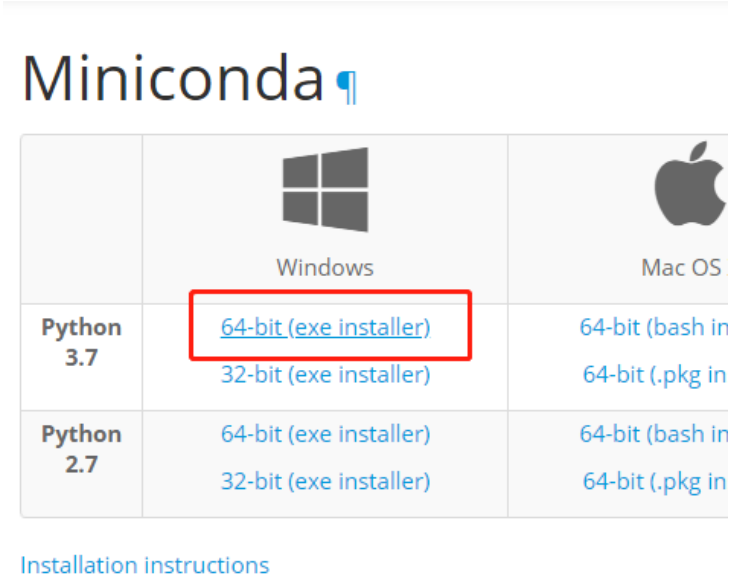
Deep Learning and Computer Vision

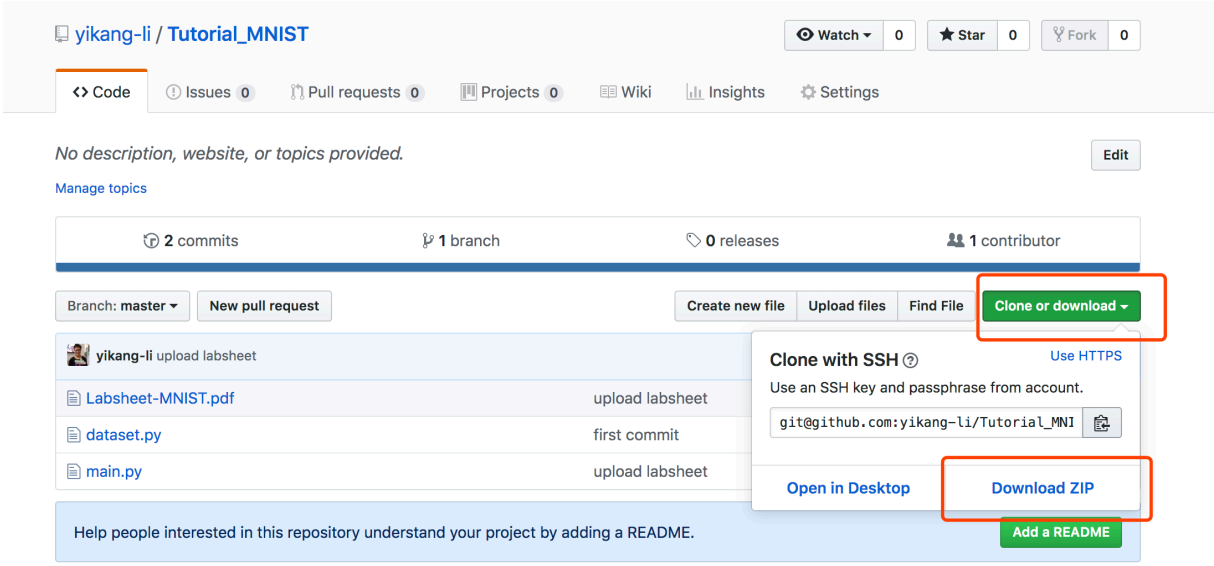
Task 1 Prepare the Environment

1. Download the Miniconda installer: https://conda.io/miniconda.html
2. Select **Python 3.7 64-bit (exe installer)** to download.



3. Click the exe and you will the following installation page:

4. Click yes or choose the default recommendations.



5. During the installation, download the workspace file: https://github.com/yikang-li/Tutorial_MNIST
6. Unzip the file to the directory you can find: \path\to\workspace\
7. After the installation, press "windows" button, search "Anaconda Prompt" and open it.
8. Install the PyTorch and Torchvision Package:

```
conda install -c pytorch pytorch torchvision
```

9. Jump to the project directory:

```
cd C:\Users\t-yikl\workspace\Tutorial-MNIST
```

Task 2 Prepare the dataset

We have collected some MINIST images for you: hand-written digits from 0 to 9.

- Download the data folder from link:
  https://tinyurl.com/y4eh745u
- Unzip the file and place it under your workspace. Images are like this:

Figure 1: examples of MNIST dataset.

Task 3 Finish your DataLoader

How to load the data in the expected way is very important in implementing your own deep learning algorithm. In this section, you should try to write your own dataset "myMNISTDataset":

1) Get the dataset instance by providing the path to the dataset folder;
2) get the categories of characters from the folder;
3) read the image from the folder;
4) get the list of the images.
5) (optional) trainform is used to transform the retrieved data to the expected format

.

```
    ####################################################
    # implement your own myMNISTDataset at ./dataset.py #
    ####################################################
    train_loader = torch.utils.data.DataLoader(
                    myMNISTDataset('data/train',
                        transform=transforms.Compose([
                            transforms.ToTensor(),
                            transforms.Normalize((0.1307,), (0.3081,))
                        ])),
        batch_size=args.batch_size, shuffle=True, **kwargs)
    test_loader = torch.utils.data.DataLoader(
                    myMNISTDataset('data/test',
                        transform=transforms.Compose([
                            transforms.ToTensor(),
                            transforms.Normalize((0.1307,), (0.3081,))
                        ])),
        batch_size=args.test_batch_size, shuffle=False, **kwargs)
```

"myMNISTDataset" is defined at "./dataset.py"

```
from torch.utils.data import Dataset

class myMNISTDataset(Dataset):
    """Face Landmarks dataset."""

    def __init__(self, root_dir, transform=None):
        """
        Args:
            root_dir (string): Directory with all the folders of images.
            transform (callable, optional): Optional transform to be applied
                on a sample.
        """
        self.root_dir = root_dir
        self.transform = transform

    def __len__(self):
        return num_images

    def __getitem__(self, idx):
        return (data, target)
```

You need to implement three functions:

- __init__: initialization function for MNIST dataset.
    a. Provide the root_dir of the train/test set.
    b. Get the categories of the dataset
    c. Get the image list of the dataset as well their paths

- __len__: get the number of the images in the dataset
- __getitem__: get an (image, target) tuple
  a. image: a PIL image object: https://pillow.readthedocs.io/en/3.1.x/reference/Image.html
  b. target: a torch.LongTensor of the digit corresponding to the image

Hint: document is a very useful tool during coding. How to use document is a necessary skill for implementing your Deep Learning ideas. So feel free to seek for help from: https://pytorch.org/docs/stable/index.html

Task 4 Model Training

Have a try of training the model with your own dataset:

python main.py

```
Train Epoch: 1 [53120/60000 (88%)]      Loss: 0.263932
Train Epoch: 1 [53760/60000 (90%)]      Loss: 0.092141
Train Epoch: 1 [54400/60000 (91%)]      Loss: 0.128506
Train Epoch: 1 [55040/60000 (92%)]      Loss: 0.189118
Train Epoch: 1 [55680/60000 (93%)]      Loss: 0.033703
Train Epoch: 1 [56320/60000 (94%)]      Loss: 0.035877
Train Epoch: 1 [56960/60000 (95%)]      Loss: 0.077423
Train Epoch: 1 [57600/60000 (96%)]      Loss: 0.117247
Train Epoch: 1 [58240/60000 (97%)]      Loss: 0.192404
Train Epoch: 1 [58880/60000 (98%)]      Loss: 0.206398
Train Epoch: 1 [59520/60000 (99%)]      Loss: 0.063671

Test set: Average loss: 0.1012, Accuracy: 9669/10000 (97%)

Train Epoch: 2 [0/60000 (0%)]   Loss: 0.145093
Train Epoch: 2 [640/60000 (1%)] Loss: 0.119384
Train Epoch: 2 [1280/60000 (2%)]        Loss: 0.101953
Train Epoch: 2 [1920/60000 (3%)]        Loss: 0.069106
Train Epoch: 2 [2560/60000 (4%)]        Loss: 0.105178
Train Epoch: 2 [3200/60000 (5%)]        Loss: 0.115073
```

Also, you can specify the arguments. Type

python main.py –help

to check the argument list.

```
Yikangs-Macbook-Pro:tutorial_MNIST yikang$ python main.py --help
usage: main.py [-h] [--batch-size N] [--test-batch-size N] [--epochs N]
               [--lr LR] [--momentum M] [--no-cuda] [--seed S]
               [--log-interval N] [--save-model]

PyTorch MNIST Example

optional arguments:
  -h, --help            show this help message and exit
  --batch-size N        input batch size for training (default: 64)
  --test-batch-size N   input batch size for testing (default: 1000)
  --epochs N            number of epochs to train (default: 10)
  --lr LR               learning rate (default: 0.01)
  --momentum M          SGD momentum (default: 0.5)
  --no-cuda             disables CUDA training
  --seed S              random seed (default: 1)
  --log-interval N      how many batches to wait before logging training status
  --save-model          For Saving the current Model
```

Try to modify the arguments and check the performance of the model, e.g. modifying the learning rate:

python main.py --lr 0.001

Task 5 Submit your dataset.py

Submit your dataset.py to tecs2461assignments@gmail.com with the tile: [TECS 2461] Assignment2-GivenName-Surname