
  Using cached nvidia_cufft_cu12-11.0.2.54-py3-none-manylinux1_x86_64.whl (121.6 MB)
Collecting nvidia-curand-cu12==10.3.2.106 (from torch)
  Using cached nvidia_curand_cu12-10.3.2.106-py3-none-manylinux1_x86_64.whl (56.5 MB)
Collecting nvidia-cusolver-cu12==11.4.5.107 (from torch)
  Using cached nvidia_cusolver_cu12-11.4.5.107-py3-none-manylinux1_x86_64.whl (124.2 MB)
Collecting nvidia-cusparse-cu12==12.1.0.106 (from torch)
  Using cached nvidia_cusparse_cu12-12.1.0.106-py3-none-manylinux1_x86_64.whl (196.0 MB)
Collecting nvidia-nccl-cu12==2.19.3 (from torch)
  Using cached nvidia_nccl_cu12-2.19.3-py3-none-manylinux1_x86_64.whl (166.0 MB)
Collecting nvidia-nvtx-cu12==12.1.105 (from torch)
  Using cached nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl (99 kB)
Requirement already satisfied: triton==2.2.0 in /usr/local/lib/python3.10/dist-packages (from torch) (2.2.0)
Collecting nvidia-nvjitlink-cu12 (from nvidia-cusolver-cu12==11.4.5.107->torch)
  Using cached nvidia_nvjitlink_cu12-12.4.127-py3-none-manylinux2014_x86_64.whl (21.1 MB)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch) (2.1.5)
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-packages (from sympy->torch) (1.3.0)
Installing collected packages: nvidia-nvtx-cu12, nvidia-nvjitlink-cu12, nvidia-nccl-cu12, nvidia-curand-cu12, nvidia-cufft-cu12, nvid
Successfully installed nvidia-cublas-cu12-12.1.3.1 nvidia-cuda-cupti-cu12-12.1.105 nvidia-cuda-nvrtc-cu12-12.1.105 nvidia-cuda-runtim
Requirement already satisfied: torchvision in /usr/local/lib/python3.10/dist-packages (0.17.1+cu121)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from torchvision) (1.25.2)
Requirement already satisfied: torch==2.2.1 in /usr/local/lib/python3.10/dist-packages (from torchvision) (2.2.1+cu121)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in /usr/local/lib/python3.10/dist-packages (from torchvision) (9.4.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (3.14.0)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (3.1.3)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (2023.6.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchv
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torc
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchv
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvisio
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvisio
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvis
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchv
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchv
```

```
Requirement already satisfied: nvidia-nccl-cu12==2.19.3 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision) (
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch==2.2.1->torchvision)
```

```
batch = 64
epochs = 20
device = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
device
```

```
device(type='cuda', index=0)
```

```
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize((0.5,), (0.5,))
])

trainset = torchvision.datasets.MNIST(root='./data', train=True, download=True, transform=transform)
valid_dataset = datasets.MNIST(root='./data', train=True, transform=transform)
testset = torchvision.datasets.MNIST(root='./data', train=False, download=True, transform=transform)
```

```
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz
Failed to download (trying next):
HTTP Error 403: Forbidden

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-images-idx3-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-images-idx3-ubyte.gz to ./data/MNIST/raw/train-images-idx3-ubyte.gz
100%|██████████| 9912422/9912422 [00:00<00:00, 38691528.63it/s]
Extracting ./data/MNIST/raw/train-images-idx3-ubyte.gz to ./data/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz
Failed to download (trying next):
HTTP Error 403: Forbidden

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-labels-idx1-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/train-labels-idx1-ubyte.gz to ./data/MNIST/raw/train-labels-idx1-ubyte.gz
100%|██████████| 28881/28881 [00:00<00:00, 1167966.97it/s]
Extracting ./data/MNIST/raw/train-labels-idx1-ubyte.gz to ./data/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz
Failed to download (trying next):
HTTP Error 403: Forbidden

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-images-idx3-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-images-idx3-ubyte.gz to ./data/MNIST/raw/t10k-images-idx3-ubyte.gz
100%|██████████| 1648877/1648877 [00:00<00:00, 9500986.24it/s]
Extracting ./data/MNIST/raw/t10k-images-idx3-ubyte.gz to ./data/MNIST/raw

Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz
Failed to download (trying next):
HTTP Error 403: Forbidden

Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-labels-idx1-ubyte.gz
Downloading https://ossci-datasets.s3.amazonaws.com/mnist/t10k-labels-idx1-ubyte.gz to ./data/MNIST/raw/t10k-labels-idx1-ubyte.gz
100%|██████████| 4542/4542 [00:00<00:00, 8627956.87it/s]Extracting ./data/MNIST/raw/t10k-labels-idx1-ubyte.gz to ./data/MNIST/raw
```

```
validation_fraction = 0.1
num = int(validation_fraction * 60000)
train_indices = torch.arange(0, 60000 - num)
valid_indices = torch.arange(60000 - num, 60000)
train_sampler = SubsetRandomSampler(train_indices)
valid_sampler = SubsetRandomSampler(valid_indices)
```

```
train_loader = DataLoader(dataset=trainset, batch_size=batch, drop_last=True, sampler=train_sampler)
valid_loader = DataLoader(dataset=valid_dataset, batch_size=batch, sampler=valid_sampler)
test_loader = DataLoader(dataset=testset, batch_size=batch, shuffle=False)
```

```
for images, labels in train_loader:
  print(images[0].shape, labels[0])
  break
```

```
torch.Size([1, 28, 28]) tensor(8)
```

```
# Display a grid of sample images
plt.figure(figsize=(10, 10))
for i, (images, labels) in enumerate(train_loader):
```

```
for i, (images, labels) in enumerate(train_loader):
  for j in range(25):
    plt.subplot(5, 5, j + 1)
    plt.imshow(images[j].squeeze(), cmap='gray')
    plt.axis('off')
    break
plt.show()
```

```
class MLP(nn.Module):
  def __init__(self, num_features, num_hidden_1, num_hidden_2, num_classes):
    super().__init__()
    self.network = torch.nn.Sequential(
    # 1st hidden layer
    torch.nn.Flatten(),
    torch.nn.Linear(num_features, num_hidden_1),
    torch.nn.BatchNorm1d(num_hidden_1),
    torch.nn.ReLU(),
    torch.nn.Dropout(0.5),
  # 2nd hidden layer
  torch.nn.Linear(num_hidden_1, num_hidden_2),
  torch.nn.BatchNorm1d(num_hidden_2),
  torch.nn.ReLU(),
  torch.nn.Dropout(0.3),
  # output layer
  torch.nn.Linear(num_hidden_2, num_classes)
)

  def forward(self, x):
    logits = self.network(x)
    return logits


model = MLP(num_features=28*28,
num_hidden_1=128,
num_hidden_2=64,
num_classes=10)
model = model.to(device)
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.SGD(model.parameters(), lr=0.1, momentum=0.9,
weight_decay=0.0001)
scheduler = torch.optim.lr_scheduler.ReduceLROnPlateau(optimizer,
factor=0.1,
mode='min')


# Define optimizer
optimizer = optim.Adam(model.parameters(), lr=0.001)


# Scheduler with step decay
scheduler = StepLR(optimizer, step_size=30, gamma=0.1)



def compute_accuracy(data_loader):
  with torch.no_grad():
    correct_pred, num_examples = 0, 0
    for i, (features, targets) in enumerate(data_loader):
      features = features.to(device)
      targets = targets.float().to(device)
      logits = model(features)
      _, predicted_labels = torch.max(logits, 1)
      num_examples += targets.size(0)
      correct_pred += (predicted_labels == targets).sum()
  return correct_pred.float()/num_examples * 100
```

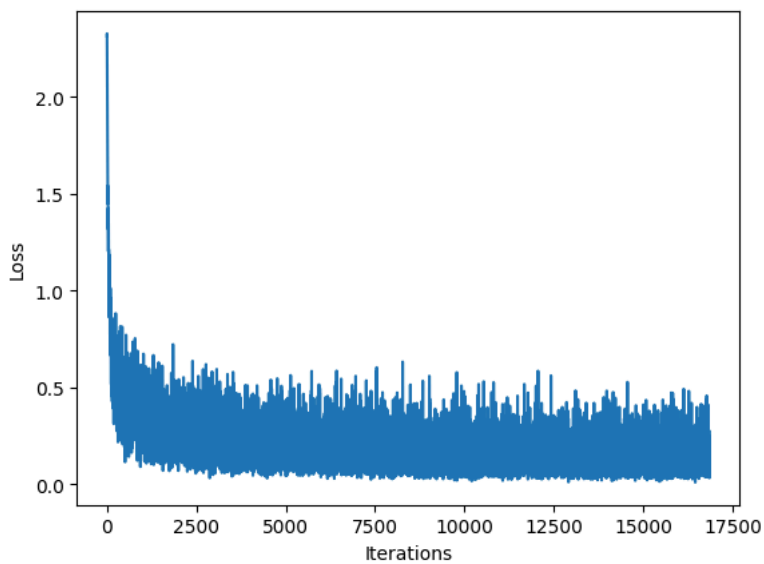
```
start_time = time.time()
minibatch_loss_list, train_acc_list, valid_acc_list = [], [], []
for epoch in range(epochs):
  model.train()
  for batch_idx, (features, targets) in enumerate(train_loader):
    features = features.to(device)
    targets = targets.to(device)
    # ## FORWARD AND BACK PROP
    logits = model(features)
    #loss = F.cross_entropy(logits, targets)
    loss = criterion(logits, targets)
    optimizer.zero_grad()
    loss.backward()
    # ## UPDATE MODEL PARAMETERS
    optimizer.step()
    # ## LOGGING
    minibatch_loss_list.append(loss.item())
    logging_interval = 100
    if not batch_idx % logging_interval:
      print("Epoch: ", epoch+1,"/", epochs,"| Batch ",batch_idx,
    "/",len(train_loader), f'| Loss: {loss:.4f}')
  model.eval()
  with torch.no_grad(): # save memory during inference
    train_acc = compute_accuracy(train_loader)
    valid_acc = compute_accuracy(valid_loader)
    print("Epoch: ", epoch+1, "/",epochs, "completed",
    f'| Train: {train_acc :.2f}% '
    f'| Validation: {valid_acc :.2f}%')
    train_acc_list.append(train_acc.item())
    valid_acc_list.append(valid_acc.item())
elapsed = (time.time() - start_time)/60
print("Time elapsed: ",elapsed, " min")
scheduler.step(minibatch_loss_list[-1])
elapsed = (time.time() - start_time)/60
print(f'Total Training Time: {elapsed:.2f} min')
test_acc = compute_accuracy(test_loader)
print(f'Test accuracy {test_acc :.2f}%')
```

```
Epoch:  1 / 20 | Batch  0 / 843 | Loss: 2.3129
Epoch:  1 / 20 | Batch  100 / 843 | Loss: 0.6639
Epoch:  1 / 20 | Batch  200 / 843 | Loss: 0.6327
Epoch:  1 / 20 | Batch  300 / 843 | Loss: 0.4612
Epoch:  1 / 20 | Batch  400 / 843 | Loss: 0.4147
Epoch:  1 / 20 | Batch  500 / 843 | Loss: 0.1736
Epoch:  1 / 20 | Batch  600 / 843 | Loss: 0.3688
Epoch:  1 / 20 | Batch  700 / 843 | Loss: 0.1921
Epoch:  1 / 20 | Batch  800 / 843 | Loss: 0.3725
Epoch:  1 / 20 completed | Train: 94.58% | Validation: 95.87%
Epoch:  2 / 20 | Batch  0 / 843 | Loss: 0.2415
Epoch:  2 / 20 | Batch  100 / 843 | Loss: 0.5284
Epoch:  2 / 20 | Batch  200 / 843 | Loss: 0.3307
Epoch:  2 / 20 | Batch  300 / 843 | Loss: 0.3839
Epoch:  2 / 20 | Batch  400 / 843 | Loss: 0.2891
Epoch:  2 / 20 | Batch  500 / 843 | Loss: 0.3198
Epoch:  2 / 20 | Batch  600 / 843 | Loss: 0.4130
Epoch:  2 / 20 | Batch  700 / 843 | Loss: 0.2191
Epoch:  2 / 20 | Batch  800 / 843 | Loss: 0.2663
Epoch:  2 / 20 completed | Train: 96.09% | Validation: 96.63%
Epoch:  3 / 20 | Batch  0 / 843 | Loss: 0.2278
Epoch:  3 / 20 | Batch  100 / 843 | Loss: 0.0984
Epoch:  3 / 20 | Batch  200 / 843 | Loss: 0.3246
Epoch:  3 / 20 | Batch  300 / 843 | Loss: 0.2178
Epoch:  3 / 20 | Batch  400 / 843 | Loss: 0.4058
Epoch:  3 / 20 | Batch  500 / 843 | Loss: 0.1555
Epoch:  3 / 20 | Batch  600 / 843 | Loss: 0.2299
Epoch:  3 / 20 | Batch  700 / 843 | Loss: 0.2520
Epoch:  3 / 20 | Batch  800 / 843 | Loss: 0.1504
Epoch:  3 / 20 completed | Train: 96.80% | Validation: 97.28%
Epoch:  4 / 20 | Batch  0 / 843 | Loss: 0.3003
Epoch:  4 / 20 | Batch  100 / 843 | Loss: 0.2204
Epoch:  4 / 20 | Batch  200 / 843 | Loss: 0.2659
Epoch:  4 / 20 | Batch  300 / 843 | Loss: 0.5489
Epoch:  4 / 20 | Batch  400 / 843 | Loss: 0.1914
Epoch:  4 / 20 | Batch  500 / 843 | Loss: 0.4866
Epoch:  4 / 20 | Batch  600 / 843 | Loss: 0.1309
Epoch:  4 / 20 | Batch  700 / 843 | Loss: 0.3487
Epoch:  4 / 20 | Batch  800 / 843 | Loss: 0.1976
Epoch:  4 / 20 completed | Train: 97.29% | Validation: 97.65%
Epoch:  5 / 20 | Batch  0 / 843 | Loss: 0.5672
Epoch:  5 / 20 | Batch  100 / 843 | Loss: 0.2858
Epoch:  5 / 20 | Batch  200 / 843 | Loss: 0.1176
```

```
Epoch:  5 / 20 | Batch  300 / 843 | Loss: 0.1573
Epoch:  5 / 20 | Batch  400 / 843 | Loss: 0.1781
Epoch:  5 / 20 | Batch  500 / 843 | Loss: 0.2880
Epoch:  5 / 20 | Batch  600 / 843 | Loss: 0.2927
Epoch:  5 / 20 | Batch  700 / 843 | Loss: 0.1496
Epoch:  5 / 20 | Batch  800 / 843 | Loss: 0.1931
Epoch:  5 / 20 completed | Train: 97.48% | Validation: 97.53%
Epoch:  6 / 20 | Batch  0 / 843 | Loss: 0.1583
Epoch:  6 / 20 | Batch  100 / 843 | Loss: 0.2131
Epoch:  6 / 20 | Batch  200 / 843 | Loss: 0.2197
Epoch:  6 / 20 | Batch  300 / 843 | Loss: 0.2703
Epoch:  6 / 20 | Batch  400 / 843 | Loss: 0.2222
Epoch:  6 / 20 | Batch  500 / 843 | Loss: 0.0785
Epoch:  6 / 20 | Batch  600 / 843 | Loss: 0.1458
```

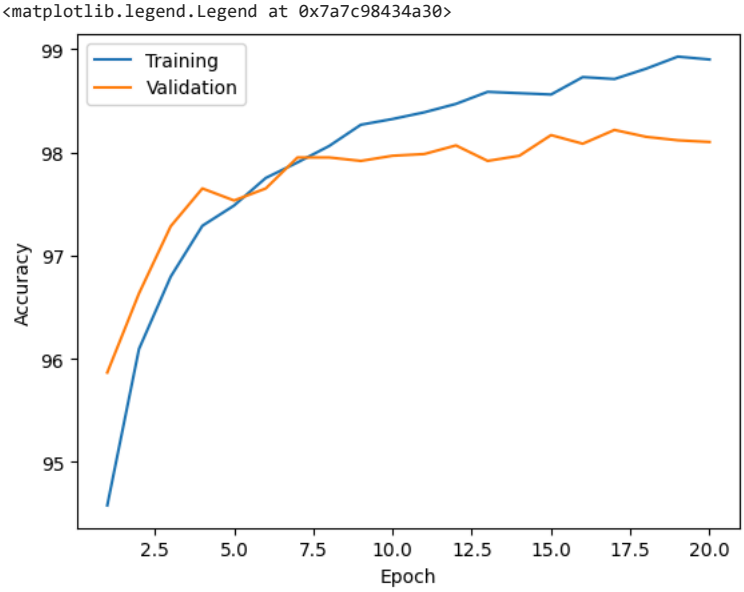
```
plt.plot(range(len(minibatch_loss_list)), minibatch_loss_list)
plt.xlabel('Iterations')
plt.ylabel('Loss')
```

```
Text(0, 0.5, 'Loss')
```

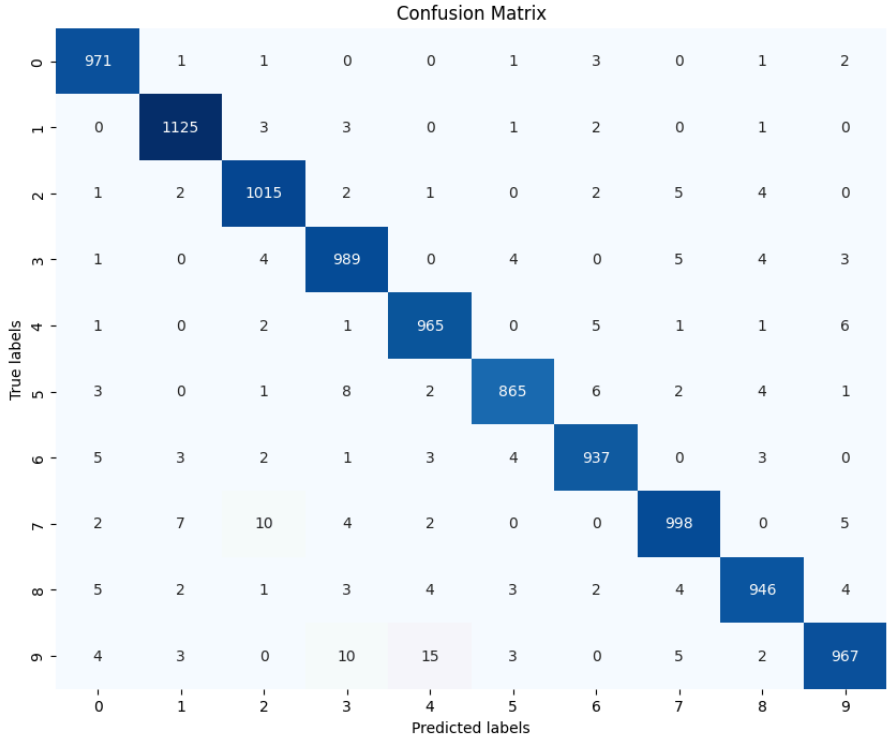



```
num_epochs = len(train_acc_list)
plt.plot(np.arange(1, num_epochs+1),
train_acc_list, label='Training')
plt.plot(np.arange(1, num_epochs+1),
valid_acc_list, label='Validation')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7a7c98434a30>
```



```
# Set the model to evaluation
model.eval()

all_predictions = []
all_targets = []

# Iterate over the test dataset
for features, targets in test_loader:
    # Move data to the device
    features = features.to(device)
    targets = targets.to(device)

    # Forward pass
    logits = model(features)

    # Get predictions
    _, predicted_labels = torch.max(logits, 1)

    # Append predictions and actual labels to the lists
    all_predictions.extend(predicted_labels.cpu().numpy())
    all_targets.extend(targets.cpu().numpy())

# Convert lists to numpy arrays
all_predictions = np.array(all_predictions)
all_targets = np.array(all_targets)

# Generate confusion matrix
conf_matrix = confusion_matrix(all_targets, all_predictions)
classes = range(10)

# Display confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=classes, yticklabels=classes, cbar=False)
plt.xlabel("Predicted labels")
plt.ylabel("True labels")
plt.title("Confusion Matrix")
plt.show()
```



```
from sklearn.metrics import classification_report

# Classification report
report = classification_report(all_targets, all_predictions, target_names=[str(i) for i in range(10)])

# Print classification report
print(report)
```

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       980
           1       0.98      0.99      0.99      1135
           2       0.98      0.98      0.98      1032
           3       0.97      0.98      0.97      1010
           4       0.97      0.98      0.98       982
           5       0.98      0.97      0.98       892
           6       0.98      0.98      0.98       958
           7       0.98      0.97      0.97      1028
           8       0.98      0.97      0.98       974
           9       0.98      0.96      0.97      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000
```