Lecture 12- Introduction to Python Project

Guiliang Liu

The Chinese University of Hong Kong, Shenzhen

CSC-1004: Computational Laboratory Using Java
Course Page: [Click]

Outline

- Python Project Management and Basic Knowledge
- Basic Machine Learning with Python

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Main Components in a Machine Learning Project

- Environment Manager (e.g., MiniConda): manage the machine learning environment.

- Machine Learning Library (e.g., Pytorch, Scikit-Learn): provide API for machine learning algorithms.

- Programming language (e.g., Python): implement the logic for handling the input/output.

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step I: Read the configuration

The main function is the entrance of a Python program.

```
if __name__ == '__main__':
    arg = read_args()

    """toad training settings"""
    config = load_config(arg)

    """train model and record results"""
    run(config)

    """plot the mean results"""
    plot_mean()
```

- Run from the command line.

  `python main.py ./config/minist.yaml`

- Read the input argument.

  `arg=read_args()`

- Load the config and run training.

  `config=load_config()`

  `run(config)`

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step I: Read the configuration

The main function is the entrance of a Python program.

The "minist.yaml" file.

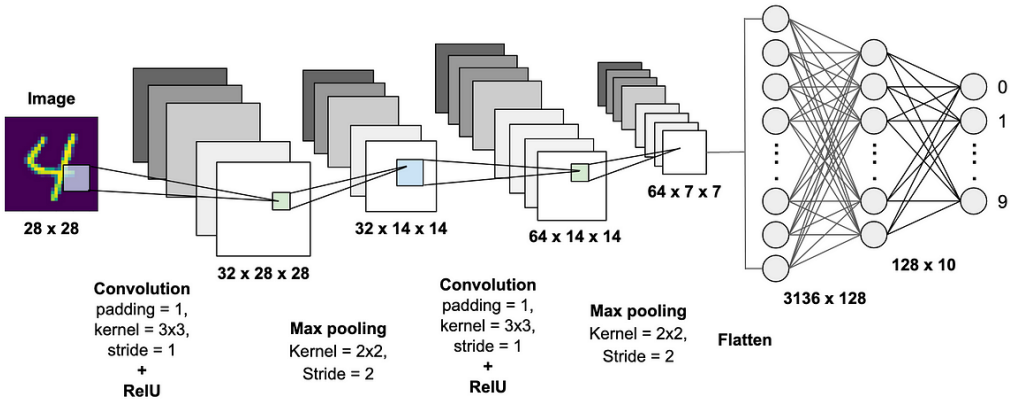
```yaml
batch_size: 64  #  input batch size for training
test_batch_size: 1000  # input batch size for testing
epochs: 15  # number of epochs to train
lr: 0.01  # learning rate
gamma: 0.7  # learning rate step gamma
no_cuda: True  # disables CUDA training
no_mps: True   # disables macOS GPU training
dry_run: False # quickly check a single pass
seed: 123  # random seeds for the three runs are 123, 321, 666
log_interval: 10  # how many batches to wait before logging training status
save_model: True  # For Saving the current Model
```

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step II: Define the Neural Network Model

Define the neural network structure with Pytorch.

Note that this is the complete structure. We use a simplified one in our code.



Step II: Define the Neural Network Model

Define the neural network structure with Pytorch.

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 32, 3, 1)
        self.conv2 = nn.Conv2d(32, 64, 3, 1)
        self.dropout1 = nn.Dropout(0.25)
        self.dropout2 = nn.Dropout(0.5)
        self.fc1 = nn.Linear(9216, 128)
        self.fc2 = nn.Linear(128, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = F.relu(x)
        x = self.conv2(x)
        x = F.relu(x)
        x = F.max_pool2d(x, 2)
```

- Define the network layers.

  def __init__(self):

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step II: Define the Neural Network Model

Define the neural network structure with Pytorch.

```
def forward(self, x):
    x = self.conv1(x)
    x = F.relu(x)
    x = self.conv2(x)
    x = F.relu(x)
    x = F.max_pool2d(x, 2)
    x = self.dropout1(x)
    x = torch.flatten(x, 1)
    x = self.fc1(x)
    x = F.relu(x)
    x = self.dropout2(x)
    x = self.fc2(x)
    output = F.log_softmax(x, dim=1)
    return output
```

- Connect these layers to map inputs (i.e., x) to outputs.

  `def forward(self, x):`

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step III: Run the code

The run() function – Part I.

```
def run(config):
    use_cuda = not config.no_cuda and torch.cuda.is_available()
    use_mps = not config.no_mps and torch.backends.mps.is_available()

    torch.manual_seed(config.seed)

    if use_cuda:
        device = torch.device("cuda")
    elif use_mps:
        device = torch.device("mps")
    else:
        device = torch.device("cpu")

    train_kwargs = {'batch_size': config.batch_size, 'shuffle': True}
    test_kwargs = {'batch_size': config.test_batch_size, 'shuffle': True}
    if use_cuda:
        cuda_kwargs = {'num_workers': 1,
                       'pin_memory': True,}
        train_kwargs.update(cuda_kwargs)
        test_kwargs.update(cuda_kwargs)
```

- Set up CUDA. CUDA is a parallel computing platform and application programming interface (API). Ignore it if you don't need to use GPU.

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step III: Write the Run() function

The run() function – Part II.

```
# download data
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize((0.1307,), (0.3081,))
])
dataset1 = datasets.MNIST('./data', train=True, download=True, transform=transform)
dataset2 = datasets.MNIST('./data', train=False, transform=transform)

"""add random seed to the DataLoader, pls modify this function"""
train_loader = torch.utils.data.DataLoader(dataset1, **train_kwargs)
test_loader = torch.utils.data.DataLoader(dataset2, **test_kwargs)
```

- Download training and testing data from the web.
- Store them in data loader for training and testing.

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step III: Write the Run() function

The run() function – Part III.

```
model = Net().to(device)
optimizer = optim.Adadelta(model.parameters(), lr=config.lr)

"""record the performance"""
epoches = []
training_accuracies = []
training_loss = []
testing_accuracies = []
testing_loss = []
```

- Initialize the neural network model.

  model=Net().to(device)

- Initialize the optimizer.

  optimizer=optim.Adadelta(...

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step III: Write the Run() function

The run() function – Part IV.

```
scheduler = StepLR(optimizer, step_size=1, gamma=config.gamma)
for epoch in range(1, config.epochs + 1):
    train_acc, train_loss = train(config, model, device, train_loader, optimizer, epoch)
    """record training info, Fill your code"""
    test_acc, test_loss = test(model, device, test_loader)
    """record testing info, Fill your code"""
    scheduler.step()
    """update the records, Fill your code"""
```

- Train the model in each iteration.
- Test the model in each iteration.
- Go over the dataset "config epochs" times.

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step IV: Write the train() function

The train() function.

```
model.train()
for batch_idx, (data, target) in enumerate(train_loader):
    data, target = data.to(device), target.to(device)
    optimizer.zero_grad()
    output = model(data)
    loss = F.nll_loss(output, target)
    loss.backward()
    optimizer.step()
'''Fill your code'''
training_acc, training_loss = None, None  # replace this line
return training_acc, training_loss
```

- Set the training mode.
  `model.train()`
- Obtain output from input (e.g., data, the images).
  `output=model(data)`
- Calculate Loss due to the estimation error.
  `loss=F.nll_loss(output, target)`

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step IV: Write the train() function

The train() function.

```
model.train()
for batch_idx, (data, target) in enumerate(train_loader):
    data, target = data.to(device), target.to(device)
    optimizer.zero_grad()
    output = model(data)
    loss = F.nll_loss(output, target)
    loss.backward()
    optimizer.step()
'''Fill your code'''
training_acc, training_loss = None, None  # replace this line
return training_acc, training_loss
```

- Calculate the gradient.
  `loss.backward()`
- Perform the gradient descent for updating the neural network.
  `optimizer.step()`

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Step V: Write the test() function

The test() function.

```
model.eval()
test_loss = 0
correct = 0
with torch.no_grad():
    for data, target in test_loader:
        '''Fill your code'''
        pass
testing_acc, testing_loss = None, None  # replace this line
return testing_acc, testing_loss
```

- Set the testing mode.

  `model.eval()`

- Stop gradient descent.

  `with torch.no_grad()`

香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen

Question and Answering (Q&A)