CS 6364 Homework 7

October 22, 2019

Deadline for the first submission: **Oct-24-2019**.
All assignments **MUST** have your name, student ID, course name/number at the beginning of your documents.
Your homework **MUST** be submitted via Blackboard with file format and name convention as follows:

HW#_Name_writeup.**pdf** (for writing part)
HW#_Name_code.**zip** (for coding part)

If you have any questions, please contact me.

For the following questions, if you need a GPU to run, Google provide a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. Here is the link: https://colab.research.google.com/notebooks/welcome.ipynb?hl=enscrollTo=5fCEDCU_qrC0

Q1 Given an input image (see the attachment), design two filters/kernels with size of 3 by 3 to detect all horizontal and vertical edges. To make it easier, you can first convert the image into gray scale using OpenCV. Your architecture only contains 1 convolutional layer, 1 pooling layer and 1 fully connected layer. Show the visualisations of ~~these three layers~~ the output of the convolutional layer, the pooled layer and the ReLU activated convolutional layer (not the fully connected layer) for both kernels.

What need to submit:

- Python programming code
- Outpus six figures (two kernels by three layers) in total.

Q2 Build an image classification model using Convolutional Neural Networks (CNN) in PyTorch. The data set (i.e. Fashion-MNIST) can be found here:

https://github.com/zalandoresearch/fashion-mnist

It consists of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes.

By building the classification model, I suggest you divide it into several subtasks:

1. Loading the data set.
   - Please explore a few samples and visualize these images.
2. Creating a validation set and preprocessing the images.
   - Split the training data set to a training set (90% samples) and a validation set (10% samples)
   - convert the images and the targets into torch format for both training and validation data sets.
3. Implementing CNNs using PyTorch.
   - Define the architecture with just 2 convolutional layers to extract features from the images and then use a fully connected dense layer to classify those features into their respective categories (i.e. two Conv2d layers and a linear layer).
   - Train the model for 25 epochs and show the validation losses by printing in console. You are expected to see that the validation loss is decreasing as epoch increases.
   - Visualize the training and validation losses by plotting them.
   - Show the accuracy of the model on the training and validation set.
4. Generating predictions for the test set.

```
Net(
  (cnn_layers): Sequential(
    (0): Conv2d(1, 4, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): BatchNorm2d(4, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace)
    (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (4): Conv2d(4, 4, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (5): BatchNorm2d(4, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (6): ReLU(inplace)
    (7): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (linear_layers): Sequential(
    (0): Linear(in_features=196, out_features=10, bias=True)
  )
)
```

- Load the test images.
- Do the pre-processing steps on these images similar to what you did for the training images.
- Generate predictions for the test set.

What to submit:

- Python programming code
- Five image visualization when loading the data set.
- Plotting visualization of the training and validation losses.
- Report the accuracy of your model on training and validation set.
- Your predictions for the test images.

Q3 Following the step by step instructions below to build a LeNet-5 CNN architecture in Pytorch using Fashion-MNIST data set provided in Q2.

https://engmrk.com/lenet-5-a-classic-cnn-architecture/

What to submit:

- Python programming code
- Visualization of plotting the training accuracy and loss after each epoch
- Your predictions for the test images.