Prof. Dr. Andreas Hotho, Albin Zehe, Konstantin Kobs
Lehrstuhl für Data Science

4./5.5.2021

1. Assignment in "Machine Learning for Natural Language Processing"

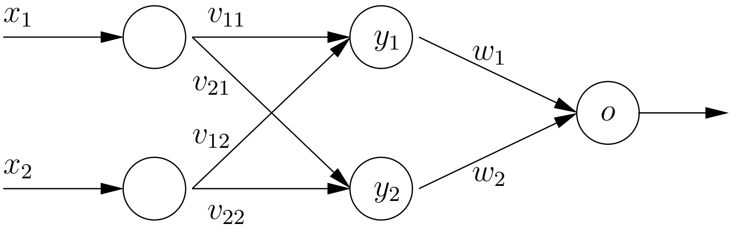
Summer Term 2021

1 General Questions

1. Identify three concrete cases where Deep Learning is used for language processing in real-world applications!
2. Name three common activation functions and their derivatives!

2 Neural Networks and Circuit Diagrams

Given the following neural network with `sigmoid` as activation function:



1. Represent the network as three equations (one each for y_1, y_2 and o)! We do not use any bias in this network.
2. Convert the network to a circuit diagram as the one shown in the lecture!
   Hint: Draw the diagram as large as possible. You will need space for the backpropagation.
3. To measure the quality of our weights, we will use the following loss function:

   $$L = \frac{1}{2} \cdot (t - o)^2,$$

   where o is the output of the network and t is the target (correct) label.

   Add the loss function to the circuit diagram from the last subtask!

4. Now that we have a circuit diagram, we can rather easily do backpropagation. Determine the partial derivatives for all *weights* ($v_{11}, v_{12}, v_{21}, v_{22}, w_1$ and w_2) in the network!

   Assume the following initial values: $v = \begin{pmatrix} 0.5 & 0.75 \\ 0.25 & 0.25 \end{pmatrix}$, $w = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix}$.

   The inputs are $(x_1, x_2) = (1, 1)$ and the target label is $t = 0$.

   Remember the intuitive properties of $+$ and $*$ from the slides:

   - $+$ just passes the gradient through, both inputs have the same partial derivative as the output
   - $*$ "switches" the input, that is: if the gradient after the $*$ is 2 and the inputs are 1 and 3, then first input gets the partial derivative $3 \cdot 2 = 6$ and the second input gets $1 \cdot 2 = 2$.

? Something to think about

5. Backpropagation would enable us to compute a gradient for the *input values* (x_1, x_2), too. What could we do with the information about the input's gradient?

3 Python

In the lecture, some commonly used optimisers for neural networks were introduced. Implement functions for `sgd_update`, `nesterov_momentum_update` and `adam_update` in Python! Each function should

- take as input
  - the current value of *one* input parameter,
  - a function returning the gradient regarding this parameter, and
  - the necessary hyper-parameters such as the learning rate, and
- return the new value for this parameter after the update.

You do *not* have to implement backpropagation in this assignment!