Benjamin Grewe, Matthew Cook, Giacomo Indiveri, Daniel Kiper, Wolfger von der Behrens, Valerio Mante
Lecture 13
Asst: Risi Nicoletta, Raphaela Kreiser, Ioana Calangiu {nicoletta, rakrei, calangiu}@ini.uzh.ch

Exercise 13.1: Error Back Propagation

Feed-forward neural networks consist of several layers of neurons. Let's denote the inputs to the first layer as $\mathbf{x}$ and the output(s) of the last layer as $\mathbf{y}$. Training such a network means finding a set of weights of the neurons, such that for a given input $\mathbf{x}$ the network answers with the desired output $\mathbf{y}$. However, we don't know exactly the functional dependency between $\mathbf{x}$ and $\mathbf{y}$ (and if we knew, we wouldn't need to build a neural network to learn it). Mostly, we are only given some samples

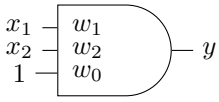
$$S = \{(\mathbf{x}, \mathbf{d})^1, \ldots, (\mathbf{x}, \mathbf{d})^l\} \tag{1}$$

of inputs ($\mathbf{x}^i$) and the corresponding desired output ($\mathbf{d}^i$). Training the network now means that we want it to get these samples right, or at least *as right as possible*. For that, we define an error function

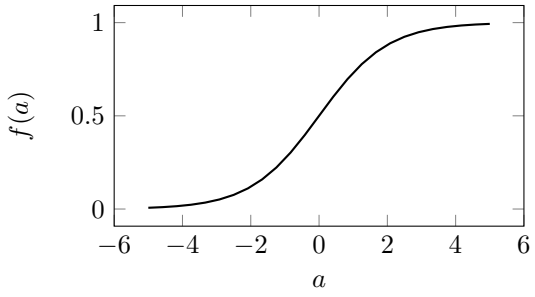
$$E(S) = \sum_i \frac{1}{2} \|\mathbf{y}(\mathbf{x}^i) - \mathbf{d}^i\|^2 \tag{2}$$

that tells us by how much we are off the desired output. Here $\mathbf{y}(\mathbf{x})$ denotes the output of the network given the input $\mathbf{x}$. The goal of the training is to tune the weights of the neurons such that the error function is minimized.

1. Let's consider the most simple neural network, consisting of only one unit with two inputs.



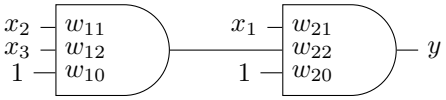
The weighted sum $a = x_1 w_1 + x_2 w_2 + w_0$ of the inputs is called the activation of that neuron. The output is a non-linear transformation $y = f(a)$ of the activation, where f, the activation function, is usually a sigmoid (we will encounter the sigmoid function also in a later exercise about the differential pair):



Answer the following questions:

- Given the above activation function and a single training sample $((x_1, x_2), (d))$, what is the error function?
- By looking at the error function, do you think that there is a closed form to compute the minimum for general activation functions? If yes, what is it? If no, what could we do instead to minimize the error?
- What is the derivative of the error function with respect to w_1 and w_2? You can assume that $f(a)$ is differentiable everywhere and write $f'(a)$ for $\frac{\partial f(a)}{\partial a}$. (Hint: You have to use the chain-rule.)
- What would change if we had more than just one training sample?

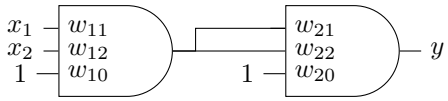
2. Now we increase the complexity of our network by replacing the former input x_2 with the output of another neuron having the same activation function $f(a)$.



Answer the following questions:

- What is the output $y(\mathbf{x})$ of the network?
- Given a single training sample $((x_1, x_2, x_3), (d))$, what is the error function?
- What is the derivative of the error function with respect to w_{22}?
- What is the derivative of the error function with respect to w_{11} and w_{12}?
- Look at your derivatives for w_{11} and w_{12}. Which terms of the computation of the derivative for w_{22} can be reused?

3. **(optional)** Finally, we want to see the changes if we reuse the output of one neuron. For that, we change our network as follows:



Using the same samples and ignoring the input values for x_1, answer the following questions:

- Given a sample $((x_1, x_2), (d))$, what is the error function?
- What is the derivative of the error function with respect to w_{11} and w_{12}? How are the terms of the computation of the gradients of w_{21} and w_{22} combined now?

Exercise 13.2: Hopfield Networks

A Hopfield net is a form of recurrent artificial neural network invented by John Hopfield in 1982. One example with seven units is represented in the figure. Let $\boldsymbol{W}$ denote the weight matrix (the matrix in which the connection strengths between the neurons are stored) of the Hopfield net, w_{ij} its elements.

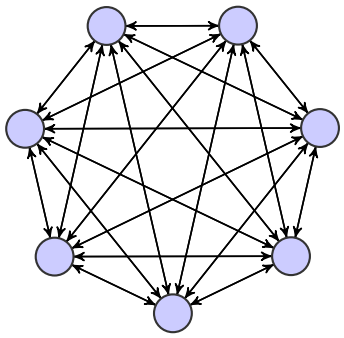


Figure 1: A Hopfield Network of 7 neurons. Vertices represent neurons, edges indicate connections.

1. Based on the figure, what are the symmetries and zero-entries of the weight matrix?

At each time step, each neuron computes an output based on its inputs. Since there is feedback, the activation does not stop after one step. But the state of the network (the value of all the units) changes in time. We are particularly interested in finding stable states, where the network does not change any more.

We now consider a Hopfield network of four neurons and the set of outputs {–1,1}. We want to learn 2 patterns: (–1,1,1,–1) and (–1,1,–1,1). That means we want that when neuron one outputs –1, neuron two outputs 1, neuron three outputs 1 and neuron four outputs –1, they keep these values at each step. Similarly for the second pattern. Assume zero threshold. Note that if inactivity of a unit is represented by –1, we also use –1 for the calculation of the weighted sum, not 0.

2. What conditions does the existence of these patterns impose on the weight matrix? (What should a given neuron do for inputs that correspond to these patterns?) Give a concrete example of weights w_{ij} that fulfill these conditions.

Hopfield nets serve as content-addressable memory systems with binary threshold units. In the retrieval phase an N-dimensional vector $\mathbf{x}$, called the probe, is imposed on the network as its state. Typically it represents a noisy version of a fundamental memory. Information retrieval is then done in accordance with the dynamical update rule. Consider a three unit Hopfield network with weight matrix:

$$W = \frac{1}{3} \cdot \begin{pmatrix} 0 & -2 & 2 \\ -2 & 0 & -2 \\ 2 & -2 & 0 \end{pmatrix}$$

3. Having three neurons how many different possible states can this network have?

4. Calculate for all possible states, what the following state will be. Assume asynchronous updates (*i.e.* one unit updates at a time), zero threshold (iff weighted inputs are ≥ 0 the unit is active) and outputs in $\{-1,1\}$. For illustration it is useful to draw this as a graph in which the possible states are nodes and the possible transitions directed edges.

5. Which of the states are stable? What are therefore the fundamental memories of the network (if there are less or more than two, you have an error in your calculation)?