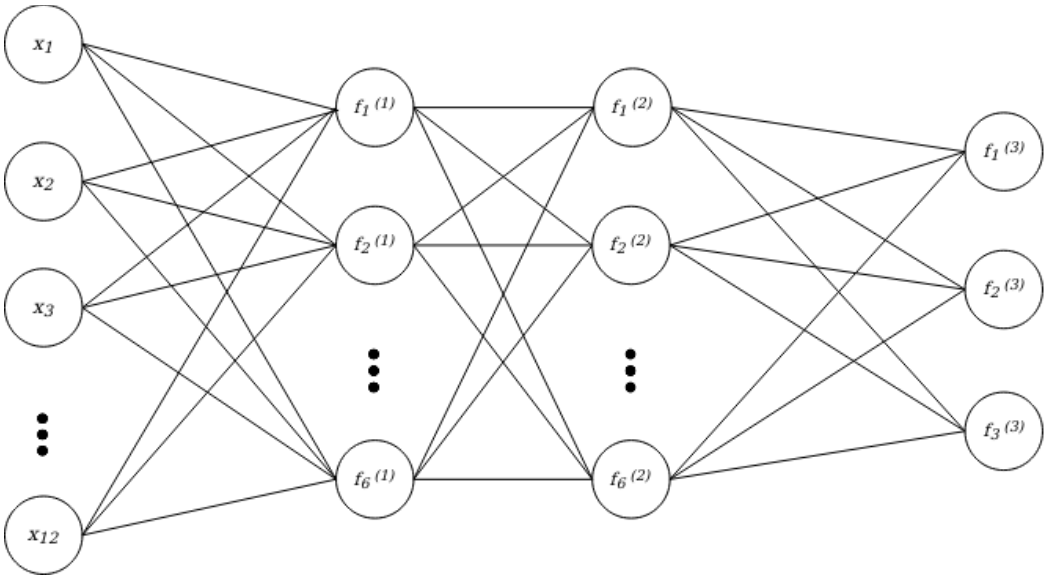
Assignment 8
Neural Networks

Kuang Yu Li, st169971@stud.uni-stuttgart.de, 3440829
Ya Jen Hsu, st169013@stud.uni-stuttgart.de, 3449448
Gabriella Ilena, st169935@stud.uni-stuttgart.de, 3440942

1 Formalizing Neural Networks

We define a neural network consisting of an input layer with 12 units, two hidden layers with 6 units each, and an output layer with 3 units corresponding to the probabilities of possible class predictions. The *tanh* activation function is used for the hidden layers and the *softmax* activation function is chosen for classification at the output layer.



Each layer can be defined as the following functions:

$$f^{(0)}(x) = x \in \mathbb{R}^{12\times 1}$$
$$f^{(1)}(\boldsymbol{x}) = g^{(1)}\left(\boldsymbol{W}^{(1)T}\boldsymbol{x} + \boldsymbol{b}^{(1)}\right)$$
$$f^{(2)}(\boldsymbol{x}) = g^{(2)}\left(\boldsymbol{W}^{(2)T}\boldsymbol{f}^{(1)}(\mathbf{x}) + \boldsymbol{b}^{(2)}\right)$$
$$f^{(3)}(\boldsymbol{x}) = g^{(3)}\left(\boldsymbol{W}^{(3)T}\boldsymbol{f}^{(2)}(\mathbf{x}) + \boldsymbol{b}^{(3)}\right)$$

where:

$$W^{(i)} : weight\ matrix\ for\ the\ i-th\ layer$$

$$W^{(1)} \in \mathbb{R}^{12\times 6},\ W^{(2)} \in \mathbb{R}^{6\times 6},\ W^{(3)} \in \mathbb{R}^{6\times 3}$$

$$b^{(i)} : bias\ vector\ for\ the\ i-th\ layer$$

$$b^{(1)} \in \mathbb{R}^{6\times 1},\ b^{(2)} \in \mathbb{R}^{6\times 1},\ b^{(3)} \in \mathbb{R}^{3\times 1}$$

$$g^{(i)}(z) : activation\ function\ for\ the\ i-th\ layer\ and\ argument\ z$$

$$g^{(1)}(z) = g^{(2)}(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

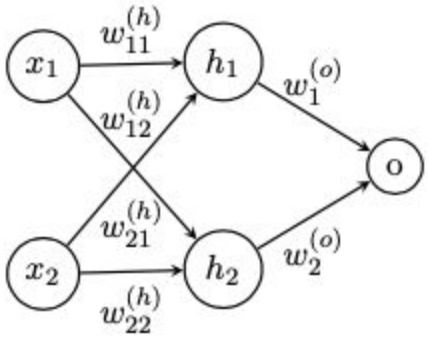
$$g^{(3)}(z) = \frac{e^{z_i}}{\sum_{j=1}^{3} e^{z_j}}$$

In this architecture, the prediction is given at the output layer $f^{(3)}$.

For the loss function, we may choose the cross-entropy loss function to calculate the loss between the predicted value (the output at the softmax layer) and the target value. We also introduce a weight factor α to penalize misclassification of the class -1 more heavily than the other classes. Assuming we have a batch size of 32, and that the true labels are one-hot encoded, then

$$CE_{loss} = \frac{1}{32}\sum_{n=1}^{32} \frac{-(1+\alpha)\log e^{f_1{}^{(3)}} - \log e^{f_2{}^{(3)}} - \log e^{f_3{}^{(3)}}}{\sum_{j=1}^{3} e^{f_j{}^{(3)}}}$$

2 Backpropagation by Hand



1. Compute the forward-pass for the input x = (2, −0.5).

$$\begin{bmatrix} h1 \\ h2 \end{bmatrix} = \max\left(\begin{bmatrix} w11 & w21 \\ w12 & w22 \end{bmatrix}\begin{bmatrix} x1 \\ x2 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = \max\left(\begin{bmatrix} 0.5 & 0.6 \\ -0.2 & 0.5 \end{bmatrix}\begin{bmatrix} 2 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0.7 \\ 0 \end{bmatrix}$$

$$o = w1 * h1 + w2 * h2 = 1 * 0.7 - 1 * 0 = 0.7$$

2. Compute the squared error loss of the forward-pass for the true value y = 1.5.

$$J = L(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(1.5 - 0.7)^2 = 0.32$$

3. Given the computed error, adjust the weight w11 via back-propagation with gradient descent using a learning rate of 0.1.

$$\frac{\mathrm{d}J}{\mathrm{d}h_1} = \frac{\mathrm{d}o}{\mathrm{d}h_1}\frac{\mathrm{d}J}{\mathrm{d}o} = w_1^{(o)}\left(\frac{1}{2}2(y-o)(-1)\right) = 1(1.5-0.7)(-1) = -0.8$$

$$\frac{\mathrm{d}J}{\mathrm{d}h_2} = \frac{\mathrm{d}o}{\mathrm{d}h_2}\frac{\mathrm{d}J}{\mathrm{d}o} = w_2^{(o)}\left(\frac{1}{2}2(y-o)(-1)\right) = -1(1.5-0.7)(-1) = 0.8$$

$$let\, z_1 = w_{11}^{(h)}x_1 + w_{21}^{(h)}x_2$$

$$h_1 = ReLU(z_1),\ ReLU'(z) = \begin{cases} 1 & z > 0 \\ 0 & z < 0 \end{cases}$$

$$\frac{\mathrm{d}J}{\mathrm{d}z_1} = \frac{\mathrm{d}h_1}{\mathrm{d}z_1}\frac{\mathrm{d}J}{\mathrm{d}h_1} = ReLU'(z_1)\frac{\mathrm{d}J}{\mathrm{d}h_1} = 1(-0.8) = -0.8$$

$$\frac{\mathrm{d}z_1}{\mathrm{d}w_{11}^{(h)}} = x_1 = 2$$

$$\frac{\mathrm{d}J}{\mathrm{d}w_{11}^{(h)}} = \frac{\mathrm{d}z_1}{\mathrm{d}w_{11}^{(h)}}\frac{\mathrm{d}h_1}{\mathrm{d}z_1}\frac{\mathrm{d}J}{\mathrm{d}h_1} = -0.8 * 2 = -1.6$$

$$w_{11}^{(h)}, new = w_{11}^{(h)} - \eta\frac{\mathrm{d}J}{\mathrm{d}w_{11}^{(h)}} = 0.5 - 0.1 * (-1.6) = 0.66$$