Labs
Machine Learning Course
Fall 2019

EPFL
School of Computer and Communication Sciences
Martin Jaggi & Rüdiger Urbanke
www.epfl.ch/labs/mlo/machine-learning-cs-433

Problem Set 12, Dec 5, 2019 (Neural Networks)

Goals. The goal of this exercise is to

- Better understand neural networks
- Implement the feed-forward function and backpropagation in a simple neural net.

Setup, data and sample code. Obtain the folder `labs/ex12` of the course github repository

github.com/epfml/ML_course

In the following problems, we will use a very simple neural network. Let's assume we have a three-layer neural net with one input layer of size $D = 4$, $L = 1$ hidden layers of size $K = 5$, and one output layer of size 1, as shown in Figure 1.

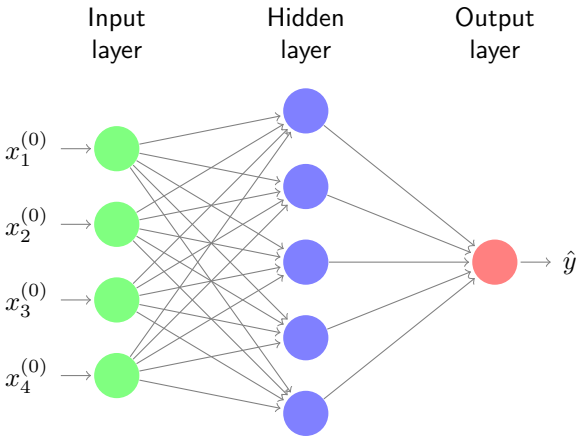

Figure 1: A simple neural network.

Problem 1 (Feed-forward in neural networks):

In our simplified neural network, we have the feed-forward function shown below:

$$x_j^{(1)} = \phi\left(z_j^{(1)}\right) = \phi\left(\sum_{i=1}^{D} w_{i,j}^{(1)} x_i^{(0)} + b_j^{(1)}\right), \tag{1}$$

$$\hat{y} = \phi\left(z_1^{(2)}\right) = \phi\left(\sum_{i=1}^{K} w_{i,1}^{(2)} x_i^{(1)} + b_1^{(2)}\right). \tag{2}$$

Use Equation 1 and Equation 2 to fill in the corresponding template function in the notebook, and pass the test. For simplicity, in the following questions, let the bias term be 0 and use the `Sigmoid` as the activation function $\phi(\cdot)$.

Problem 2 (Backpropagation in neural network):

Assume that we use the squared error as our loss function, as shown in Equation 3:

$$\mathcal{L} = \frac{1}{2}(\hat{y} - y)^2, \tag{3}$$

where we have only one sample in our case, and y is the true value while $\hat{y}$ is the network prediction.

Evaluate the derivative of $\mathcal{L}(\boldsymbol{w})$ with respect to weights $w_{i,1}^{(2)}$ and $w_{i,j}^{(1)}$, and implement the corresponding function in the notebook.

Problem 3 (Effect of regularization):

What is the effect of regularization on the weights? To get some insight, let $\boldsymbol{\Theta}$ be the vector of all weights in the neural network. Recall that we do not penalize the bias terms. Therefore, let us ignore them in the following. Let $\boldsymbol{\Theta}^\star$ be a parameter that minimizes the cost function $\mathcal{L}$ for the given test set (where the cost function does not include the regularization). We would like to study how the optimal weight changes if we include some regularization.

In order to make the problem tractable, assume that $\mathcal{L}(\boldsymbol{\Theta})$ can be locally expanded around the optimal parameter $\boldsymbol{\Theta}^\star$ in the form

$$\mathcal{L}(\boldsymbol{\Theta}) = \mathcal{L}(\boldsymbol{\Theta}^\star) + \frac{1}{2}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^\star)^\top \boldsymbol{H}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^\star),$$

where $\boldsymbol{H}$ is the Hessian whose components are the entries

$$\frac{\partial^2 \mathcal{L}}{\partial \Theta_i \partial \Theta_j}.$$

Now add a regularization term of the form $\frac{1}{2}\mu|\boldsymbol{\Theta}|_2^2$.

1. Show that the optimum weight vector for the regularized problem is given by

   $$\boldsymbol{Q}(\boldsymbol{\Lambda} + \mu \boldsymbol{I})^{-1}\boldsymbol{\Lambda}\boldsymbol{Q}^\top \boldsymbol{\Theta}^\star,$$

   where $\boldsymbol{H} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^\top$ is the SVD of the symmetric matrix $\boldsymbol{H}$, $\boldsymbol{Q}$ is an orthonormal matrix, and $\boldsymbol{\Lambda}$ is a diagonal matrix whose entries are non-negative and decreasing along the diagonal.

2. Show that $(\boldsymbol{\Lambda} + \mu \boldsymbol{I})^{-1}\boldsymbol{\Lambda}$ is again a diagonal matrix whose i-th entry is now $\lambda_i/(\lambda_i + \mu)$.

3. Argue that along the dimensions of the eigenvectors of $\boldsymbol{H}$ that correspond to large eigenvalues λ_i essentially no changes occur in the weight, but that along the dimensions of eigenvectors of very small eigenvalues the weight is drastically decreased.