CMPG-765/CMPT- 465 Neural Networks and Learning Systems

Homework-1

1. Design the **Hebb** function to calculate the weights for a given input/output mapping using the Hebb rule in Matlab using the m-language or any other language. The *Hebb function shall be robust*, so it shall not depend on the length of the input, the size of a learning set, and on a specific type of input (binary/real). Inputs and desired outputs shall be transferred to the function as its arguments. The Hebb functions shall return a weighting vector. (**undergrads - 50/100, graduates – 30/100**)
2. Using the Hebb function, which you designed, apply the Hebbian learning rule to input/output mappings described by all 16 Boolean functions of 2 variables. Test for each input/output mapping whether the weights obtained by the Hebbian rule implement it or not. Do not forget that Boolean functions must be represented in the {1,-1} alphabet, not {0,1}.
   (**undergrads - 40/100, graduates – 30/100**)
3. (**Required for graduate students and extra credit for undergrads**). Using the Hebb function, which you designed, apply the Hebbian learning rule to input/output mappings described by all 16 Boolean functions of 2 variables substituting their inputs (1, -1) → (0.5, -0.3) and (-1, 1) → (-0.5, 0.7), but keeping inputs (1, 1) and (-1, -1) and all initial function values. Test for each input/output mapping whether the weights obtained by the Hebbian rule implement it or not.
   (**undergrads – 30 extra credit points, graduates – 30/100**)

4. Write a brief report with your conclusions. A report should be strictly technical. It should mostly consist of a table of functions, which you tested< followed by the row (column) where it should be marked whether a Hebbian weighting vector implements this function. (**10/100**)
5. Turn in your source code, a screen shot of its test run and your report.

Input/output mappings in the traditional alphabet {0, 1}

y1	y2		AND					XOR	OR	NOR	NXOR					NAND	
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

The same input/output mappings in the alphabet {1,-1} suitable for neurons and neural networks

x1	x2		AND					XOR	OR	NOR	NXOR					NAND	
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	1	-1	1	-1	1	-1	1	-1	1	-1	1	-1	1	-1	1	-1