MICAS901 - Introduction to Optimization: Final Project

Please submit your solution by email before
December 1st, 2020

Hadi Ghauch, Telecom Paris,
email: hadi.ghauch@telecom-paristech.fr

All the steps and derivations are needed for full credit. The solution for each project should be submitted individually by each student (one student per project). Regarding the questions that need programming or coding, feel free to use any programing language you are comfortable with. The project is worth 50% of total course grade.

Build your own DNN!

Part 1: deep linear network

We will start with a 2-layer deep linear network, which is a special case case obtained by setting all activation functions to an identity. Thus, the corresponding training problem is given by the following optimization problem:

$$\min_{\boldsymbol{W}_1,\boldsymbol{W}_2} \frac{1}{N}\sum\nolimits_{i\in[N]} \|\boldsymbol{W}_2\boldsymbol{W}_1\boldsymbol{x}_i - \boldsymbol{y}_i\|_2^2 + \lambda_1\|\boldsymbol{W}_1\|_F^2 + \lambda_2\|\boldsymbol{W}_2\|_F^2 \tag{1}$$

We will use the BodyFat data set. The goal of this part is to solve (1) by implementing and comparing several methods: backpropagration (BP), minibatch SGD with an appropriate minibatch size, ADAM, and block-coordinate descent (BCD). Ensure that you select a 'good choice' the model complexity (dimensions of $\boldsymbol{W}_1$, $\boldsymbol{W}_2$), then fix them for all the above methods. Recall that, it is better to start with complex model (large dimensions for $\boldsymbol{W}_1$, $\boldsymbol{W}_2$), and mitigate overfitting with regularization.

a) **BP with constant step-size**: derive the equations of BP for this problem (use the matrix form for conciseness), and implement them.
Find the best hyper-parameters, λ_1, λ_2, and the stepsize, using cross validation on a help-out test set.

b) **mini-bath SGD**: For the SGD method, you may use the mini-batch version. Write the update eqts for each layer as a function of the mini-batch and implement these updates.
Find the best hyper-parameters, λ_1, λ_2, the stepsize, using cross validation on a held-out test set.
Propose your method to select the batch size. Explain your reasoning
Sanity check: set the batch size equal to the training set, and verify (numerically) that you recover a similar behavior as BP (from part a))

c) **ADAM**: Refer to algorithm presented in the lecture. Write the update eqts of ADAM for each layer as a function of the exponential decay factors for the first and second order moments, β_1, β_2.
Find the best hyper-parameters, λ_1, λ_2, the stepsize, and β_1, β_2 using cross validation on a held-out test set.

d) **BCD**: Derive each BCD subproblem, and the resulting update eqts for each layer, in closed (matrix) form. You may assume for this part that $\boldsymbol{W}_1$ is a tall matrix, and $\boldsymbol{W}_2$ is fat. Implement the resulting algorithm.
Find the best hyper-parameters, λ_1, λ_2, using cross validation on a held-out test set.

e) Compare the training error and test error for all these methods, after convergence is reached (with an appropriate choice of convergence criteria for each). Then computing their test MSE on a held-out set for cross validation.

f) Compare these methods in terms of computational complexity, complexity of hyper-parameter tuning, and the number of # outer-loop iterations

g) How do the derivations of BP change, when another loss function is used, e.g. cross-entropy ? Find an efficient way to write/derive equations of BP, when using the cross entropy loss function. This observation is a significant advantage of BP.

Hints: Notice that you will to compute derivative w.r.t a matrix. You may consult this guide on matrix differentiation: `https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf` You may also use the matrix differentiation rules in Lecture 1. You will need to use the invariance of the trace with respect to circular permutation. For matrices of appropriate dimensions, $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$, the following holds:

$$\text{tr}(\boldsymbol{ABC}) = \text{tr}(\boldsymbol{CAB}) = \text{tr}(\boldsymbol{BCA})$$

Part II: deep neural network

Let us now add the following logistic activation, where $\sigma(\boldsymbol{x}) = 1/(1+\exp(-\boldsymbol{x}))$ at each layer. Recall that, the function σ is applied in an element-by-element manner on its input argument. Find an appropriate normalization of the training set, to take into account the fact that the logistic activation 'compresses' its output between $[0, 1]$. Thus, the optimization problem considered here is the following:

$$\min_{\boldsymbol{W}_1, \boldsymbol{W}_2} \frac{1}{N} \sum\nolimits_{i \in [N]} \|\sigma(\boldsymbol{W}_2 \sigma(\boldsymbol{W}_1 \boldsymbol{x}_i) - \boldsymbol{y}_i\|_2^2 + \lambda_1 \|\boldsymbol{W}_1\|_F^2 + \lambda_2 \|\boldsymbol{W}_2\|_F^2 \tag{2}$$

a) Redo the questions from Part I-a) to Part I-f)
What is the limitation in applying the BCD method (in Part I-d)) to optimization problem (2) ?

b) Compare the test/prediction error of all the method in Parts I and II. Give a qualitative argument to explain this difference.