Neural Networks for Images 2023 - Exercise #1

submission date: 30/3/2023

Programming Task: Classification using convolutional neural networks (CNNs)

In this exercise you will explore and evaluate the importance of various components of a classification CNN. This will be done by adding, modifying and removing different components. In each of these experiments you will need to implement the change, train the network, rationalize the results and document the results and your conclusions as to what happened and why in a pdf report that you submit (see Submission Guidelines below).

In the preparation of this report (as well as for reports of future assignments in this course) it is recommended to use tools designed for this purpose, either Weights and Biases (recommended), or another tool (e.g., tensorboard). Such tools will help you produce aesthetic plots of your experiments, along with the configuration/hyper-parameters used, and more.

The baseline network for this exercise can be found on this page (as part of a detailed pytorch tutorial). The network is trained over the CIFAR10 dataset, which consists of 10 object classes with 60000 32-by-32 RGB images in total (50000 training, 10000 test). Note that you can use the entire dataset, or a smaller subset of it, in some of the experiments.

Specifically, your tasks are:

1. **Architecture.** Implement a “standard” CNN consisting of a few convolutional layers that reduce the spatial image dimensions (via pooling), while increasing the number of feature maps (number of channels). Complete this construction by reshaping the responses into a vector and apply an output FC layer that produces the output.
    a. Plot the train and test losses and explore how they change as you change the total number of filters (neurons) in the network to the extent of overfitting and underfitting (also report the total number of learnable parameters).
    b. Find a configuration resulting with the lowest test error and use it for the following sections.
    c. Report these tests and their results in your report. The search space is huge and hence you can choose a strategy as to how you will increase and decrease the network’s size.

2. **Importance of Non-Linearity.** What happens to the performance of the network once you remove all the non-linear components it contains. List these components and explain the results obtained. Increase the network's size and see if the performance improves. Did this work? Explain.
3. **Cascaded Receptive Field.** In order to assess the importance of using a multi-scale CNN to obtain a large receptive field, let us achieve this using a shallower network. Thus, rather than shrinking the image size using repeated convolutions interleaved with pooling operators, let us directly map the output of the first convolutional layer to the output layer. This can be implemented in two ways: (i) use an FC layer on the activations of the first conv layer, or (ii) use a "global average pooling" operator where you basically compute the average of the activations in each channel (i.e., the image x and y dimensions collapse) and then apply an FC layer. Consider both options, make a choice and adapt the architecture to obtain a reasonably-sized model. Report your choices and their justification. Did this new network perform any better than the one in Part 1? Can you explain why?

We expect you to report and elaborate on every practical task in the report, using your own words and your own analysis of what you've done. Include everything that you think is crucial for us to understand your way of thinking.

Theoretical Questions:

1. Formally show that the condition L[x(i+k)](j) = L[x(i)](j+k) over a linear operator L receiving a 1D signal x(i) with offset k (the outer parentheses refer to the output signal), corresponds to a convolution. L operates over signals, i.e., it receives an input signal and outputs a signal, and the output signal index is denoted by j. Hint: decompose the signal x(i) into a weighted sum of translated delta functions, and then use the linearity of L. What input signal will output the underlining filter of the convolution?
2. When reshaping a 2D activation map (resulting from a convolutional layer) and feeding it into an FC layer, how important is the order (is there a good ordering or a bad ordering)? Why?
3. The convolution operator is LTI (linear translation invariant). What about the following operators:
    a. The ReLU activation function
    b. The strided pooling layer (i.e., the one that always picks the top-right pixel in each block).
    c. The addition of a bias
    d. Multiplication with a fully-connected matrix

    Explain your answers.

Submission Guidelines:

The submission is individual (in **singles**)**.** Please submit a single zip file named “ex1_ID.zip”. This file should contain your code, along with an ”ex1.pdf” file which should contain your answers to the theoretical part as well as the figures and analysis for the practical part. In addition, please include in this compressed archive a README with your name and cse username. Please write readable code, with documentation where needed, as the code will also be checked manually.

Tips: This is not a difficult assignment, but experimentation can be more time-consuming than it might seem. You have two weeks, start working on it sooner, rather than later.