Exercise 5 - **Assignment 2**

Deep Learning Lab
Due: Thursday 18 November 2021, 10:00 pm (time in Lugano)

October 19, 2021

1 Image Classification Using ConvNets [90/100 points]

The CIFAR-10 dataset contains 50 000 training images and 10 000 test images. Each image is of size 32×32 with three channels for red, green, and blue. Each image belongs to one of ten classes: plane, car, bird, cat, deer, dog, frog, horse, ship, or truck. Your task is to implement a convolutional neural network to correctly classify the images.

1.1 Dataset

Like the MNIST dataset we used in the exercise 4, the CIFAR-10 dataset is already available in PyTorch via `torchvision.datasets.CIFAR10`. The respective API documentation can be found here.

1. (4 pt) Load the corresponding dataset. Check the number of images. You can plot images using `matplotlib.pyplot.imshow`. Visualize a few images from the training set. Be careful with the shape expected by `imshow`: the color channel needs to be the last dimension.

2. (15 pt) While `torchvision.datasets.CIFAR10` provides images with values taking between 0 and 1 by default, these values are not standardized to have zero mean and unit variance. Instead of feeding these images as is to the model, we can first normalize these values for each channel. The `transform` argument of `torchvision.datasets.CIFAR10` allows to apply multiple transformations to the original image data. For that, multiple transformations can be first composed via `torchvision.transforms.Compose`, and the result can be given as an argument to `transform`. Use `transforms.Normalize` to normalize the data to have mean 0 and std 1 (carefully check its documentation and provide the correct arguments).

3. (5 pt) The original dataset does not contain a validation set; we thus need to create one. We'll follow to common procedure: Split the original training set into a validation set (the last 1 000 images) and an effective training set (the first 49 000 images). After you have created your dataset object, your dataloader (`torch.utils.data.DataLoader`) can be given a different sampler using the `sampler` argument. For our case, the SubsetRandomSampler is useful because it will pick samples from a given list of indices. Create two dataloaders for the training and validation set by using the same original dataset but with random samplers using different subsets of the data.

1.2 Model

1. (10 pt) Implement the following convolutional neural network using rectified linear activation functions (ReLUs):

   (a) Convolutional layer 1: 32 filters, 3×3.

   (b) Convolutional layer 2: 32 filters, 3×3.

   (c) Max-pooling layer 1: 2×2 windows.

(d) Convolutional layer 3: 64 filters, 3×3.
(e) Convolutional layer 4: 64 filters, 3×3.
(f) Max-pooling layer 2: 2×2 windows.
(g) Fully connected layer 1: 512 units.
(h) Softmax output layer.

1.3 Training

1. (20 pt) Implement the training pipeline. Make sure that you can monitor the current training loss and accuracy every n steps (with a reasonable choice of n), as well as, validation accuracy after each epoch. You should also keep track of the best validation accuracy and the corresponding epoch.
2. (1 pt) Use the following *hyperparameters* to train your model:
   (a) SGD optimizer with learning rate 10^{-3} and momentum 0.9.
   (b) A batch size of 32.
   (c) Train for a total of 20 epochs.
   (d) Use the cross entropy loss.
3. (5 pt) Train your model. The final validation accuracy should be close to 70%.
4. (8 pt) Plot the evolution of the loss and accuracy for your training and validation set. Give a brief analysis and explanation for the discrepancies and similiarties between the training and validation metrics throughout training.
5. (10 pt) There are many ways to improve image classification results for a given neural architecture. Besides the search for better hyperparameters, this might include model regularization and data augmentation. For the simple architecture described above, dropout is an appropriate way to regularise. Study how dropout is implemented in PyTorch, and add a dropout layer after each max-pooling and fully-connected layer. As before, report the evolution of the models performance and compare to the previous run. Note, dropout might slow down convergence; increase the number of epochs if necessary.
6. (10 pt) Experiment with other hyperparameter settings and diligently document your results. Your best model should have a validation accuracy over 75%.
7. (2 pt) Once you have found your best model, compute the test set accuracy and document the result.

1.4 Questions [10/100 points]

Provide a **brief** answer (ideally two, max. three sentences for each item) to the following questions **in your own words** and add them to your report. You can find more information in the relevant subsections of chapter 6,8,9, of the Deep Learning Book.

1. (2 pt) What is the softmax layer and why is it the last layer of the network?
2. (2 pt) What does the momentum parameter of SGD do? Why is it a good idea to set that hyper-parameter to a non-zero value?
3. (2 pt) What is the difference between a convolutional and a fully-connected layer in terms of how parameters are used? Explain why it is a good idea for processing images.
4. (2 pt) Here we used 2D convolution for images. Name two types of data for which you can use 1D convolution.
5. (2 pt) What does dropout do?

Submission

You should deliver the following by the deadline stipulated on iCorsi3:

- **Report**: a single *pdf* file that clearly and concisely provides evidence that you have accomplished each of the tasks listed above. The report should not contain source code (not even snippets). Instead, if absolutely necessary, briefly mention which functions were used to accomplish a task.
- **Source code**: a single Python script that could be easily adapted to accomplish each of the tasks listed above. The source code will be read superficially and checked for plagiarism. Therefore, if a task is accomplished but not documented in the report, it will be considered missing. Note: Jupyter notebook files are not acceptable.

Please carefully read the instructions above to prepare your submission. Failure to stick to these rules may result in reduction of points.