Lab Course: Distributed Data Analytics
Exercise Sheet 6

Prof. Dr. Dr. Schmidt-Thieme, Daniela Thyssens
Information Systems and Machine Learning Lab
University of Hildesheim
Submission deadline: Friday June 17, 23:59PM (on LearnWeb, course code: 3116)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit two items a) a zipped file containing python scripts and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission needs to be made before the deadline, only through learnweb.
4. Unless explicitly stated, you are not allowed to use scikit, sklearn or any other library for solving any part. All implementations must be done by yourself.

1 Exercise 1: PyTorch Network Analysis (10 points)

For this first task, you are required to implement LeNet (see Annex) for the following datasets for image classification:

- MNIST
- CIFAR10

Both datasets are available in the PyTorch Datasets library and you may use the library to service the model. **Note**: LeNet was originally written for MNIST dataset which has 1-Channel black and white images, so you would need to keep that in mind when you want to implement it for CIFAR10, which is a 3-Channel RGB image dataset. Furthermore, the image size for the two datasets is different, that is something to keep in mind as well when adapting LeNet for CIFAR10.

Some configurations for you to try when training MNIST and CIFAR are as follows:

- Learning rate [0.1, 0.01, 0.001]
- Optimizer [SGD, ADAM, RMSPROP]

Note that you do not necessarily need to provide results for all combinations of these configurations. A valid approach would be to fix the choice of the Optimizer and try different learning rates for that Optimizer only.

One thing that is essential to learn when training network models is using **Tensorboard with PyTorch**. Tensorboard enables us to visualize the network performance in an effective manner. Please report the Train/Test **accuracy** and Train/Test **loss** using Tensorboard. Refer to the Annex section for useful tips. In addition to the Train/Test metrics, please also visualize the activation maps in the Tensorboard. Activation maps enable us to see what the network is seeing as our inputs traverse the network and are indicative of the receptive field of our CONV layers. Vary the kernel size in the CONV layers and show the difference it creates in our activation maps.
NOTE: When you modify the kernel size, it impacts the output size of the CONV layer and the input/output sizes of the layers following the change in kernel size would need to be adapted as well.
Try 2 different kernel sizes and report the impact they have on the activation maps.

2 Exercise 2: Custom Task (10 points)

In Exercise 1 we implemented a well-known network for image classification. Building upon that learning, we will now extend the network to an image regression setting. Using the MNIST dataset, we will design a network that will consume (N, K, C, W, H) batches of images where N, K, C, W, H are batch size, number of images to sum, channels, width and height respectively. The output of the network will be the sum of the number in the K many images. For instance, if we have $K = 3$ and the three images contain the digits 1, 5 and 3, then the output of the network should be 9.
To setup this network, you need to write a custom dataset that will extend the basic dataset module to return a batch-shaped N, K, C, W, H data format instead of the typical N, C, W, H data shape. You should choose K many random images to populate the batch instances. You will need to adapt you network to consume this extended dataset dimension. Use the $x = x.\text{view}(N * K, C, W, H)$ trick discussed in the lab session. Your ground truth target during training will be the sum of the K many images. This can be calculated by summing the integer labels already associated with the images. The design for the network output (predictions) is left up to you, the only requirement here is that it should be a numeric value. Keeping in mind that we are no linger looking at a classification problem but rather a regression problem, our loss should also be modified to reflect that. **HINT**: Regular Maintenance Saves Energy.
For this task, you are to report:

- Train/Test Error on Tensorboard
- 10 sets of test images (each set containing K many images) and the output from your trained network (final value that your model outputs)

Please display the output from your trained network only **after** your model is done training, having a visual output will help you get in the habit of visually inspecting your network behavior.

Annex

1. LeNet `http://yann.lecun.com/exdb/lenet/`
2. LeNet Original paper: LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.
3. PyTorch Installation `https://pytorch.org/get-started/locally/`
4. Conda Install `https://anaconda.org/pytorch/pytorch-cpu`
5. PyTorch datasets `https://pytorch.org/vision/stable/datasets.html`
6. PyTorch dataloaders `https://pytorch.org/tutorials/beginner/basics/data_tutorial.html`
7. PyTorch Custom dataloaders `https://pytorch.org/tutorials/beginner/basics/data_tutorial.html#creating-a-custom-dataset-for-your-files`
8. Tensorboard `https://pytorch.org/docs/stable/tensorboard.html`