Assignment 2

Advanced Deep Learning

6 October 2022

Use the results obtained in Assignment 1, Exercise 2, *Q3* to fix a neural architecture on which you got the best results in terms of accuracy and overfitting behaviour.

Adapt the code in `mnist-assignment.py` file (provided with Assignment 1) to solve the following exercises.

Exercise 1. *Modify the code of* `mnist-assignment.py` *to train the chosen network using SGD with minibatches of different sizes (starting from size 1). Discuss how the SGD method with various minibatch sizes compares in terms of speed of convergence and final accuracy on test set with GD.*

Exercise 2. *Initialize all the weights and all biases to a common value (for instance* 0*). Try to train the network either with GD or SGD. What happens? Why?*

Exercise 3. *Sort the example of MNIST by class label into a vector of examples* $(z_1, \dots, z_{60000})$ *in which* $z_i = (x_{\sigma(i)}, y_{\sigma(i)})$*, with* σ *being a permutation of* $\{1, s, \dots, 60000\}$ *with the property that*

$$y_{\sigma(i)} = \begin{cases} 0; & \text{if } 1 \le i \le n_0 \\ 1; & \text{if } n_0 < i \le n_0 + n_1 \\ \vdots & \\ 9, & \text{if } \sum_{k=0}^{8} n_k < i \le \sum_{k=0}^{9} n_k = 60000 \end{cases}$$

where n_j *is the number of example of class* j*. Then construct the sequence of examples* $(e_n)_{n>0}$ *defined as*

$$e_n := z_{n \bmod 60000}, \qquad n > 0.$$

This basically means: Order the MNIST dataset in order to have first all the "zeros", then all the "ones", then all the "twos" and so on, then create a stream $n \mapsto e_n$ *by cycling over this ordered list of example over and over. Of course the sequence* $(e_n)_{n>0}$ *is not unique (why?), just choose one. Then*

- *Use an Online Gradient method on the sequence of examples* $(e_n)_{n>0}$*.*
- *Compare it with the results of a SGD method with minibatch size* 1*.*

  Q1. Can you find values of the learning rate for which this online method is convergent? If so what is the speed of convergence compared to SGD? Discuss.

  Q2. Is the accuracy obtained on the test set comparable to the one that you obtain using SGD?

  Q3. Can the convergence analysis done for SGD in Lecture 3 be adapted to prove the convergence of online methods? Why?

Exercise 4. *Set up an experiment on MNIST to compare SGD and ADAM in terms of*

- *Sensibility to the learning rate (in ADAM the learning rate is the parameters that in the slides of Lecture 3 is indicated as* τ *and in the original article is denotes as* α*. In the PyTorch documentation it is called* γ *and in the code it is the variable that you set by specifying the argument* `lr` *of the Adam optimizer class.)*
- *Speed of convergence.*
- *Final accuracy on test set.*

* * *

Instructions for the answers

1. You are expected to collect your answers in a report that should be delivered in the form of a single PDF file for each assignment.
2. You need to structure your report so that in any point it is clear what is the question you are answering to.
3. Clarity and usage of appropriate non-ambiguous notation will have an influence in the final grade.
4. The conciseness of the answers will be as well an important parameters that will be taken into account in the final grade. In particular you are supposed to give complete and satisfactory answers without going out of topic or arguing on facts or results that are not relevant. [The Merriam-Webster dictionary defines the adjective concise as *"free from all elaboration and superfluous detail"*.]

* * *

Useful Resources
Besides the slides of the lecture and the notes of the lecture here are some useful resources that I encourage you to use:

- Chapter 1 of our companion book to the ML texbook *Machine Learning, a Constrained Based Approach* 2nd edition that you can find here `https://sailab.diism.unisi.it/mlbook/` by clicking on "Get Companion Book". In this chapter you can find a very broad introduction to Machine Learning and some information to set up an environment to manage Python installations and get acquainted with PyTorch. This is still a draft so please do not circulate it.
- Some snippets of code to gain some familiarity with Python and PyThorch can be found in the same webpage `https://sailab.diism.unisi.it/mlbook/` in the section "Code"–"Chapter 1".
- Useful resources for the Python website `https://www.python.org` and the PyTorch website `https://pytorch.org` where you can find information on the installation and a documentation of the languages.

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, 2006).

[2] V. N. Vapnik, *Statistical Learning Theory* (John Wiley & Sons, 1998).

[3] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning* (MIT Press, 2016).

[4] L. Ambrosio, N. Gigli, G. Savaré, Gradient Flows in Metric Spaces and in Spaces of Probability Measures (Birkhäusen Verlag, 2005) This is a very advanced and technical book, be careful!

[5] D. P. Kingma, J. L. Ba, `arXiv:1412.6980`