Project Report

Domain adaption on images (MNIST, SVHN)

General

Author: Ruslan Sabirov
Group: B17-DS-02
Task: https://hackmd.io/@ImadBek/Hyp3Fs2_r

Seed

I fixed seed of all used modules (random, np.random, torch) to 5 and set some variable to make it reproducible on CUDA.

Problems

I have faced these problems:

1) Not enough RAM to store all the data (only 8 Gb). Solved by using Google Collab and then converting it to py-file for submission.

2) Achieved very good results of MNIST-test accuracy (>80) but forgot to fix seed of torch. Googled how to make PyTorch model training reproducible.

3) Could not found right values for learning rate for different optimizers. Did hyperparameters tuning

4) Created dataset by torchvision.datasets.SVHN() and tried to print shape of one particukar image. It was (3, 28, 28). I was confused why there are still 3 channels even after applying grayscale normalization. Found out that transformations are lazy and done later, right before usage.

1-3. Accuracies and losses

	`SVHN-train set`	`SVHN-test set`	`MNIST-test set`
Accuracy (%)	93%	89%	63%
Average loss	0.2733	0.3784	2.3350

4. Explain your baseline model, how many layers?

In total there are 4 layers: 2 convolution and 2 linear. Model is just optimized CNN model from labs notebooks. Kernel size for convolution layers and number of in/out features of linear layers were increased.

5. Try different techniques and give the three accuracies mentioned above with and without them.

(Batch-norm, different activations, different Optimizers, dropout, etc)

	SVHN-train set	SVHN-test set	MNIST-test set
No dropout, No batchnorm	93 %	89 %	63 %
No dropout, Yes batchnorm	94 %	91 %	60 %
Yes dropout, No batchnorm	92 %	89 %	60 %
Yes dropout, Yes batchnorm	93 %	91 %	58 %

6. Discuss the results with regards to Domain Adaptation.

Domain adaptation is actually a really huge problem because it is really common situation that train and test datasets differ much. That happens because there were made wrong assumptions about data. For example, you are automatically checking if uploaded avatar contains a human face or not. You have trained the model on images with a white background and supposed that users also will do the same. But that did not happen, there were actually no pictures with a white background.

For this dataset, I have assumed that there is exactly one digit on each image, but that is not true about svhn: there are some parts of the neighbor digits on the images.
Even though the model solves the problem of DA well. 60-63% is quite good with such few preprocessing (only grayscale, resize and normalization). It is much better than random classifier (10% accuracy)

For the second submission more complicated models and preprocessing techniques could be applied.