Assignment 5 - Neural Networks

Full Marks : 120

Question 1: (80)

Task: The aim of this assignment is to train and test Convolutional Neural Networks for image classification on CIFAR10 dataset using PyTorch Module and to acquaint yourself with wandb which is a cool tool for monitoring large experiments.

Data:

- train_images: Consist of 50000 images of 32 x 32 RGB images.
- train_labels: Consist of 50000 labels from 10 classes for the images in train_images. The labels are described below.
- test_images: Consist of 10000 images of 32 x 32 RGB images.
- test_labels: Consist of 10000 labels from 10 classes for the images in test_images.

Labels:

Each training and test image is classified into **ANY ONE** of the following labels:

0 - Airplane	1 - Automobile
2 - Bird	3 - Cat
4 - Deer	5 - Dog
6 - Frog	7 - Horse
8 - Ship	9 - Truck

Implementation Details:

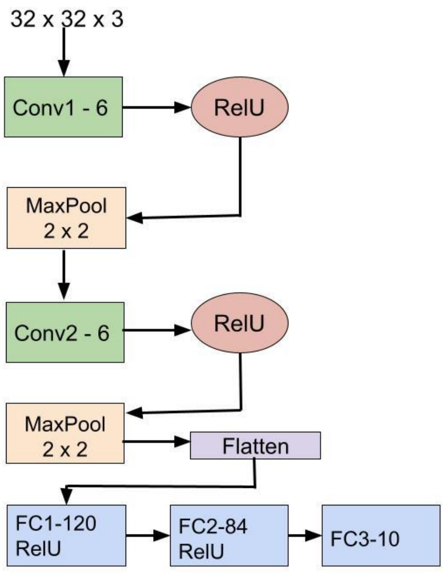
Supporting files are available here: https://github.com/SoumiDas/CS60020_S2021

1. **Data Load:** Use the file **main.py** given in the above link to load the data and carry on further experiments.

2. **Model Creation**: Use the following structure to train your Convolutional Neural Network. The notations are given below: **(15)**

   Convx-N : Here N is the size of the output channels.

FCx-N : Here N is the size of the output layers.
Maintain a **kernel size of 5 x 5** for the convolutional layers.



Use the file **model.py** to build the framework.

3. **Training Module**: Implement a mini-batch SGD using **main.py** to train the CNN. Use the following configurations while training: **(40)**

- Use SGD optimizer with learning rate = 0.001, momentum = 0.9 and cross-entropy as the loss function.
- Use Adam optimizer with learning rate = 0.01 and cross-entropy as the loss function.
- Use SGD optimizer with learning rate = 0.001, momentum = 0.9 and squared error loss as the loss function.
- Use Adam optimizer with learning rate = 0.01 and squared error loss as the loss function.

You can use early stopping too if loss converges beforehand.

For each of the configurations above, retain/save the best model (yielding best test set accuracy while testing at each epoch).
Use the best saved models to report the final test set accuracies for the four configurations for Q2 under Submission Details below.

4. **Compare with ResNet34**: Extract the pre-trained features (of dimension 512) from the penultimate layer from ResNet34 module and train a 3 layer neural network of the following structure: **(10)**

Layer 1: Input features - 512, Output features - 256

Layer 2: Input features - 256, Output features - 128
Layer 3: Input features - 128, Output - 10

Use SGD optimizer, learning rate = 0.01 , momentum = 0.9 and cross-entropy as the loss function and train till training loss converges.

Submission Details:

You should submit the following in zipped format (Question1.zip):

- **Report** as asked below (Report.pdf) using wandb (details below) with all the contents as mentioned under the '**Submission Details**'. Analyse the observations and explain them. **(15)**
- **Python codes**
    - main.py - you can add functions as per requirements.
    - model.py

Your report should contain the following details:

1. Parameters of the above described Convolutional Neural Network (in diagram) in the following way:

Layer	Input Size	Output Size	# of Parameters

2. For all the four configurations, plot a single visualization diagram of Confusion Matrix for all the 10 classes in the test set and indicate which class got classified the best and the worst among all.

3. **Wandb:** Log all the results (training loss vs epochs, test loss vs epochs, test set accuracy vs epochs, sample prediction samples) using four configurations in the wandb dashboard and create a report of the same.
   Also plot the (training loss vs epochs, test loss vs epochs, test set accuracy vs epochs) for the ResNet34 architecture in the same report and explain your observations.

Is the pre-trained model features helping with better prediction accuracies than the best configuration out of the four defined above? State yes or no, with reasons.

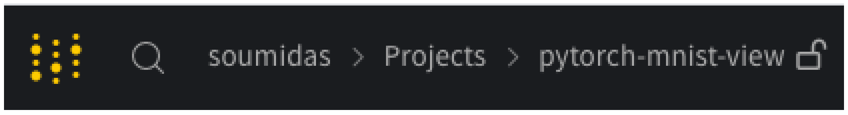
Additional information

--

A tutorial of training CNN on MNIST and using Wandb to log results can be found in this link:
https://colab.research.google.com/drive/1XBC88XMqMFpl7AenpHXPnFhhAQN8pYiQ?usp=sharing

a. A sample report generated from wandb artifacts can be found in https://www.dropbox.com/s/nofyinh8k5ze30a/MNIST%20using%20CNN%20in%20Pytorch%20_%20pytorch-mnist-view%20%E2%80%93%20Weights%20%26%20Biases.pdf?dl=0 .
b. The link for the report can be also viewed in https://wandb.ai/soumidas/pytorch-mnist-view/reports/MNIST-using-CNN-in-Pytorch--Vmlldzo2Njg0Mjg .
c. Some guide on creating nice reports can be found in this link https://docs.wandb.ai/guides/reports

Please make sure to make the project "public" under which you're saving the artifacts. A sample image of what you can see on your wandb profile when you view your project in public mode:



Project name for this case is **pytorch-mnist-view** . Please note the lock sign beside it which indicates it's public. Please make the wandb report organised and explain your observations.

==

Question 2: (40)

1. Suppose we have a fully connected network with three inputs x_1, x_2, x_3 . The network has two hidden layers with three neurons each and sigmoid activation functions. It has an output layer which is a softmax layer. Considering all weights as equal to 1 and all biases to be equal to 0, what will be the output y of the network in terms of the function of **x** = $[x_1, x_2, x_3]$? Write the network architecture and show the calculation.
2. Show that for a binary classification problem, the minimisation of cross-entropy is equivalent to minimizing KL divergence between true and predicted distributions.
3. Draw a neural network with only one hidden layer (number of neurons depends on the expansion) which implements the following expression:

$$y = (P \lor Q) \oplus (\neg R \lor \neg S)$$

where v = Logical OR, $\land$ = Logical AND, $\neg$ = Logical NOT, $\oplus$ = XOR . Show the weights from each hidden unit and the resultant value in the output unit. Expand the expression above which leads you to drawing the neural network.

Submission Instructions: Please submit a pdf with all the answers to the above three questions, written preferably in Latex. (Filename: Question2.pdf)