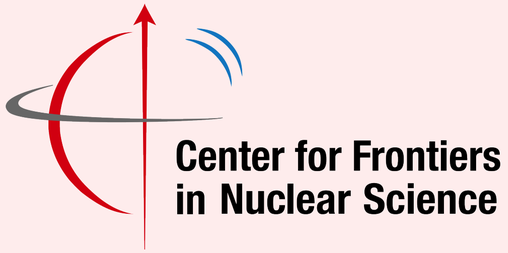
Online Resources for ML/AI Beginners

Win Lin
Stony Brook University

CFNS MLLM second meeting
12/06/2024



Google Colab

- https://colab.research.google.com: sign in with your google account
  - Colab notebook: Jupyter notebook hosted by Colab
  - Upload data to google drive
  - Code runs on Google’s cloud servers
  - Everything is on the drive - i.e. Code can also be shared via google drive
  - Can link to GitHub
- Provide ML courses, from basic terminologies to more advance ML models including neural network

Google Colab

- https://colab.research.google.com: sign in with your google account
  - Colab notebook: Jupyter notebook hosted by Colab
  - Upload data to google drive
  - Code runs on Google's cloud servers
  - Everything is on the drive - i.e. Code can also be shared via google drive
  - Can link to GitHub
- Provide ML courses, from basic terminologies to more advance ML models including neural network

- Limitation for free access:
  - All their computing resources availability are dynamic. It depends on what they have available currently..

    No current/average resources information are shown 😞
  - GPU/TPU (not always available)
  - RAM (generally < 12 GB)
  - Notebook can run at most 12 hours

Alternative notebook/cloud resources

- Paperspace Gradient
  - https://www.paperspace.com
  - Can save up to 5 projects
  - 5 GB storage
  - 1 free GPU (?)

- Kaggle
  - https://www.kaggle.com
  - 20 GB of storage
  - 1 free GPU: NVIDIA Tesla P100
  - 1 free TPU: v3-8
  - Provides a platform for discussion, code sharing, coding competitions etc.
  - Provides courses including intro to Python

kaggle™

TensorFlow

- https://www.tensorflow.org
- Sufficient to uses Kersa API in most cases
- Install and run locally, or run on Google Colab

Example with Kersa on Google Colab:

Load the library

Load a dataset

Build a model

Configure and compile model

Train and evaluate

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
  loss='sparse_categorical_crossentropy',
  metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

PyTorch

- https://pytorch.org
- Install and run locally, can also run with Amazon Web Services, Google Cloud Platform, Microsoft Azure
- Offers C++ API
- Tutorials are also in Chinese, Japanese and Korean (translated by volunteers)

Example:

```
import torch
from torch import nn
from torch.utils.data import DataLoader
from torchvision import datasets
from torchvision.transforms import ToTensor

# Download training data from open datasets.
training_data = datasets.FashionMNIST(
    root="data",
    train=True,
    download=True,
    transform=ToTensor(),
)

# Download test data from open datasets.
test_data = datasets.FashionMNIST(
    root="data",
    train=False,
    download=True,
    transform=ToTensor(),
)

batch_size = 64
```

PyTorch

- https://pytorch.org
- Install and run locally, can also run with Amazon Web Services, Google Cloud Platform, Microsoft Azure
- Offers C++ API
- Tutorials are also in Chinese, Japanese and Korean (translated by volunteers)

Example:

```
# Create data loaders.
train_dataloader = DataLoader(training_data, batch_size=batch_size)
test_dataloader = DataLoader(test_data, batch_size=batch_size)

for X, y in test_dataloader:
    print(f"Shape of X [N, C, H, W]: {X.shape}")
    print(f"Shape of y: {y.shape} {y.dtype}")
    break

# Get cpu, gpu or mps device for training.
device = (
    "cuda"
    if torch.cuda.is_available()
    else "mps"
    if torch.backends.mps.is_available()
    else "cpu"
)
print(f"Using {device} device")
```

PyTorch

- https://pytorch.org
- Install and run locally, can also run with Amazon Web Services, Google Cloud Platform, Microsoft Azure
- Offers C++ API
- Tutorials are also in Chinese, Japanese and Korean (translated by volunteers)

Example:

```
# Define model
class NeuralNetwork(nn.Module):
    def __init__(self):
        super().__init__()
        self.flatten = nn.Flatten()
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(28*28, 512),
            nn.ReLU(),
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 10)
        )

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

model = NeuralNetwork().to(device)
print(model)
```

PyTorch

- https://pytorch.org
- Install and run locally, can also run with Amazon Web Services, Google Cloud Platform, Microsoft Azure
- Offers C++ API
- Tutorials are also in Chinese, Japanese and Korean (translated by volunteers)

Example:

```
# Optimizing the Model Parameters
loss_fn = nn.CrossEntropyLoss()
optimizer = torch.optim.SGD(model.parameters(), lr=1e-3)

def train(dataloader, model, loss_fn, optimizer):
    size = len(dataloader.dataset)
    model.train()
    for batch, (X, y) in enumerate(dataloader):
        X, y = X.to(device), y.to(device)

        # Compute prediction error
        pred = model(X)
        loss = loss_fn(pred, y)

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

        if batch % 100 == 0:
            loss, current = loss.item(), (batch + 1) * len(X)
            print(f"loss: {loss:>7f}  [{current:>5d}/{size:>5d}]")
```

PyTorch

- https://pytorch.org
- Install and run locally, can also run with Amazon Web Services, Google Cloud Platform, Microsoft Azure
- Offers C++ API
- Tutorials are also in Chinese, Japanese and Korean (translated by volunteers)

Example:

```
def test(dataloader, model, loss_fn):
    size = len(dataloader.dataset)
    num_batches = len(dataloader)
    model.eval()
    test_loss, correct = 0, 0
    with torch.no_grad():
        for X, y in dataloader:
            X, y = X.to(device), y.to(device)
            pred = model(X)
            test_loss += loss_fn(pred, y).item()
            correct += (pred.argmax(1) == y).type(torch.float).sum().item()
    test_loss /= num_batches
    correct /= size
    print(f"Test Error: \n Accuracy: {(100*correct):>0.1f}%, Avg loss: {test_loss:>8f} \n")

    epochs = 5
for t in range(epochs):
    print(f"Epoch {t+1}\n-------------------------------")
    train(train_dataloader, model, loss_fn, optimizer)
    test(test_dataloader, model, loss_fn)
print("Done!")

# Save the model
torch.save(model.state_dict(), "model.pth")
print("Saved PyTorch Model State to model.pth")
```

Last slide

Thank you :)