Homework 1 Boolean functions

To find all possible linearly separable cases for n-dimensional Boolean functions, one has to do sampling in the possible combinations of binary digits, especially in high-dimension cases.

When the trial starts, create a random Boolean output, a $2^n \times 1$ vector, containing only -1 or 1. Using this output with the learning rule during each epoch to iteratively update the weight and threshold. The output that has been generated before should be skipped so that the same Boolean function will not be counted twice. Also, a total error variable accumulates the differences between generated output and the one from McCulloch-Pitts dynamics. For the weight and threshold that has zero total error in one epoch, they can separate the n-dimension points and thus one linearly separable Boolean function is found.

n	Boolean functions 2^{2^n}	Linearly separable Boolean functions	My result
2	16	14	14
3	256	104	104
4	65536	1882	1304
5	4294967296	94572	2

Table 1: Result from Matlab experiment

For the n-dimensional Boolean function, the total number of possible combinations is 2^{2^n}. This gives the data in the second column.

The Matlab experiment always gives the correct result when the dimension $n = 2, 3$. In these two circumstances, the number of trials, 10^4, is much larger than the number of their possible Boolean functions, so that the trials can cover all the linearly separable functions.

For $n = 4$, the average amount is 1304 which is 30% smaller than the correct number. As the number of Boolean functions is now 6 times larger than trials, it is highly possible the program cannot find all the functions needed.

When the dimension goes up to $n = 5$, the final result mainly outputs 0 but may sometimes output a single digit like 2 or 3. It means the program hardly finds any accepted functions. This is because the number of trials is too small compared with 94572 possible functions in 10^9 possibilities.