FNLP Tutorial 2

1 The Softmax Function

The softmax function takes an arbitrary vector $\mathbf{v}$ as input, with $|\mathbf{v}|$ dimensions. It computes an output vector, also of $|\mathbf{v}|$ dimensions, whose ith element is given by:

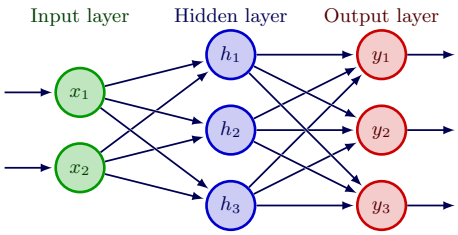
$$\text{softmax}(\mathbf{v})_i = \frac{\exp(\mathbf{v}_i)}{\sum_{j=1}^{|\mathbf{v}|} \exp(\mathbf{v}_j)}$$

1. What is the purpose of the softmax function?
2. What is the purpose of the expression in the numerator?
3. What is the purpose of the expression in the denominator?

2 Feed-forward neural networks

Consider a two-layer neural network with the topology visualised below, with the corresponding weights and bias values in the table. The hidden layer is followed by a non-linear function: the ReLU. The output layer is followed by a non-linear function too: the softmax. Read up on those functions and how to work with feedforward neural networks in sections 7.1 to 7.3 from J&M (only available in the 3rd edition!).

The network can be used for simple classification for three output classes. An input (consisting of features x_1 and x_2) belongs to one of the three classes, and you will classify an example input.



$w_{x_1,h_1} = 0.23$ $b_{h_1} = 0.13$ $w_{h_1,y_1} = 0.02$ $b_{y_1} = 0.31$
$w_{x_1,h_2} = -2.90$ $b_{h_2} = 0.90$ $w_{h_1,y_2} = -0.13$ $b_{y_2} = -0.18$
$w_{x_1,h_3} = -0.45$ $b_{h_3} = 0.24$ $w_{h_1,y_3} = 0.47$ $b_{y_3} = -0.56$
$w_{x_2,h_1} = 1.10$ $w_{h_2,y_1} = -0.88$
$w_{x_2,h_2} = 0.05$ $w_{h_2,y_2} = 1.74$
$w_{x_2,h_3} = -0.01$ $w_{h_2,y_3} = -0.09$
$w_{h_3,y_1} = -0.55$
$w_{h_3,y_2} = 0.72$
$w_{h_3,y_3} = -2.22$

1. Compute the class an input with $x_1 = 1.50$, $x_2 = 3.11$ would belong to. Show the intermediate computations, not just the final class.
2. Now imagine that you want to perform classification, but one input can belong to multiple classes. For example, when classifying a sentence with an emotion, that sentence can capture both anger and despair. To enable multi-class classification in this network, what adaptation would you make to its structure or the non-linear functions it uses?
3. Going back to the original example with $x_1 = 1.50$, $x_2 = 3.11$, use back-propagation to compute the derivative of each parameter if the gold label for that example was $y = 1$ (corresponding with the first class). Assume we use cross-entropy as the loss function:

$$\mathbf{L}(x) = -\sum_i p_i(x) \log q_i(x)$$

Where $p_i(x)$ is the target probability of the i-th class for the gold labels (in this case, $p_1(x) = 1$, $p_2(x) = 0$, $p_3(x) = 0$) and $q_i(x)$ is the probability the network assigns to the i-th class.

4. What are the benefits of back-propagation?

5. Update the parameters of the first layer, following simple gradient descent. Assume a learning rate $\mu = 0.1$.