TP1: Levelwise Derivatives

Thursday 3rd October, 2019
joao.candido@hesge.ch
Deadline: Wednesday 16th October, 2019, 23:59

1 Objective

The goals of this TP is to understand and implement the levelwise derivatives. This TP is very similar to "TP 7: Logistic and Softmax regression" you did last semester. The main difference is that we ask you all the derivatives/gradients computed during the backpropagation. We will implement again the softmax classifier and will extend it with another layer to build an MLP.

You are going to use the *CIFAR-10* data set. If you are using **windows** the get_dataset.sh script will not work (only mac and linux are available), you can ask other students the data folder (or send me an email)!

2 Detailed Instructions

You are going to fill a few missing functions in the python script[1] and in jupyter notebook to implement the exercises that we ask. So first of all read and understand the given python script. To run your code you have to run the TP1_Levelwise_derivatives.ipynb notebook. You are going to use the Cifar 10 data set.

You have to send a **formal** report and your code.

For this TP the following steps will need to be done for the Softmax classifier and the MLP, all this formulas and derivation have to be in your report. If you don't include in your report your derivations **your code will not be taken into account**. Same if your code is not based on your derivations.

2.1 Softmax classifier

1. The forward step, i.e. the computation of the probabilities (i.e. *forward()* method).

[1] Part of the given code is based on Stanford's repository

2. The cost function of the softmax classifier (i.e. *loss()* method).
   (a) Write down the cost function of the softmax classifier on your report.
   (b) Derive the gradient of the softmax classifier. You can follow the comments in *backward()* method to know which derivatives you need (you should present 7 formulas).
   (c) Write down the cost function of the logistic regression when a L_2 regularizer is added.
   (d) Derive the the gradient of the logistic regression cost when a L_2 regularizer is added.
3. Based on your derivations, implement the *backward()* method in the softmax classifier class and its derivative with an L_2 regularizer. **Reminder:** All the formulas and derivations (i.e. how do you get a given formula) must be in the report ! Otherwise the code it's not taken into account.
4. Implement Stochastic Gradient Descent, SGD
   (a) Fill the missing part of the *train()* method inside the softmax_classifier.py script.
5. Fill the missing part in TP1_levelwise_derivatives.ipynb notebook and train your classifier for different learning rates and regularization strengths.
   (a) This section should be very similar to what you did in last TP (maybe the same ?).

2.2 Multilayer Perceptron (MLP)

Follow exactly the same thing as before (softmax classifier) but with a MLP. The MLP can be seen as an extension of the number of layers of the softmax classifier (which has only one layer). In this exercise we ask you to implement an MLP with two layers. The output of the first layer is the input of the second layer.

All the steps, formulas derivations have to be in report to make your code count! Instead of 7 formulas in 2.1.2.(a), now you have **11 formulas** to report.

3 Reminders

- When we minimize the cost function using Gradient Descent (GD) the weights are updated after seeing all the training instances
- When we minimize the cost function using Stachastic Gradient Descent (SGD) the weights are updated after seeing a mini batch the training instances.

- The L2 regularizer is also called Ridge Regression. It adds "squared magnitude" of weights as penalty term to the cost function. If the regularizer parameter is zero then you can imagine we get back to the cost function.

4 Comments

- This TP is very similar to the previous one, you can take code for your previous work if it fits the requirements of this one. We focus on the derivations and implementation of local and global derivatives.
- All the material needed to complete this TP is in the course, take a look at it before ask a question!
- As usually, we try to give you tests for your implementation. If you're sure your implementation is correct and the test fails, please contact me!

General instructions

You have to put your work in *cyberlearn* saved in a zip file using as name this format: TP_7_LASTNAME_Firstname. You can clean the *datasets* folder by running *clean.sh* before upload your work.