Universität des Saarlandes
Prof. Dr. Dietrich Klakow
Lehrstuhl für Signalverarbeitung
NNTI Winter Term 2020/2021

Exercise Sheet 8

Optimization

(Proposed Solutions)

Deadline: 19.01.2021, 23:59

Instructions

This assignment covers the concepts of optimization in machine learning. For references, check Chapter 8 from the lecture slides and the DL Book.

Exercise 8.1 - Empirical Risk Minimization (1 + 1 + 1 = 3 points)

You have been introduced to the concept of *Empirical Risk Minimization (ERM)* in the lecture. Consider the risk $\mathcal{R}$ of loss $\mathcal{L}$ for the hypothesis space $h \in \mathcal{H}$ over training data $\mathcal{D}_n$. Refer to chapter 8 slide 4 and try to answer the following questions.

a) What is *empirical risk* and how is it different from *expected risk*? Provide a suitable mathematical expression for *empirical risk*.

b) ERM is a generalization of many popular methods in machine learning. Mention two such methods you can derive from the ERM expression in *part (a)* stating the necessary changes you need to do to get there.

c) What is the main drawback of ERM? State two ways to tackle this problem (with mathematical expressions).

Proposed Solution 8.1

a) Empirical risk $\mathcal{R}_n(h)$ is the average loss of an estimator f for a finite set of data $\mathcal{D}_n$ drawn from a distribution P while expected risk is the expectation, over all possible datasets, of the difference between a model's predictions and the actual targets. Because we do not know the original distribution P, we instead minimize the empirical risk over a training dataset $\mathcal{D}_n$ drawn from P.

$$\mathcal{R}_n(h) = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(h(x_i), y_i)$$

b) *Least Squares Estimation*: When we take $\mathcal{H} = \{h(x) : h(x) = \theta^T x, \forall\theta \in \mathbb{R}^p\}$ and $\mathcal{L}(y, p) = (y - p)^2$.
Maximum Likelihood Estimate: Negative log-likelihood loss function.

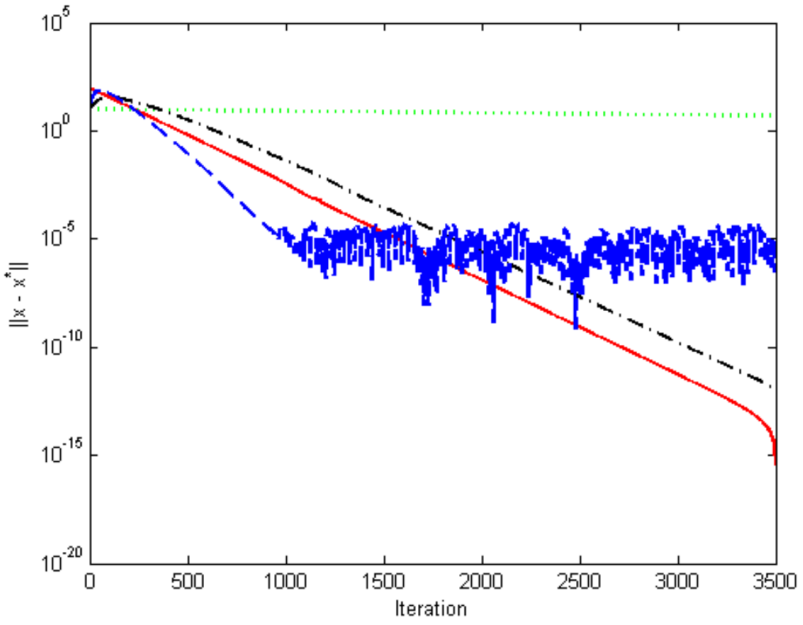
$$\hat{\theta} = \text{argmin}\frac{1}{n}\sum_{i=1}^{n}\log(1 + \exp(-y_i\theta^T x_i))$$

c) ERM is prone to *overfitting*. It can be solved by:

- Method of *Sieves* and *Structured Risk Minimization (SRM)*: Take $\mathcal{H}_1$, $\mathcal{H}_2$,..., $\mathcal{H}_n$,... to be a sequence of increasing sized spaces. For example, one typically has $\mathcal{H}_k \subset \mathrm{H}_{k+1}$ and $\bigcup \mathrm{H}_k = \mathcal{H}$. Given the training data $\mathcal{D}_n$, one finds $\hat{h}_n$ by minimizing $\hat{h}_n = \underset{h \in \mathcal{H}_n}{\operatorname{argmin}} \hat{R}_n(h)$.
- *Penalized ERM* or *Regularization*: $\hat{h}_n = \underset{h \in \mathcal{H}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y_i - h(x_i)) + \lambda_n \Omega(h)$, where $\lambda_n > 0$ balances the trade-off between goodness-of-fit and model complexity.

Exercise 8.2 - First-Order Optimization (2 + 1 + 1 = 4 points)

You have learned about different first-order optimization methods from SGD to Adam. Review these concepts in the lecture slides 14-32 of chapter 8 and page 290 onward in the DL Book and try to answer the following questions.



a) The figure above plots the losses of four models using four different first-order optimization strategies to solve an unconstrained minimization problem for a function f which is L-smooth and m-strongly convex. The optimization methods used are as follows:

- **Method 1**: A gradient method with step-size $\alpha = \frac{1}{L}$
- **Method 2**: A method with convergence rate $f(x_k) - f(x^*) \leq (1 - 2\sqrt{\frac{m}{L}})^k C$ (where C is a constant)
- **Method 3**: SGD + Momentum
- **Method 4**: Nesterov's Accelerated Gradient (NAG) (DL book p. 296)

Your task is to match the lines in the plot to the four optimization methods listed above. Explain your predictions (no mathematics needed but your explanations should support your choices from a mathematical perspective).

b) You would have seen that most models prefer to use *adam* as its optimizer due to its performance. However, Wilson et al. (2017) states that adaptive methods often generalize worse than SGD even though they have better training performance. Read the paper and mention why you think this happens. What would you suggest as an alternative approach to solve this? Try to think on your own.

c) Following the order `SGD` → `Momentum` → `AdaGrad` → `RMSProp` → `Adam`, state a drawback of each method and how its successor (in the order) attempts to solve (generally) that problem. (*1 sentence each*)

Proposed Solution 8.2

a) The green curve is **Method 1** (the gradient method) since its guaranteed convergence rate is much slower than the rates of Nesterov's method and the triple momentum method. The convergence rate of the triple momentum method is slightly faster than Nesterov's method (by a constant factor). Heavy-ball method is not guaranteed to converge for general smooth strongly-convex functions and may oscillate. Therefore, the black curve is **Method 4** (Nesterov's method), and the red curve is **Method 2** (the triple momentum method). Then the blue curve is **Method 3** (the Heavy-ball method or gradient with momentum).

Exercise 8.3 - Second-Order Optimization (1 + 1 + 0.5 + 0.5 = 3 points)

You have learned about the second-order Newton's method of optimization. Refer to page 307 onward in the DL Book and try to answer the following questions.

a) Why do we need second-order methods when we already have first-order optimization algorithms? State an important condition for second-order methods to work.

b) State the expression for the second-order *Taylor* expansion and the corresponding parameter update steps for *Newton's method*. Justify (with mathematical reasoning) the usability of this method.

c) *Quasi-Newton* algorithms are an improvement over the existing second-order methods. Discuss how they improve *Newton's method*.

d) The most famous *Quasi-Newton* method is the `BFGS` algorithm. Read about this method (DL book page 312) and explain its functionality how it overcomes the problem of Newton's method.

Note: You can refer additional resources for parts (c) and (d) if needed. If you do, make sure to cite the references that you used.

Proposed Solution 8.3

a) Second-order optimization is the advancement of first-order optimization in neural networks. It provides an additional curvature information of an objective function that adaptively estimates the step-length of optimization trajectory in training phase of neural network.
Second-order optimization methods work if and only if the Hessian matrix is positive definite everywhere, which means the function $f(x)$ is *convex*.

b) Second-order Taylor expansion:

$$L(w) \approx L(w_0) - (w - w_0)^T \nabla_w L(w_0) + \frac{1}{2}(w - w_0)^T \mathbf{H}_w L(w_0)(w - w_0)$$

where $\mathbf{H}_w$ is the Hessian matrix.
Solving for the critical point we obtain the Newton parameter update:

$$w^* = w_0 - \mathbf{H}_w L(w_0)^{-1} \nabla_w L(w_0)$$

It is impractical use Newton's method for deep neural networks (N=tens or hundreds of millions) because the Hessian has $O(N^2)$ elements and inverting it takes $O(N^3)$.

c)

d) A nice explanation can be found here.

Submission instructions

The following instructions are mandatory. If you are not following them, tutors can decide to not correct your exercise.

- You can and are encouraged to submit the assignment as a team of two students. Submitting as a team will be mandatory for the next assignment.
- Hand in zip file containing the PDF with your solutions and the completed iPython notebook, or you could also hand in just the iPython notebook containing everything.
- Therefore make sure to write the Microsoft Teams user name, student id and the name of each member of your team on your submission.
- Your assignment solution must be uploaded by only **one** of your team members to the 'Assignments' tab of the tutorial team (in **Microsoft Teams**).
- If you have any trouble with the submission, contact your tutor **before** the deadline.