00103335: Deep Learning and Reinforcement Learning
Homework 2

Note. Unless otherwise noted, section and equation numbers refer to those in DL (Goodfellow et al.).

1. Consider the XOR problem described in Section 6.1.
   (a) For a perceptron with MSE loss and linear output, verify that the solution is $\boldsymbol{w} = \mathbf{0}$ and $b = 1/2$.
   (b) Is the problem solvable by a perceptron with cross-entropy loss and sigmoid output? Find the solution in this case.
   (c) Is the solution for the two-layer feedforward network unique? If not, find a solution different from the one given in the book.

2. Prove that the solutions to optimization problems (6.14) and (6.16) are the conditional mean and median of $\boldsymbol{y}$ given $\boldsymbol{x}$, respectively.

3. Numerical differentiation is an alternative approach to back-propagation for computing the gradient. This can be done, for example, by applying the central difference approximation

$$\frac{\partial J}{\partial \theta} = \frac{J(\theta + \varepsilon) - J(\theta - \varepsilon)}{2\varepsilon} + \text{remainder}$$

   to each parameter of the network.
   (a) Show that the remainder term is $O(\varepsilon^2)$.
   (b) Determine the time complexity of this algorithm and compare it with that of back-propagation.

4. It is mentioned in Section 7.5 that, "For some models, the addition of noise with infinitesimal variance at the input of the model is equivalent to imposing a penalty on the norm of the weights." State this formally for a feedforward network with MSE loss and prove your claim.

5. Consider a feedforward network with one hidden layer $\boldsymbol{h}$ and regularized loss (7.48), where $\Omega(\boldsymbol{h}) = \|\boldsymbol{h}\|_1$. Devise a back-propagation algorithm to solve this problem.

6. Prove that the weight scaling inference rule is exact for
   (a) regression networks with conditionally normal outputs;
   (b) deep networks with softmax outputs and linear hidden layers.

7. State and prove a convergence theorem for stochastic gradient descent under conditions (8.12) and (8.13). *Hint*: See Robbins and Monro (1951, *Ann. Math. Statist.*, 22, 400–407).

8. In this exercise, we establish a convergence result for gradient descent with Polyak averaging.
   (a) Let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_T$ be an arbitrary sequence of vectors. For the algorithm with initialization $\boldsymbol{w}^{(1)} = \mathbf{0}$ and update rule

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \boldsymbol{v}_t,$$

   show that

$$\sum_{t=1}^{T} \langle \boldsymbol{w}^{(t)} - \boldsymbol{w}^*, \boldsymbol{v}_t \rangle \leq \frac{\|\boldsymbol{w}^*\|^2}{2\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \|\boldsymbol{v}_t\|^2.$$

   (b) Let f be a convex, ρ-Lipschitz function, $\boldsymbol{w}^* = \arg\min_{\|\boldsymbol{w}\| \leq B} f(\boldsymbol{w})$, and $\overline{\boldsymbol{w}} = \sum_{t=1}^{T} \boldsymbol{w}^{(t)} / T$. Use part (a) to show that the gradient descent algorithm for minimizing f with $\eta = B/(\rho\sqrt{T})$ satisfies

$$f(\overline{\boldsymbol{w}}) - f(\boldsymbol{w}^*) \leq \frac{B\rho}{\sqrt{T}}.$$

9. Represent the convolution example in Figure 9.1 (3×4 input, 2×2 kernel, "valid" convolution) as matrix multiplication with a doubly block circulant matrix.

10. Consider the pooling example in Figure 9.9. Design a set of filters such that the max pooling unit can learn to be invariant to (a) rotation, and (b) scaling.

11. The Hopfield network is a type of recurrent network consisting of n units with states s_i and update rule

$$s_i \leftarrow \sigma\left(\sum_{j \neq i} w_{ij} s_j - \theta_i\right),$$

where $\sigma(x) = 2I(x \geq 0) - 1$, $w_{ij} = w_{ji}$, and $w_{ii} = 0$. The network is updated in an asynchronous manner, so that one unit is randomly selected and updated at each time step. Prove that the network will eventually reach a stable state at a local minimum of the energy function

$$E(\boldsymbol{s}) = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} s_i s_j + \sum_{i=1}^{n} \theta_i s_i.$$

12. Design a recurrent neural network to approximate the dynamics of the Lorenz 96 model

$$\frac{dx_i}{dt} = (x_{i+1} - x_{i-2})x_{i-1} - x_i + F, \quad i = 1, \ldots, n,$$

where F is a forcing constant and the indices are cyclic so that $x_{-1} = x_{n-1}$, $x_0 = x_n$, and $x_{n+1} = x_1$.

13. UML (Shalev-Shwartz and Ben-David) Exercise 20.1

14. UML Exercise 20.2

15. UDL (Prince) Problem 12.3

16. UDL Problem 12.6

17. UDL Problem 12.10