Joint global workshop on Sustainable Development in
Computer-Assisted Interventions and Diagnosis

Deep learning with PyTorch

Medical image classification demo

Amoon Jamzad

May 25, 2022

DL frameworks

	Keras	PyTorch
API	High-Level	Low-Level
Coding	Python	Python
Architecture	Simple	Complex
Speed	Slow	Fast
Data Sets	Small	Large

>hackr.io

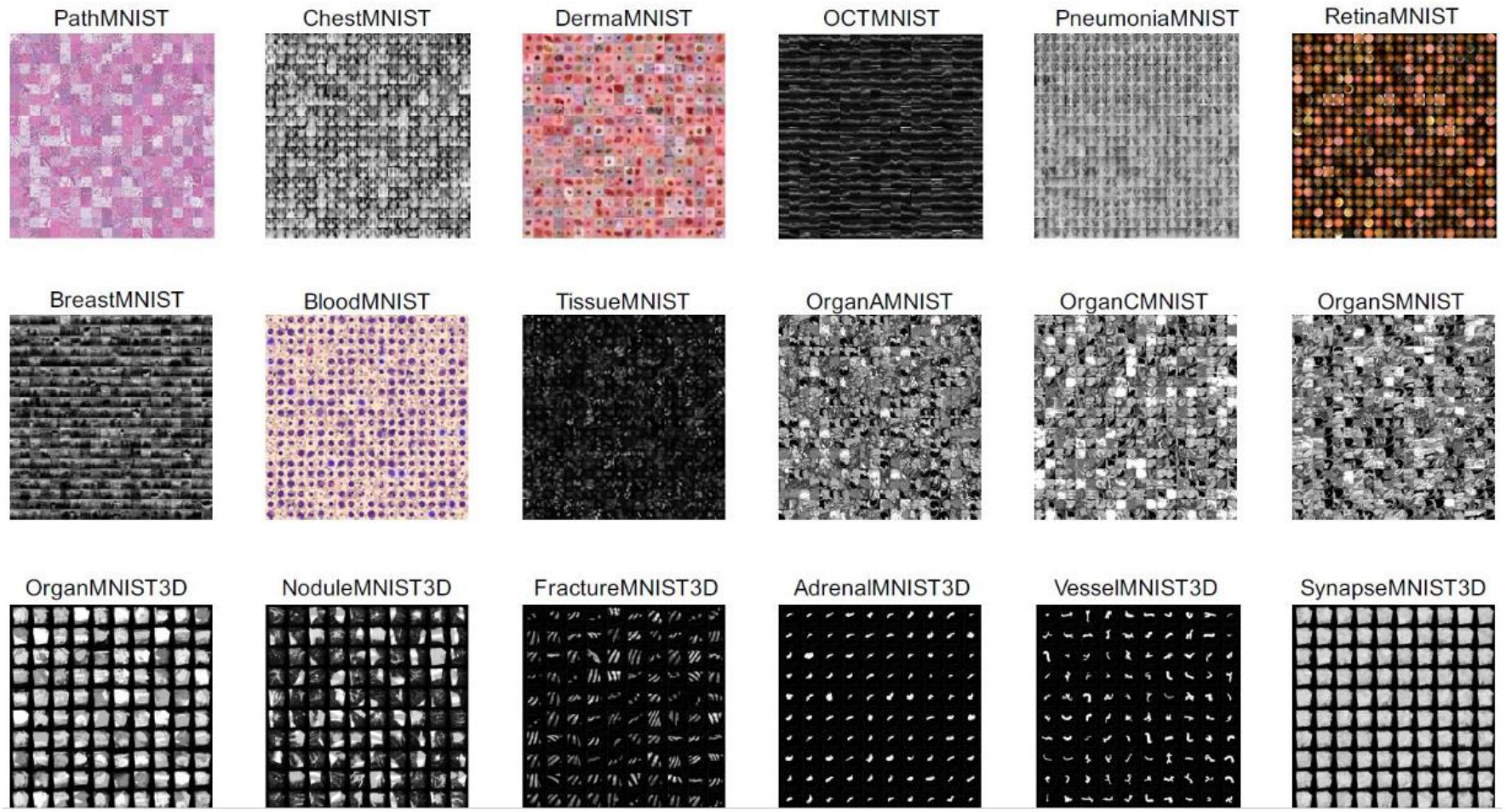
- Suggested reading: https://www.simplilearn.com/keras-vs-tensorflow-vs-pytorch-article

General structure

- Data exploration/preparation
  - Patient-based data stratification
  - Dataset/Dataloader
- Model generation
  - Layers
  - Size calculation
- Training parameter definition
  - Loss/metric/optimizer
- Model training/evaluation
  - Learning curve visualization
  - Early stopping

MedMNIST

https://medmnist.com/



BreastMNIST	Breast Ultrasound	Binary-Class (2)	780 samples	546 / 78 / 156

Custom Dataset class

- __init__
  - Initialization of data, labels, and parameters
- __len__
  - returns the size of the dataset, and
- __getitem__
  - returns a sample from the dataset given an index

- More info: https://towardsdatascience.com/building-efficient-custom-datasets-in-pytorch-2563b946fd9f

CNN receptive field and size

$$n_{out} = \left\lfloor \frac{n_{in} + 2p - k}{s} \right\rfloor + 1$$

n_{in}: number of input features
n_{out}: number of output features
k: convolution kernel size
p: convolution padding size
s: convolution stride size

- More info: https://blog.mlreview.com/a-guide-to-receptive-field-arithmetic-for-convolutional-neural-networks-e0f514068807

Coding tutorial

```
# define 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.ReLU())

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

        self.layer3 = nn.Sequential(
            nn.Conv2d(16, 32, kernel_size=3),
            nn.ReLU())

        self.layer4 = nn.Sequential(
            nn.Conv2d(32, 32, kernel_size=3),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2))

        self.fc = nn.Sequential(
            nn.Linear(32 * 4 * 4, 64),
            nn.ReLU(),
            nn.Linear(64, num_classes))
```