colab.research.google.com/drive/1vFso2bmRqpPXXcJSTfBkjg2nc8Ku7NH

Lab4.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text ▷ Run all

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

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

transform = transforms.ToTensor()
train_dataset = datasets.MNIST(root='./data', train=True, transform=transform, download=True)
test_dataset = datasets.MNIST(root='./data', train=False, transform=transform)

train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=32, shuffle=False)

class FFNN(nn.Module):
    def __init__(self):
        super(FFNN, self).__init__()
        self.flatten = nn.Flatten()
        self.fc1 = nn.Linear(28*28, 128)
        self.relu = nn.ReLU()
        self.fc2 = nn.Linear(128, 10)

    def forward(self, x):
        x = self.flatten(x)
        x = self.relu(self.fc1(x))
        x = self.fc2(x)
        return x
```

What can I help you build?

Share Gemini

RAM Disk

Lab4.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Run all

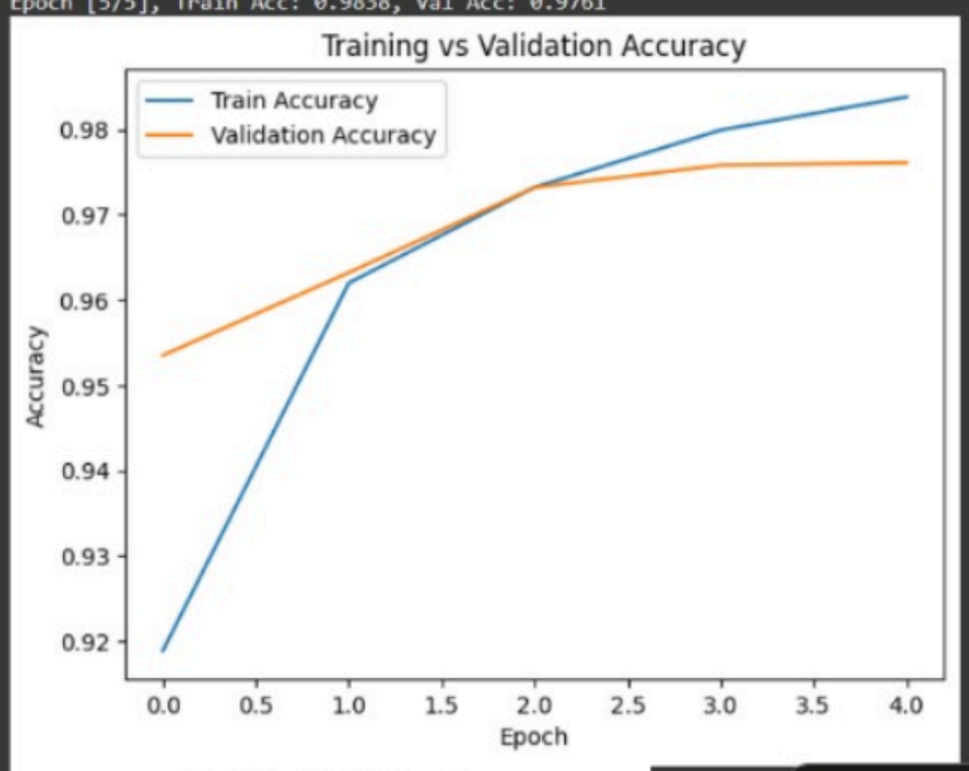
RAM Disk

```
# 4. Loss and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters())

# 5. Train the model
num_epochs = 5
train_acc_history = []
val_acc_history = []

for epoch in range(num_epochs):
    model.train()
    correct, total = 0, 0
    for images, labels in train_loader:
        images, labels = images.to(device), labels.to(device)

        outputs = model(images)
        loss = criterion(outputs, labels)

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

    train_accuracy = correct / total
    train_acc_history.append(train_accuracy)

    # Validation
    model.eval()
    correct, total = 0, 0
    with torch.no_grad():
        for images, labels in test_loader:
```

What can I help you build?

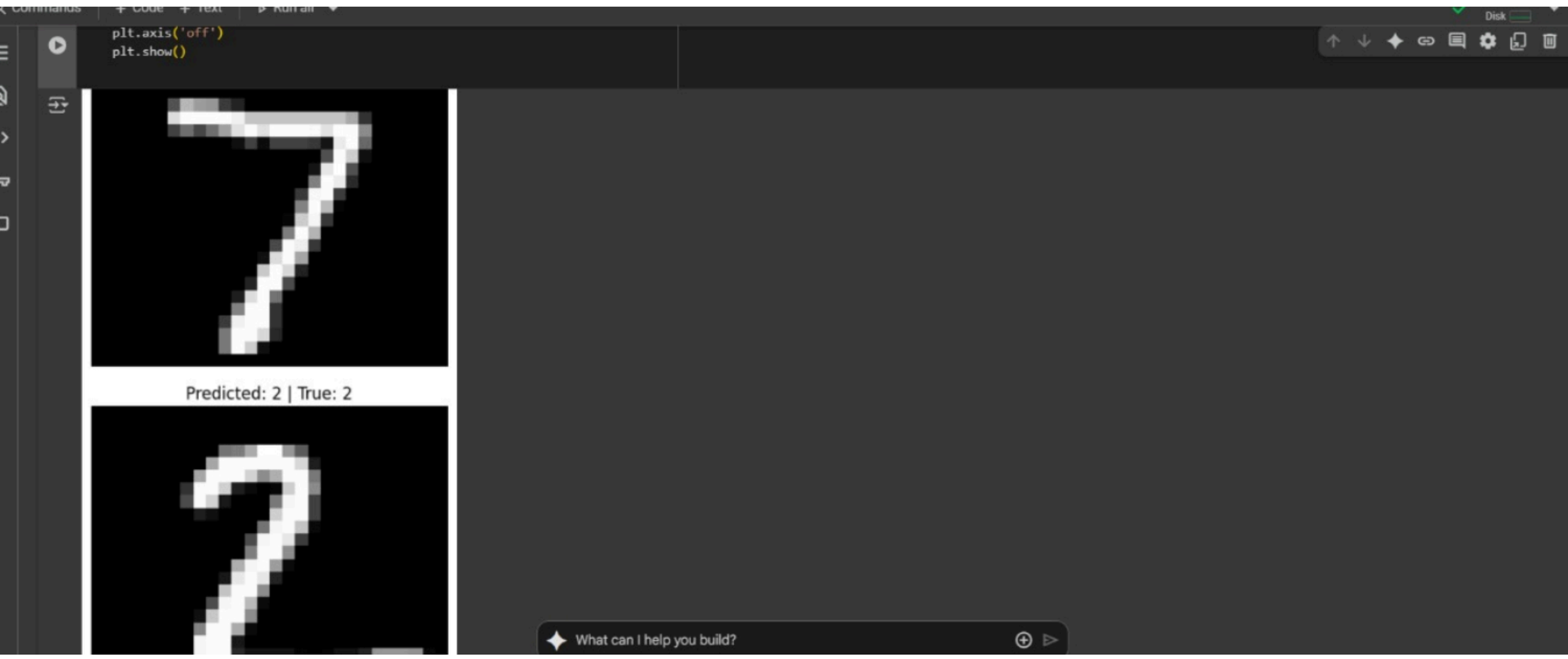
```
    plt.axis('off')
    plt.show()
```

```
Epoch [1/5], Train Acc: 0.9189, Val Acc: 0.9535
Epoch [2/5], Train Acc: 0.9620, Val Acc: 0.9632
Epoch [3/5], Train Acc: 0.9732, Val Acc: 0.9732
Epoch [4/5], Train Acc: 0.9799, Val Acc: 0.9758
Epoch [5/5], Train Acc: 0.9838, Val Acc: 0.9761
```



Predicted: 7 | True: 7

What can I help you build?


plt.axis('off')
plt.show()
Predicted: 2 | True: 2
What can I help you build?