Neural Networks 2018
Homework 3
MLP and Backpropagation

General instructions

Print your report and put it in the box of Neural Networks next to room 216 in Bernoulliborg. Also send a pdf version of the report by e-mail to neuralnets19+HW_3@gmail.com. The assignment is due on May 13^{th} at 13:00. You should e-mail one archive file containing at least your report, so the easiest way of doing this is just to archive all of your files at once. Name it as follows:
s1234567_DonaldTrump_s7654321_SantaClaus.zip.
The maximum length of the report for this homework is **10 pages**.

Additional Information

For this assignment you can receive 10 points. If you receive p points, your homework grade is given by grade $= p$, and it will be averaged with the lab assignment grade for this week.

1 Theory questions (2 pt.)

a) What is the *credit assignment problem*?

b) What are the three factors that determine the error of a neuron in a hidden layer (the local gradient of a neuron at that layer) after the forward pass has finished? How do they influence the local gradient, what do they mean?

c) Give the mathematical definition of the sigmoid function. Clearly state which symbols are constants and what the variable is. How do the constants of the function affect the shape?

d) Why is it impractical to initialize the weights with very high values?

e) Describe three criteria that you can use to determine when to stop learning. Provide one disadvantage per criterion.

f) How can you speed up the learning of a network besides increasing the learning rate? Explain.

g) How can you verify that a network is generalizing for some training data?

h) Describe the problem of *overfitting.*

i) What is network pruning?

j) Describe at least two ways of accomplishing network pruning.

2 An MLP on paper (2 pt.)

a) Draw a two-layer neural network with 2 input neurons, 3 hidden neurons and 2 output neurons. Enumerate the neurons from top to bottom.

b) Why is this considered a two-layer network instead of a 3-layered network?

input neuron	weights to first hidden neuron
1	0.3
2	0.6

Table 1: Connections from the two original input neurons to the first hidden neuron.

Let W^h and W^o be the matrices that represent the weights between the input and the hidden layer and the weights between the hidden and the output layer respectively.

In W^h the k-th row will correspond to the connections that go from the k-th input neuron to the hidden layer. For W^o the weights are represented similarly: the weights from the k-th row correspond to the connections that go from the k-th hidden neuron to the output layer.

c) What are the dimensions (number of rows and columns) of W^h and W^o?

d) How do the weights in a single column of a weight matrix relate to each other?

Let a_{ij} be the entry at row i and column j in an $m \times n$ matrix A. Note that matrix indexing starts at 1 (and not at 0).

e) Mark w^h_{12} and w^h_{21} in your drawing by giving the connections a color.

f) Extend the drawing of your network by creating an augmented input layer. What are the dimensions of W^h and W^o now?

g) All thresholds at the input are set to 0.5. Table 1 shows the connections between the first two input nodes and the first hidden node. Extend the table with a third row that represents the augmented part of the input. Where can we find the values of this table in W^h?

h) The input at the first input neuron is 1. At the second input neuron it is 0. Let $\vec{\boldsymbol{x}}$ be the row vector containing the inputs of the augmented network. Compute the activation of the first hidden neuron using $\vec{\boldsymbol{x}}$ and the values in Table 1.

i) Explain why we can compute the activation of all hidden neurons by performing the following vector-matrix multiplication:

$$\vec{\boldsymbol{a}}^h = \vec{\boldsymbol{x}} W^h$$

where $\vec{\boldsymbol{a}}^h$ is the activation vector where the k-th element corresponds to the k-th activation in the hidden layer.

j) Let $\vec{\boldsymbol{y}}^h$ be the output of the hidden layer. How do we compute the activation of the output layer $\vec{\boldsymbol{a}}^o$? Write down the multiplication in terms of one or more vectors and/or matrices. Transpose the terms if necessary.

3 Backpropagation on Paper (3pt)

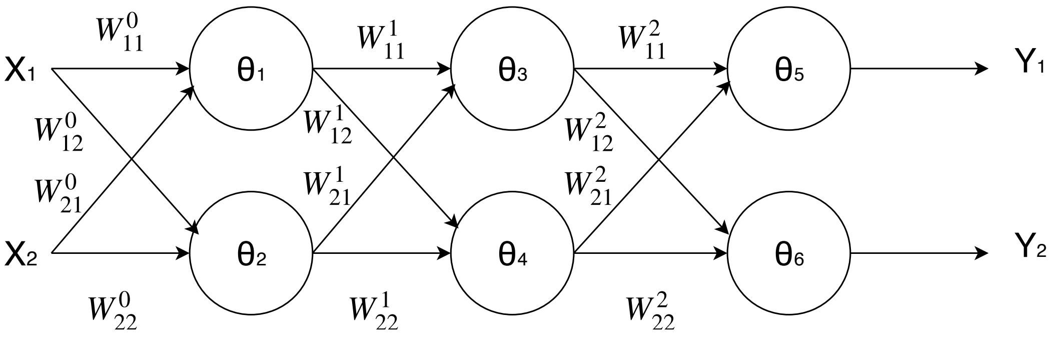

Figure 1: MLP with two hidden layers

In this assignment you are given a multi-layer perceptron (MLP), see Figure 1. Given this MLP and its parameters in Table 1-3, a learning rate of $\alpha = 0.1$, an input of $\mathbf{x} = [1, 1]$ and a target of $\mathbf{y} = [1, 1]$ apply backpropagation to the network. To do so first forwardpropagate the input through the network by calculating progressively the activations of all neurons and by calculating the error of the output neurons. Then you need to backpropagate through the network. Do so by clearly denoting and calculating the local gradients of each layer. If possible, report the local gradients and all weight changes that you get for each of the following activation functions. Denote the input layer as h, the hidden layers as i and j, and the output layer as k.:

W_{11}	W_{12}	W_{21}	W_{22}	θ_1	θ_2
-0.5	0	0.5	1	0	0

Table 2: Weight matrix for W^0

W_{11}	W_{12}	W_{21}	W_{22}	θ_3	θ_4
-0.5	0	0.5	1	0	0

Table 3:Weight matrix for W^1

W_{11}	W_{12}	W_{21}	W_{22}	θ_5	θ_6
-0.5	0	0.5	1	0	0

Table 4:Weight matrix for W^2

1. Linear activation function: $f(a) = a$
2. Sigmoid activation function: $\sigma(a) = \frac{1}{1+e^{-(a-\theta)}}$
3. Threshold activation function[1]: $f(a) = (a > \theta) \cdot 1$

4 Essay questions (3 pt.)

1. A problem with backpropagation is how computationally expensive it can become. A commonly used process to speed up training is *batch learning*, where the input data set is split into n-sized batches and the network is only trained once a batch has been processed. Explain how *batch learning* differs from *on-line learning* (training one input at a time). When would you use one over the other, and vice versa?

2. Adding more hidden layers leads to the *vanishing gradient problem.* Explain what this problem is and how the use of certain activation functions can reduce this problem.

3. Another notorious problem in neural networks is *overfitting.* If a network has too much freedom to choose its decision surface, it will start to learn noise and intricacies in the training data. One way to solve this is through the process of pruning. Explain how pruning helps reduce overfitting.

[1] Threshold activation funciton also expressed as: If $(a > \theta)$ then $f(a) = 1$, else $f(a) = 0$

4. Other than pruning, there are other methods that help avoid overfitting through methods that determine a suitable number of hidden nodes. Describe one method that determines it pre-training and one that determines it dynamically while training is ongoing. Explain why they help avoid overfitting.