Optimization Methods for Machine Learning - Fall 2017

Assignment # 1 - MLP and Generalized RBF Network

Laura Palagi

Dip. di Ingegneria informatica automatica e gestionale A. Ruberti, Sapienza Università di Roma

Posted on November 6, 2017 - due date December 4, 2017

Instructions

Homework will be done in groups formed by 1 to 3 people: each group must hand in their own answers. We will be assuming that, as participants in a graduate course, every single student will be taking the responsibility to make sure his/her personal understanding of the solution to any work arising from such collaboration.

Homework must be send by an email both to the teaching assistant Ing. Tommaso Colombo (tommaso.colombo@uniroma1.it) and to laura.palagi@uniroma1.it with subject **[OMML-2017] Project 1**. After you submit, you will receive an acknowledgement email that your project has been received. If you have not received an acknowledgement email within 2 days after you submit then contact the instructors.

The mail must contain as attachment a .zip or .tar.gz file with both a typed report in English and the source code following instructions in the text of the project. **The report must be of at most 5 pages excluded figures that must be put at the end.**

Evaluation criteria Homework is due at latest at midnight on the due date. For late homework, the score will be decreased. It is worth 85% for the next 48 hours. It is worth 70% from 48 to 120 hours after the due date. It is worth 50% credit after 120 hours delay.
You have three questions Q1, Q2 both with two points (Q1.1; Q1.2; Q2.1, Q2.2) and Q3.
The grade are Italian style namely in the range [0,30], being 18 the minimum degree to pass the exam.
Answering only the first questions allows to get up to 24. Answering questions 1 and 2 allows to get up to 28. Answering Q1, Q2 and Q3 allows to get up to 31 (30 cum laude). Please note also the last summary point of the homework (mandatory).
The first homework accounts for 35% of the total vote of the exam.
For the evaluation of the first homework the following criteria will be used:

1. check of the implementation (60% Tommaso Colombo)
2. Quality of the explanation document and of the overall job (40% - Laura Palagi).

In this assignment you will implement neural networks for regression. We want to reconstruct in the region $[0,1]\times[0,1]$ the Franke's function (see http://www.sfu.ca/∼ ssurjano/franke2d.html):

$$\begin{aligned} F(x) \quad &= 0,75\exp\left(-\frac{(9x_1-2)^2}{4}-\frac{(9x_2-2)^2}{4}\right)+0,75\exp\left(-\frac{(9x_1+1)^2}{49}-\frac{(9x_2+1)}{10}\right) \\ &+0,5\exp\left(-\frac{(9x_1-7)^2}{4}-\frac{(9x_2-3)^2}{4}\right)-0,2\exp\left(-(9x_1-4)^2-(9x_2-7)^2\right) \end{aligned}$$

The data set is obtained by sampling on 100 random points x^i the function and adding a uniform noise, i.e. $y^i = f(x^i)+\varepsilon^i$ and ε^i is a random number in $[-10^{-1},10^{-1}]$. You can use the Numpy.random library in Python and you must initialize the generator using the leader's student number (matricola) as a seed. The data set obtained is

$$\{(x^p,y^p):\ x^p\in\mathbb{R}^2,\ y^p\in\mathbb{R},\ p=1,\ldots,100\}.$$

You divide the data set for the learning problem obtained by random sampling into a training set and a test set (choose percentage of training data not less than 70%). Let P be the number of instances in the training set, as you fixed them.

Question 1. (Full minimization) Consider a shallow Feedforward Neural Network (FNN) (one only hidden layer) that provides the model $f(x)$. We will select either MLP or RBF network, being ω the parameters to be settled by the optimization procedure and π the hyper-parameters of the network to be settled by means of an heuristic procedure using the value of the test error.

The parameters ω are find by minimizing the regularized training error

$$E(\omega;\pi)=\frac{1}{2P}\sum_{p=1}^{P}\|f(x^p)-y^p\|^2+\rho\|\omega\|^2 \tag{1}$$

and ρ in the range $[10^{-5}\div 10^{-3}]$ is one of the hyper parameters.

1. (max score up to 22) Consider a shallow MLP with linear output unit, so that

$$f(x)=\sum_{j=1}^{N}v_j g\left(\sum_{i=1}^{n}w_{ji}x_i-b_j\right)$$

where $\omega=(v,w,b)$. The number of neurons N of the hidden layer is an hyper- parameter. It must be possible to specify N as an input parameter.

The activation function $g(\cdot)$ is the *hyperbolic tangent*

$$g(t):=\tanh(t/2)=\frac{1-e^{-\sigma t}}{1+e^{-\sigma t}} \tag{2}$$

where σ is an hyper parameter. (g is available in Python with $\sigma=1$: Numpy.tanh)

Write a program which implements the regularized training error function $E(v,w,b)$ (as obtained by (1) using $f(x)$ of the MLP network) and uses a Python routine of the optimization toolbox (scipy.optimize) to determine the parameters v_j, w_{ji}, b_j which minimize it. The code must produce a plot of the approximating function found.

Please note that it is not required to evaluate the gradient (there are Python routines to approximate gradient in scipy.optimize), but it may be useful to use it.

Analyse the occurrence of overfitting/underfitting varying the number of neurons N and the parameters ρ and σ (you can also decide to fix $\sigma = 1$ if using the Phyton implementation of Tanh).

In the report you must state:

- the final setting for N, ρ and σ; how do you choose them and if you can put in evidence over/under fitting, when and if you can explain why;

- which optimization routine do you use for solving the minimization problem, the setting of its parameters (optimality accuracy, number of iterations ecc) and the returned message in output (successful optimization or others, computational time, number of its or number of function/gradient evaluations, final accuracy ecc) if any;

- the value of the error on the training and test set;

- the plot of the function representing the approximating function obtained by the MLP in comparison with the true one.

2. (max score up to 24) Consider an RBF network

$$f(x) = \sum_{j=1}^{N} v_j \phi(\|x^i - c_j\|)$$

where $\omega = (v,c)$ $v \in \mathbb{R}^N$ and $c_j \in \mathbb{R}^2$ $j = 1,\ldots,N$.

You must choose as RBF function $\phi(\cdot)$ the Gaussian function

$$\phi(\|x - c_j\|) = e^{-(\|x-c_j\|/\sigma)^2} \quad \sigma > 0 \tag{3}$$

where σ ia an hyper-parameter. You need to have the number of RBF units N of the hidden layer and the positive parameter σ in the RBF function as parameters that can be changed as input.

Write a program which implements the regularized training error function of the RBF network $E(v,c)$ (as obtained by (1) using $f(x)$ of the RBF network) and uses a Python routine of the optimization toolbox for its minimization with respect to both (v,c). The code must produce a plot of the approximating function found.

Please note that gradient is not required to be evaluated but it may be useful to use it.

Analyse the occurrence of overfitting/underfitting varying the number of units N, the spread parameters σ and ρ.

In the report you must state:

- the values of the parameters ρ

- the final setting for the number of neurons N and of the parameter σ; how do you choose them and if you can put in evidence over/under fitting, when and why, if you have an explanation for this behaviour;

- which optimization routine do you use for solving the minimization problem, the setting of its parameters (optimality accuracy, number of iterations ecc) and the returned message in output (successful optimization or others, computational time, number of its or number of function/gradient evaluations, final accuracy ecc) if any;

- the value of the error on the training and test set;

- the plot of the function representing the approximating function obtained by the RBF network in comparison with the true one.

- a comparison of performance between MLP and RBF networks both in terms of quality of approximation (training and test error) and in efficiency of the optimization (number of function/gradient evaluation and computational time needed to get the solution). Please put these values in a table.

Question 2. (Two blocks methods)

1. (max score up to 26) Consider again the shallow MLP with linear output unit as defined at Ex. 1 of Question 1. Use the values of N, ρ, σ you fixed at Ex. 1.

   Write a program which implements an *Extreme Learning* procedure, namely that fix randomly the values of w_{ij}, b_j for all i, j and use a Python routine or any other optimization library to minimize the quadratic convex training error $E(v)$ of the only variables $v \in \mathbb{R}^N$.

   For the random generation of w, b you can make any choice. We suggest to identify a range and repeat the random choice more than once.

   In the report you must state:

   - which optimization routine do you use for solving the quadratic minimization problem and the setting of its parameters. Compare performance of the optimization process w.r.t Full optimization of Question 1 (Ex. 1)

   - the values of the error on the training and test set; Compare these values with the ones obtained by the Full optimization of Question 1 (Ex. 1)

   - the plot of the function representing the approximating function obtained by the Extreme Learning MLP in comparison with the one obtained by Full minimization.

2. (max score up to 28)

   Consider again a RBF network as in Ex. 2 of Question 1.

The number of RBF units N of the hidden layer and the positive parameter σ in the RBF function are those chosen at Ex 2. Question 1.

Write a program which implements a method with unsupervised selection of the centers. You can either select the centers randomly on the P points of the training set or used a Clustering procedure as implemented in Phyton.

Choose the weights by minimizing the regularized error $E(v)$ using a suitable Python routine or any other optimization library to minimize the quadratic convex function.

In the report you must state:

- which optimization routines of the Optimization toolbox do you use to solve the weights subproblems and the setting of its parameters.

- the value of the error on the training and test set;

- the plot of the function representing the approximating function obtained by the RBF with unsupervised selection of the centers in comparison with the true one.

Question 3. (Decomposition method)

3. (max score up to 31)

Consider either the shallow MLP or the shallow RBF network.

Write a program which implements a two block decomposition method which alternates the convex minimization with respect to the output weights v and the non convex minimization woth respectto the other parameters (either (w,b) for the MLP or the centers c for the RBF network respectively).

Set the regularization parameter ρ, number of neurons N nd spread σ at the values you selected in Question 1.

You must use appropriate Python routines of the optimization toolbox for solving the two block minimization problems. In this case it is required that you calculate the gradient of the nonconvex block.

Compare the results with those obtained at the corresponding exercise of Question 1.

In the report you must state:

- which optimization routines of the Optimization toolbox did you use to solve the two block subproblems and the setting of the parameters. Consider to adjust accuracy in optimization wity increasing iterations;

- the stopping criteria of the decomposition procedure; consider the use early stopping rule.

- the number of outer iterations (number of subproblems solved), number of function/gradient evaluation and computational time needed to get it.

- the value of the error on the training and test set;

- the plot of the function representing the approximating function obtained by the RBF with block decomposition in comparison with the true one.

- Comparison with the corresponding network obtained by random selection at the corresponding Exercise of Question 2. Comparison are both in the quality of the network (test error), and in the efficiency of the optimization procedure (number of function/gradient evaluation and computational time needed to get the solution). Please put these values in a table.

4. (MANDATORY) In the final report (pdf) the comparison among all the implemented methods - in term of accuracy in learning and computational effort in training - must be gathered into a final table (a minimal sample below)

Ex	FFN	settings	Training error	test error	optimization time
Q1.1	Full MLP	$N =?$, $\sigma =?$, $\rho =?$			
Q1.2	Full RBF	$N =?$, $\sigma =?$, $\rho =?$			
Q2.1	Extreme MLP	$N =?$, $\sigma =?$, $\rho =?$			
Q2.2	Unsupervised c RBF	$N =?$, $\sigma =?$, $\rho =?$			
Q2.3	The ONE you chose				