Neural Networks - Homework 1

Thibault Desjonquères 19990602-8890

September 22, 2022

Code explanation

Purpose of the network.
This network determines whether an n-dimensional Boolean function is linearly separable or not, and if so, in how many ways. In a linearly separable function, there exist at least one vector separating inputs of different kinds (+1,-1 in this case). If such vector can be found, the problem can be solved using one neuron. In this network, the neuron's learning is supervised, meaning that at every iteration, the weights and thresholds are adjusted according to Hebb's rule. In this network, each dimension is represented by a neuron, which is updated by it's own weight. There is no interconnection between neurons of the same order.

The provided code is written in two parts. First, a set of targets is created. It is composed of every manner in which combinations of (+1,-1) can be arranged, in n-dimensions. For example, when n=2, this set contains $(2^{2^n}) = 16$ combinations.

Code description.
Second, the training of the neurons takes place. A Boolean is generated (shown with n=2 in 1). The code performs linear combinations of the weights and each X_j^μ represented in table 1. This linear combination is the temporary output, which is next, compared attractively to the 16 members of the training set. The target set is used as a training set. At each loop, the weights and thresholds are are adjusted, until the temporary output and the target have eventually become identical. In other words, at each iteration, the temporary output attempts to copy the target. If it manages to do so, it means the target can be used to separate the considered Boolean function linearly.

Results discussion
About the pre-generation of targets, it must be said that for a small dimension, it is very likely to find the 16 combinations of (-1,+1) within 10000 generation of random combinations. However, the bigger the dimension, the more iterations will be needed to find all different combinations (for n = 4,$(2^{2^n}) = 65536$ combinations). The best we can do is to generate an incomplete set of targets, as big as possible, and work with it.
About the amount of linearly separable Booleans functions, the greater the dimension, the harder it becomes to separate every input of the function. In summary, amongst the "limited" amount of targets we generated, it is very likely that none of them can be used to separate the function linearly. This idea is essential to understand the obtained results shown in Table (3). The code manages to find the correct amount of linearly separable function for n=2,3, but fails to do so for higher dimensions.

X1	**X2**
-1	-1
-1	1
1	-1
1	1

Table 1: Boolean in 2 variables

Results (14)			
-1	1	-1	1
1	-1	-1	-1
-1	-1	-1	-1
-1	1	-1	-1
1	-1	1	-1
1	1	-1	1
1	1	-1	-1
1	1	1	-1
-1	-1	1	1
1	-1	1	1
-1	-1	1	-1
-1	1	1	1
1	1	1	1
-1	-1	-1	1

Table 2: List of the 14 combinations which are linearly separable.

n-dimension	**Linearly separable Boolean functions (Code solutions)**	**Total number of combinations (2^{2^n})**	**Linearly separable Boolean functions (Tabulated values)**
2	14	16	14
3	104	256	104
4	252	65536	1882
5	None	4294967296	94572

Table 3: Number of combination making the corresponding boolean function linearly separable.