Exercises for the course
Deep Neural Networks
Summer semester 2018

Machine Learning Group
Department of Electrical Engineering and Computer Science
Fakultät IV, Technische Universität Berlin

Exercise Sheet 1

Exercise 1: Log-Likelihood (10 + 10 P)

Let μ be the output of a neural network predicting the mean of some Laplacian distribution

$$p(t;\mu) = \frac{1}{2\sigma}\exp\Big(-\frac{|t-\mu|}{\sigma}\Big).$$

We collect a dataset composed of neural network inputs $\boldsymbol{x}_1,\dots,\boldsymbol{x}_N$ and associated targets $t_1,\dots,t_N$. We denote by $\mu_1,\dots,\mu_N$ the predictions of the neural network for these points. We would like to optimize the neural network to produce predictions that match the targets. For this, we use the negative log-likelihood loss function given by:

$$\ell(\mu,t) = -\log p(t;\mu).$$

(a) *Compute* the gradient of the loss function with respect to the output μ of the neural network.

(b) We now assume that the neural network is made of a single linear layer: $\mu = \boldsymbol{w}^\top \boldsymbol{x}$, where $\boldsymbol{w}$ is the vector of parameters to be learned. *State* the chain rule for transmitting the gradient from the output of the neural network to the model parameters, and use it to compute the gradient of the loss function with respect to $\boldsymbol{w}$.

Exercise 2: Shared Parameters (10 + 10 + 10 P)

Let x_1, x_2 be two observed variables. Consider the two-layer neural network that takes these two variables as input and builds the prediction y by computing iteratively:

$$z_3 = x_1 w_{13}, \quad z_4 = x_2 w_{24}, \qquad a_3 = 0.5\, z_3^2, \quad a_4 = 0.5\, z_4^2, \qquad y = a_3 + a_4.$$

(a) *Draw* the neural network graph associated to these computations.

We now consider the loss function $\ell(y,t) = \frac{1}{2}(y-t)^2$ where t is a target variable that the neural network learns to approximate.

(b) Using the rules for backpropagation, compute the derivatives $\partial\ell/\partial w_{13}$ and $\partial\ell/\partial w_{24}$ required for gradient descent.

Let us now assume that w_{13} and w_{24} cannot be adapted freely, but are a function of the same shared parameter v:

$$w_{13} = \log(1+\exp(v)) \qquad \text{and} \qquad w_{24} = -\log(1+\exp(-v))$$

(c) Assume you have already computed $\partial\ell/\partial w_{13}$ and $\partial\ell/\partial w_{24}$. *State* the multivariate chain rule that links the gradient $\partial\ell/\partial v$ the these quantities and *apply* it in order to get an analytic expression of $\partial\ell/\partial v$.

Exercise 3: Layered Networks (10 + 10 P)

Consider a deep neural network made of several layers of neurons. Let $\boldsymbol{a}^{(l)} = (a_1^{(l)},\dots,a_d^{(l)})$ be the activations of the d neurons at layer (l), and $\boldsymbol{a}^{(l+1)} = (a_1^{(l+1)},\dots,a_h^{(l+1)})$ be the activations of the h neurons at layer $(l+1)$. These activations are computed as:

$$\forall_{j=1}^{h} : z_j^{(l+1)} = \textstyle\sum_{i=1}^{d} a_i^{(l)} w_{ij} + b_j \qquad a_j^{(l+1)} = \exp(z_j^{(l+1)})/(1+\exp(z_j^{(l+1)}))$$

where w_{ij}, b_j denote the parameters of the mapping between these two layers.

Show by application of the chain rule that the following relations hold:

(a) $$\frac{\partial\ell}{\partial z_j^{(l+1)}} = a_j^{(l+1)}\cdot(1-a_j^{(l+1)})\cdot\frac{\partial\ell}{\partial a_j^{(l+1)}}.$$

(b) $$\frac{\partial\ell}{\partial \boldsymbol{a}^{(l)}} = W\cdot\Big(\boldsymbol{a}^{(l+1)}\odot\big(1-\boldsymbol{a}^{(l+1)}\big)\odot\frac{\partial\ell}{\partial \boldsymbol{a}^{(l+1)}}\Big).$$

where W is a matrix of size $d\times h$ containing the parameters w_{ij}, and where $\odot$ denotes the element-wise product.

Exercise 4: Programming (30 P)

Download the programming files on ISIS and follow the instructions.