ECE/CS 559 - Fall 2020
Homework #1
Due: 09/30/2020, 11:00pm.

Erdem Koyuncu

- You are allowed to discuss the homework problems with your classmates, but you are supposed to do your assignment individually.
- You cannot use an existing machine learning / neural networking / etc. library.
- **READ VERY CAREFULLY:** You need to submit a hard copy of your codes to Gradescope. The same code should be uploaded to Box (I shared you a separate link for code repository where you can upload) in the format XX-YYYYYYYYY-LastName.py, where XX is the homework number, and YYYYYYYYY is your UIC net ID. The extension should be chosen appropriately according to the programming language. For example, for Homework 1, if your UIC net ID is 12345678, your name is James Bond, and you are writing in C++, you should name the file 01-12345678-Bond.cpp. Your single code file should be able to compile and execute without any errors. Submissions to Gradescope and Box should both be before the deadline. Failure to follow this procedure or submitting a non-functional code will be interpreted as failure of the programming assignment.

1. **(30 pts)** Design a two-layer neural network with the **signum activation function** (i.e. $\text{sgn}(x) = 1$ if $x > 0$, $\text{sgn}(x) = -1$ if $x < 0$, and $\text{sgn}(0) = 0$) such that the network implements the logic gate $f(x_1, x_2, x_3) = \overline{x_1}x_2x_3 + x_1\overline{x_2}$. Assume that the input of -1 is used to represent a `FALSE`, and an input of 1 is used to represent a `TRUE`. Show your work and draw the final network. Note that in class, we have discussed examples where we have instead used the step activation function and a 0 for `FALSE`.

2. **(30 pts)** Consider the network in Fig. 1. In the $x - y$ plane, sketch the region where $z = 1$. Show your work. Make sure you correctly indicate which part of the boundaries belong to the region $z = 1$. Recall that $u(x) = 1$ if $x \geq 0$ and $u(x) = 0$ if $x < 0$.

3. **(40 pts)** Write a computer program that runs the perceptron training algorithm with the step activation function $u(\cdot)$. **You have to include the source files of your computer program together with the solution of the problem**. Implement the following steps.

   (a) Do not be scared at the fact that the question has a million steps. Most of the steps are very simple and they are just there to make your life easier.

   (b) Pick (your code should pick it) w_0 uniformly at random on $[-\frac{1}{4}, \frac{1}{4}]$.

   (c) Pick w_1 uniformly at random on $[-1, 1]$.

   (d) Pick w_2 uniformly at random on $[-1, 1]$.

   (e) Write in your report the numbers $[w_0, w_1, w_2]$ you have picked.

   (f) Pick $n = 100$ vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ independently and uniformly at random on $[-1, 1]^2$, call the collection of these vectors $\mathcal{S}$.

   (g) Let $\mathcal{S}_1 \subset \mathcal{S}$ denote the collection of all $\mathbf{x} = [x_1\ x_2] \in \mathcal{S}$ satisfying $[1\ x_1\ x_2][w_0\ w_1,\ w_2]^T \geq 0$.

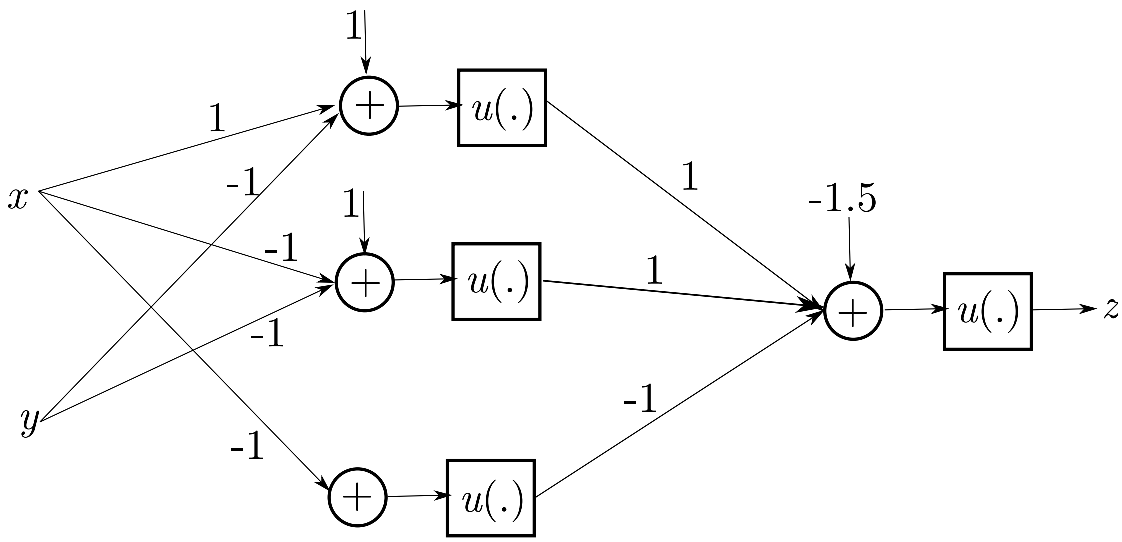

Figure 1: The neural network for Problem 2.

(h) Let $\mathcal{S}_0 \subset \mathcal{S}$ denote the collection of all $\mathbf{x} = [x_1\ x_2] \in \mathcal{S}$ satisfying $[1\ x_1\ x_2][w_0\ w_1,\ w_2]^T < 0$.

(i) In one plot, show the line $w_0 + w_1x_1 + w_2x_2 = 0$, with x_1 being the "x-axis" and x_2 being the "y-axis." In the same plot, show all the points in $\mathcal{S}_1$ and all the points in $\mathcal{S}_0$. Use different symbols for $\mathcal{S}_0$ and $\mathcal{S}_1$. Indicate which points belong to which class. An example figure may be as shown in Fig. 2 (My labels look bad, I expect you to do a better job!).

(j) Use the perceptron training algorithm to find the weights that can separate the two classes $\mathcal{S}_0$ and $\mathcal{S}_1$ (Obviously you already know such weights, they are w_0, w_1 and w_2 above, but we will find the weights from scratch, and the training sets $\mathcal{S}_0$ and $\mathcal{S}_1$). In detail,

   i. Use the training parameter $\eta = 1$.
   ii. Pick w_0', w_1', w_2' independently and uniformly at random on $[-1, 1]$. Write them in your report.
   iii. Record the number of misclassifications if we use the weights $[w_0', w_1', w_2']$.
   iv. After one epoch of the perceptron training algorithm, you will find a new set of weights $[w_0'', w_1'', w_2'']$.
   v. Record the number of misclassifications if we use the weights $[w_0'', w_1'', w_2'']$.
   vi. Do another epoch of the perceptron training algorithm, find a new set of weights, record the number of misclassifications, and so on, until convergence.
   vii. Write down the final weights you obtain in your report. How does these weights compare to the "optimal" weights $[w_0, w_1, w_2]$?

(k) Regarding the previous step, draw a graph that shows the epoch number vs the number of misclassifications.

(l) Repeat the same experiment with $\eta = 10$. Do not change $w_0, w_1, w_2, \mathcal{S}, w_0', w_1', w_2'$. As in the case $\eta = 1$, draw a graph that shows the epoch number vs the number of misclassifications.

(m) Repeat the same experiment with $\eta = 0.1$. Do not change $w_0, w_1, w_2, \mathcal{S}, w_0', w_1', w_2'$. As in the case $\eta = 1$, draw a graph that shows the epoch number vs the number of misclassifications.

(n) Comment on how the changes in η effect the number of epochs needed until convergence.

(o) Comment on whether we would get the exact same results (in terms of the effects of η on training performance) if we had started with different $w_0, w_1, w_2, \mathcal{S}, w_0', w_1', w_2'$.

(p) Do the same experiments with $n = 1000$ samples. Comment on the differences compared to $n = 100$.

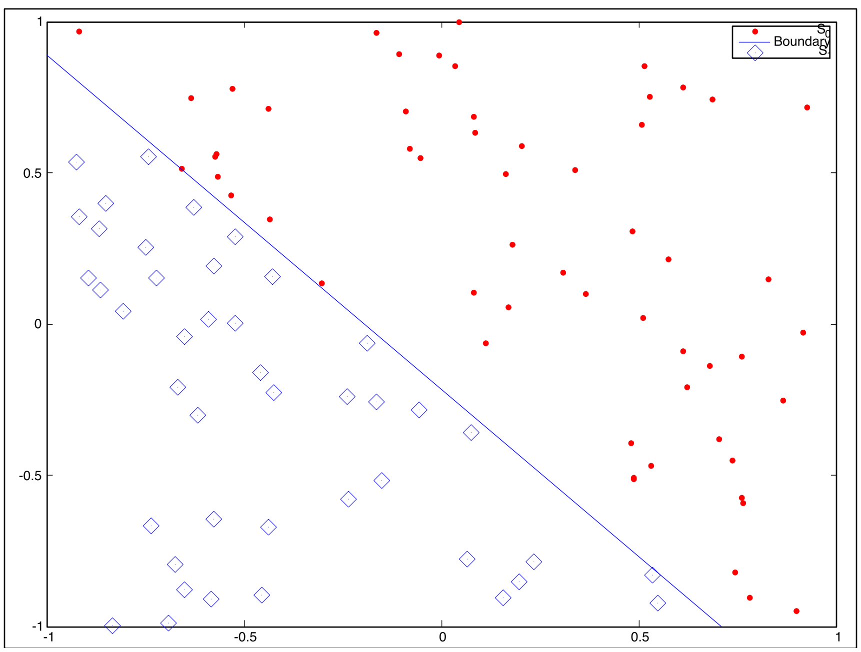

Figure 2: An example figure for Problem 3i.