University of Stavanger

Stavanger, November 24, 2022

Theoretical exercise 4

ELE520 Machine learning

Problem 1

Show that if the transfer function of the hidden units is linear, a two-layer network is equivalent to a one-layer network. (Hint: Express the forward compuatations and from input to output and see what happens when the nonlinear functions are replaced with linear functions like in the input layer. Use matrix notationin with input vector $\boldsymbol{x}$, layer outputs $\boldsymbol{y}^r$ and weight matrices $\boldsymbol{\Theta}^r, r = 1, 2.$)

Problem 2

We want to design a three layer neural net to discriminate between the characters A, P, C, F which corresponds to the classes $\omega_1, \omega_2, \omega_3, \omega_4$ respectively. The characters are characterised by the number of corners, holes and symmetrical properties.

The training vectors we shall use is givenb as $\boldsymbol{x}_1 = (4\ 1\ 1)^T$, $\boldsymbol{x}_2 = (4\ 1\ 0)^T$, $\boldsymbol{x}_3 = (2\ 0\ 1)^T$ and $\boldsymbol{x}_4 = (2\ 0\ 0)^T$ where $\boldsymbol{x}_i$ are the training vector for class ω_i.

The corresponding target vectors are $y_1 = (1\ 0\ 0\ 0)^T$, $y_2 = (0\ 1\ 0\ 0)^T$, $y_3 = (0\ 0\ 1\ 0)^T$ and $y_4 = (0\ 0\ 0\ 1)^T$.

The net is configured with number of inputs, hidden nodes and outputs according to $l = 3$, $n_H = 3$ and $M = 4$. In addition we will use a bias for each layer. The net is initialised with arbitrary values for the weights. An unlinear sigmmoid is used which is of the type $f(z^r) = \frac{1}{1+\exp^{-z^r}}$.

As you will see, this network introduces bias units. You will need to acommodate the learning rules to this.[1]

a) Draw the network when the arbitrary first layer weights is given by $\boldsymbol{\theta}_1^1 = (0.0\ \ -0.5\ 0.5)^T$, $\boldsymbol{\theta}_2^1 = (0.5\ \ -0.5\ 0.0)^T$, $\boldsymbol{\theta}_3^1 = (-0.5\ 0.0\ 0.5)^T$ where $\boldsymbol{\theta}_j^1$ are the weights connected to the hidden node number j. The bias weights for the hidden nodes are 0.5, -0.5 and 0.5.

[1] The learning rules presented in the textbook includes bias, but is a bit more involved as it is multilayered and uses indexing to a greater extent than what has been presented in the lectures.

Trygve Eftestøl, Professor
Faculty of Science and Technology
Department of Electrical and Computer Engineering
Kjølv Egelands hus
University of Stavanger
N-4036 Stavanger.
Telephone: +47 51 83 10 00.
Telefax: +47 51 83 17 50.
E-mail: trygve.eftestol@uis.no
www.ux.his.no/~trygve-e

Correspondingly, the second layer weights are given by $\boldsymbol{\theta}_1^2 = (0.5\ -0.5\ 0.5)^T$, $\boldsymbol{\theta}_2^2 = (0.0\ -0.5\ 0.5)^T$, $\boldsymbol{\theta}_3^2 = (-0.5\ 0.5\ 0.0)^T$ and $\boldsymbol{\theta}_4^2 = (0.5\ 0.0\ -0.5)^T$ where $\boldsymbol{\theta}_k^2$ are the weights connected to output node number k. The bias weights to the output nodes are -0.5 0.5, -0.5 and 0.5.

b) Normalise the feature vectors in the training set according to $\boldsymbol{x} = \boldsymbol{x}/4$, so that $0 <= x_i <= 1$.

c) Do one *forward computation* and commpute $J(\boldsymbol{\theta})$ for the normalised training vector $\boldsymbol{x}_1$.

d) Do one *backward computation* with learning rate $\mu = 1$ based on the result from the previous subtask to update the weights in the net.