Cognitive Algorithms - Exercise Sheet 6

Mulitlayer Perceptron (MLP)

Department of Machine Learning - TU Berlin

Task 1 - Linear activation function [2 Points]

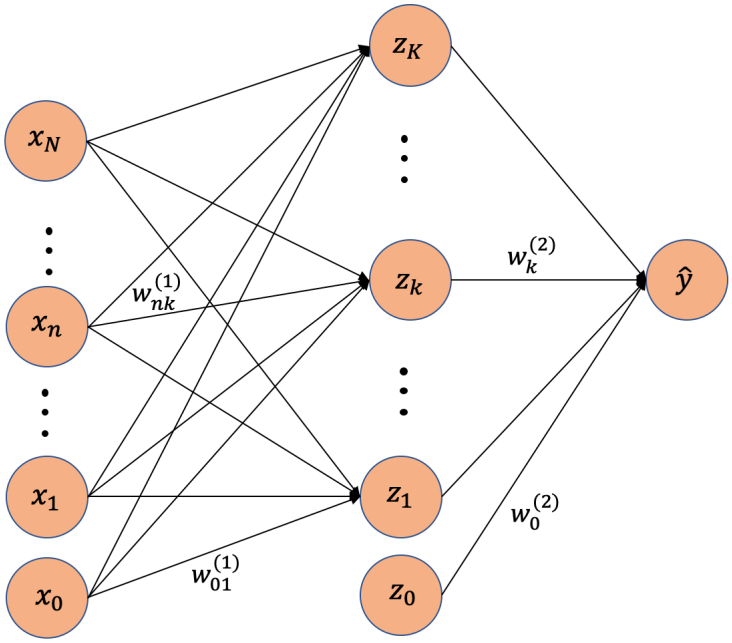

Figure 1: MLP with a single output neuron

Consider an MLP with an input layer with N input neurons, one hidden layer with K hidden neurons and an output layer with a single output neuron as depicted in Fig. 1. The input of the n^{th} input neuron is denoted by x_n which is equal to the output of the same input neuron. The input and output of the k^{th} hidden neuron are denoted by a_k and z_k, respectively, while the output of the single output neuron is denoted by $\hat{y}$. $w_{nk}^{(1)}$ defines the weight connecting the n^{th} input neuron to the k^{th} hidden neuron, and $w_k^{(2)}$ the weight connecting the k^{th} hidden neuron to the output neuron. $w_{0k}^{(1)}$ and $w_0^{(2)}$ are the biases in the first and second layer, respectively. x_0 and z_0 are bias nodes, do not receive any input, and are always $x_0 = z_0 = 1$. In this task all neurons have a linear activation function, thus the output of a hidden neuron can be written as

$$z_k = w_{0k}^{(1)} + \sum_{n=1}^{N} w_{nk}^{(1)} x_n \tag{1}$$

and the total output becomes

$$\hat{y} = w_0^{(2)} + \sum_{k=1}^{K} w_k^{(2)} z_k \tag{2}$$

Show that there exists an MLP without hidden layers, which models the same function.

Task 2 - Forward propagation [2 Points]

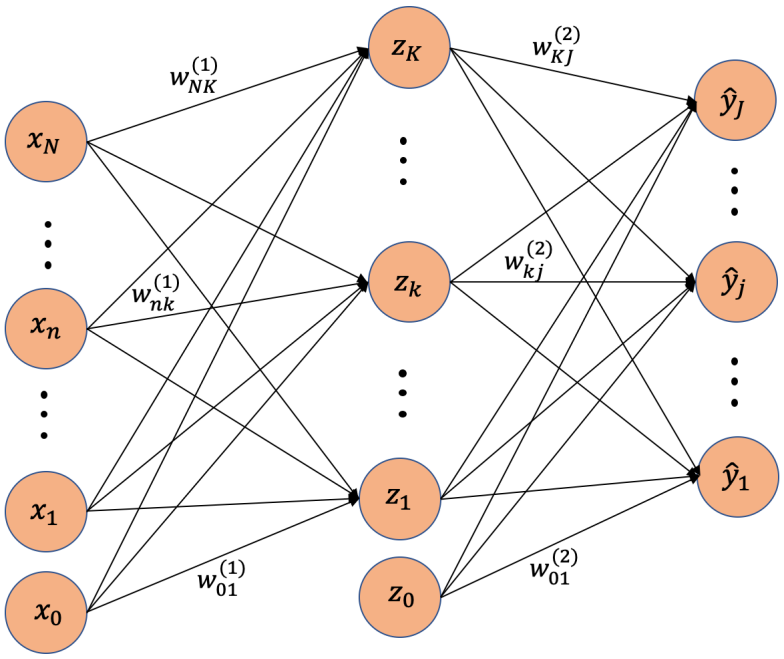

Figure 2: MLP with multiple output neurons

Consider a three-layered network (one input, one hidden, one output layer) as illustrated in Fig. 2 together with a sum-of-squares error, in which the output units have linear activation functions, so that $\hat{y}_j = a_j$, while the hidden units have sigmoidal activation functions given by

$$h(a) = \tanh(a) \tag{3}$$

$$\text{with} \qquad \tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}} \tag{4}$$

$$\text{and} \qquad h'(a) = 1 - h(a)^2. \tag{5}$$

We also consider a standard sum-of-squares error function, so that for data point n the error is given by

$$E_n = \frac{1}{2} \sum_{j=1}^{J} (\hat{y}_j - y_j)^2. \tag{6}$$

Note that $\hat{y}_j$ denotes the output of the j$^{\text{th}}$ output neuron (e.g. a predicted label in case of classification) whereas y is the corresponding label for a particular input x_n, i.e. the "true label" or also sometimes referred to as "ground truth". The output layer starts at index $j = 1$, as it does not contain a bias for the subsequent layer.

Compute forward propagation, i.e. compute the activation a_k of the k^{th} neuron in the hidden layer and the output of the output layer neurons $\hat{y}_j$.

Hint: Remember that the activation of a neuron refers to the input that its corresponding activation function receives.

(a) $a_k =$

(b) $\hat{y}_j =$

Task 3 - Backpropagation [2 Points]

Consider the model in Fig. 2. To perfom backpropagation, we need to compute the derivative of the error function w.r.t. the corresponding weights. Compute the derivative w.r.t. the first and second-layer weights.

Hint: Use the chain rule.

(a) $\dfrac{\partial E_n}{\partial w_{kj}^{(2)}} =$

(b) $\dfrac{\partial E_n}{\partial w_{nk}^{(1)}} =$