```
!pip install medmnist
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting medmnist
  Downloading medmnist-2.2.1-py3-none-any.whl (21 kB)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.9/dist-packages (from medmnist) (1.2.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.9/dist-packages (from medmnist) (1.5.3)
Requirement already satisfied: torchvision in /usr/local/lib/python3.9/dist-packages (from medmnist) (0.15.1+cu118)
Requirement already satisfied: torch in /usr/local/lib/python3.9/dist-packages (from medmnist) (2.0.0+cu118)
Collecting fire
  Downloading fire-0.5.0.tar.gz (88 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 88.3/88.3 kB 5.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: Pillow in /usr/local/lib/python3.9/dist-packages (from medmnist) (8.4.0)
Requirement already satisfied: scikit-image in /usr/local/lib/python3.9/dist-packages (from medmnist) (0.19.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.9/dist-packages (from medmnist) (4.65.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.9/dist-packages (from medmnist) (1.22.4)
Requirement already satisfied: six in /usr/local/lib/python3.9/dist-packages (from fire->medmnist) (1.16.0)
Requirement already satisfied: termcolor in /usr/local/lib/python3.9/dist-packages (from fire->medmnist) (2.2.0)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.9/dist-packages (from pandas->medmnist) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.9/dist-packages (from pandas->medmnist) (2022.7.1)
Requirement already satisfied: PyWavelets>=1.1.1 in /usr/local/lib/python3.9/dist-packages (from scikit-image->medmnist) (1.4.1)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.9/dist-packages (from scikit-image->medmnist) (23.1)
Requirement already satisfied: networkx>=2.2 in /usr/local/lib/python3.9/dist-packages (from scikit-image->medmnist) (3.1)
Requirement already satisfied: tifffile>=2019.7.26 in /usr/local/lib/python3.9/dist-packages (from scikit-image->medmnist) (2023.4.12)
Requirement already satisfied: imageio>=2.4.1 in /usr/local/lib/python3.9/dist-packages (from scikit-image->medmnist) (2.25.1)
Requirement already satisfied: scipy>=1.4.1 in /usr/local/lib/python3.9/dist-packages (from scikit-image->medmnist) (1.10.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.9/dist-packages (from scikit-learn->medmnist) (3.1.0)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.9/dist-packages (from scikit-learn->medmnist) (1.2.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.9/dist-packages (from torch->medmnist) (1.11.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.9/dist-packages (from torch->medmnist) (3.1.2)
Requirement already satisfied: triton==2.0.0 in /usr/local/lib/python3.9/dist-packages (from torch->medmnist) (2.0.0)
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.9/dist-packages (from torch->medmnist) (4.5.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.9/dist-packages (from torch->medmnist) (3.11.0)
Requirement already satisfied: lit in /usr/local/lib/python3.9/dist-packages (from triton==2.0.0->torch->medmnist) (16.0.1)
Requirement already satisfied: cmake in /usr/local/lib/python3.9/dist-packages (from triton==2.0.0->torch->medmnist) (3.25.2)
Requirement already satisfied: requests in /usr/local/lib/python3.9/dist-packages (from torchvision->medmnist) (2.27.1)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.9/dist-packages (from jinja2->torch->medmnist) (2.1.2)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.9/dist-packages (from requests->torchvision->medmnist) (1.26.15)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.9/dist-packages (from requests->torchvision->medmnist) (3.4)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.9/dist-packages (from requests->torchvision->medmnist) (2.0.12)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.9/dist-packages (from requests->torchvision->medmnist) (2022.12.7)
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.9/dist-packages (from sympy->torch->medmnist) (1.3.0)
Building wheels for collected packages: fire
  Building wheel for fire (setup.py) ... done
  Created wheel for fire: filename=fire-0.5.0-py2.py3-none-any.whl size=116952 sha256=7273f6f8e20954387b9ecad0144296ec82eca67649487ee1310162c2fdf789c5
  Stored in directory: /root/.cache/pip/wheels/f7/f1/89/b9ea2bf8f80ec027a88fef1d354b3816b4d3d29530988972f6
Successfully built fire
Installing collected packages: fire, medmnist
Successfully installed fire-0.5.0 medmnist-2.2.1
```

```python
from tqdm import tqdm
import numpy as np
import torch
import torch.nn as nn
import torch.optim as optim
import torch.utils.data as data
import torchvision.transforms as transforms
import medmnist
from medmnist import INFO, Evaluator
```

```
data_flag = 'pathmnist'
# Data_flag = 'breastmnist'
download = True

NUM_EPOCHS = 3
BATCH_SIZE = 128
lr = 0.001

info = INFO[data_flag]
task = info['task']
n_channels = info['n_channels']
n_classes = len(info['label'])

DataClass = getattr(medmnist, info['python_class'])
```

```
data_transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize(mean=[.5], std=[.5])
])
```

```
print(train_dataset)
print("===================")
print(test_dataset)
```

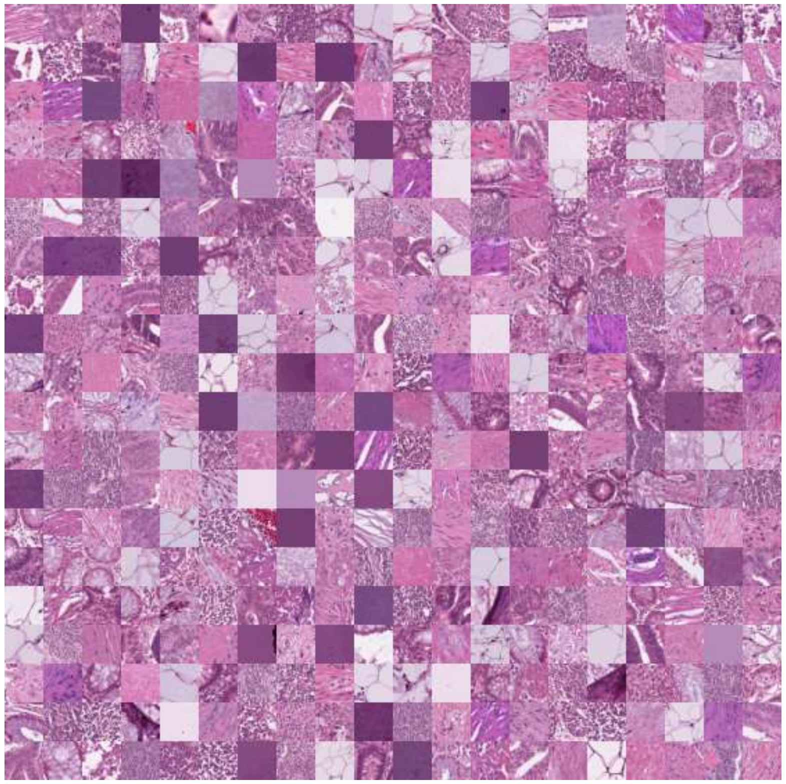
```
Dataset PathMNIST (pathmnist)
    Number of datapoints: 89996
    Root location: /home/three/.medmnist
    Split: train
    Task: multi-class
    Number of channels: 3
    Meaning of labels: {'0': 'adipose', '1': 'background', '2': 'debris', '3': 'lymphocytes', '4': 'mucus', '5': 'smooth muscle', '6': 'normal colon mucosa', '7': 'cancer-assoc
    Number of samples: {'train': 89996, 'val': 10004, 'test': 7180}
    Description: The PathMNIST is based on a prior study for predicting survival from colorectal cancer histology slides, providing a dataset (NCT-CRC-HE-100K) of 100,000 non-c
    License: CC BY 4.0
===================
Dataset PathMNIST (pathmnist)
    Number of datapoints: 7180
    Root location: /home/three/.medmnist
    Split: test
    Task: multi-class
    Number of channels: 3
    Meaning of labels: {'0': 'adipose', '1': 'background', '2': 'debris', '3': 'lymphocytes', '4': 'mucus', '5': 'smooth muscle', '6': 'normal colon mucosa', '7': 'cancer-assoc
    Number of samples: {'train': 89996, 'val': 10004, 'test': 7180}
    Description: The PathMNIST is based on a prior study for predicting survival from colorectal cancer histology slides, providing a dataset (NCT-CRC-HE-100K) of 100,000 non-c
    License: CC BY 4.0
```

```
# Visualization

train_dataset.montage(length=1)
```

```
# Montage

train_dataset.montage(length=20)
```



```
# Define a simple CNN model

class Net(nn.Module):
    def __init__(self, in_channels, num_classes):
        super(Net, self).__init__()

        self.layer1 = nn.Sequential(
            nn.Conv2d(in_channels, 16, kernel_size=3),
            nn.BatchNorm2d(16),
            nn.ReLU())

        self.layer2 = nn.Sequential(
            nn.Conv2d(16, 16, kernel_size=3),
            nn.BatchNorm2d(16),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2))

        self.layer3 = nn.Sequential(
            nn.Conv2d(16, 64, kernel_size=3),
            nn.BatchNorm2d(64),
            nn.ReLU())

        self.layer4 = nn.Sequential(
            nn.Conv2d(64, 64, kernel_size=3),
            nn.BatchNorm2d(64),
            nn.ReLU())

        self.layer5 = nn.Sequential(
            nn.Conv2d(64, 64, kernel_size=3, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2))

        self.fc = nn.Sequential(
            nn.Linear(64 * 4 * 4, 128),
            nn.ReLU(),
            nn.Linear(128, 128),
            nn.ReLU(),
            nn.Linear(128, num_classes))

    def forward(self, x):
        x = self.layer1(x)
        x = self.layer2(x)
        x = self.layer3(x)
        x = self.layer4(x)
        x = self.layer5(x)
        x = x.view(x.size(0), -1)
        x = self.fc(x)
        return x

model = Net(in_channels=n_channels, num_classes=n_classes)

# Define loss function and optimizer
if task == "multi-label, binary-class":
    criterion = nn.BCEWithLogitsLoss()
else:
    criterion = nn.CrossEntropyLoss()

optimizer = optim.SGD(model.parameters(), lr=lr, momentum=0.9)
```

```
# Train

for epoch in range(NUM_EPOCHS):
    train_correct = 0
    train_total = 0
    test_correct = 0
    test_total = 0

    model.train()
    for inputs, targets in tqdm(train_loader):
        # Forward + Backward + Optimize
        optimizer.zero_grad()
        outputs = model(inputs)

        if task == 'multi-label, binary-class':
            targets = targets.to(torch.float32)
            loss = criterion(outputs, targets)
        else:
            targets = targets.squeeze().long()
            loss = criterion(outputs, targets)

        loss.backward()
        optimizer.step()
```

```
100%|██████████| 704/704 [01:06<00:00, 10.53it/s]
100%|██████████| 704/704 [01:32<00:00,  7.60it/s]
100%|██████████| 704/704 [01:31<00:00,  7.66it/s]
```

```
# Evaluation

def test(split):
    model.eval()
    y_true = torch.tensor([])
    y_score = torch.tensor([])

    data_loader = train_loader_at_eval if split == 'train' else test_loader

    with torch.no_grad():
        for inputs, targets in data_loader:
            outputs = model(inputs)

            if task == 'multi-label, binary-class':
                targets = targets.to(torch.float32)
                outputs = outputs.softmax(dim=-1)
            else:
                targets = targets.squeeze().long()
                outputs = outputs.softmax(dim=-1)
                targets = targets.float().resize_(len(targets), 1)

            y_true = torch.cat((y_true, targets), 0)
            y_score = torch.cat((y_score, outputs), 0)

        y_true = y_true.numpy()
        y_score = y_score.detach().numpy()

        evaluator = Evaluator(data_flag, split)
        metrics = evaluator.evaluate(y_score)

        print('%s  auc: %.3f  acc:%.3f' % (split, *metrics))


print('==> Evaluating ...')
test('train')
test('test')
```

```
==> Evaluating ...
train  auc: 0.967  acc:0.713
test  auc: 0.932  acc:0.556
```

```
data_flag = 'organmnist3d'
download = True

info = INFO[data_flag]
DataClass = getattr(medmnist, info['python_class'])


train_dataset = DataClass(split='train',  download=download)

# Encapsulate data into dataloader form
train_loader = data.DataLoader(dataset=train_dataset, batch_size=BATCH_SIZE, shuffle=True)
```

```
Using downloaded and verified file: /home/three/.medmnist/organmnist3d.npz
```

```
x, y = train_dataset[0]

print(x.shape, y.shape)
```

```
(1, 28, 28, 28) (1,)
```

```
for x, y in train_loader:
    print(x.shape, y.shape)
    break
```

```
torch.Size([128, 1, 28, 28, 28]) torch.Size([128, 1])
```

```
frames = train_dataset.montage(length=1, save_folder="tmp/")
frames[10]
```

```
frames = train_dataset.montage(length=20, save_folder="tmp/")

frames[10]
```