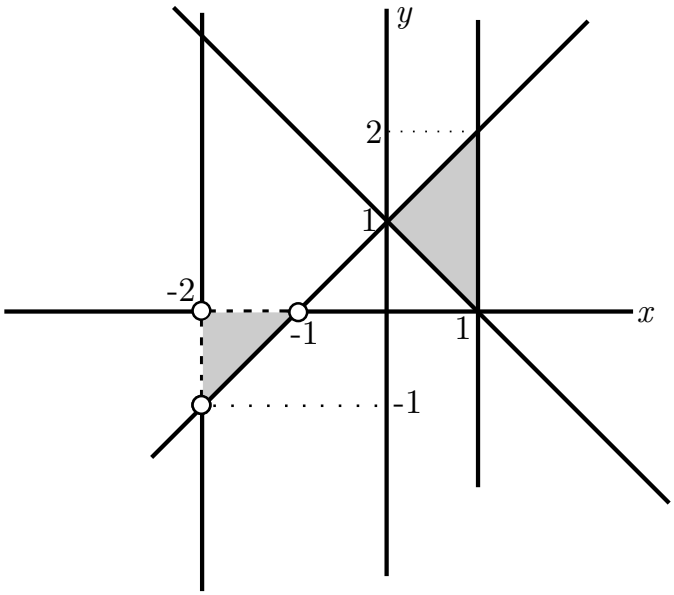
ECE/CS 559 - Fall 2019 - Homework #2
Due: 09/26/2019, the end of class.

Erdem Koyuncu

Note: All notes in the beginning of Homework #1 apply.

1. **(30 pts)** A neural network with the step activation function $u(\cdot)$ (i.e., $u(x) = 1$ if $x \geq 0$ and $u(x) = 0$ if $x < 0$) and inputs (x, y) provides an output of 1 if (x, y) is a member of the shaded region. The network provides an output of 0 if (x, y) is not a member of the shaded region. Find the neural network (its topology and weights). Note that for the leftmost shaded region, as I have tried to show in the figure, the boundary points $\{(x, 0) : -2 \leq x \leq -1\} \cup \{(-2, y) : -1 \leq y \leq 0\}$ are excluded.



2. **(70 pts)** In this computer experiment, we will use the multicategory PTA for digit classification. Multicategory PTA is a simple extension of PTA and will be described in the following.

   (a) Read the contents of the webpage `http://yann.lecun.com/exdb/mnist/`

   (b) There are 4 files listed in the beginning of the page as `training set images`, `training set labels`, `test set images`, and `test set labels`, download them.

   (c) Each image is 28×28, so that we will have a neural network $28 \times 28 = 784$ nodes in the input layer, and 10 nodes in the output layer. We will ignore the biases. We wish to find $784 \times 10 = 7840$ weights such that the network outputs $[1\ 0\ 0\ \cdots\ 0]^T$ if the input image corresponds to a 0, $[0\ 1\ 0\ \cdots\ 0]^T$ if the input image corresponds to a 1, and so on.

   (d) You will use the first n ($n \leq 60000$) elements of `training set images` and `training set labels` to train our network via the multicategory perceptron training algorithm. Since the patterns are not linearly separable, the misclassification errors may not converge to 0 (unlike the last experiment in HW#1). You need to stop the iterations (epochs) when the ratio of misclassified input patterns falls below some threshold ϵ. The algorithm for this phase may thus be as follows:

   - **0)** Given η, ϵ, n:
   - **1)** Initialize $\mathbf{W} \in \mathbb{R}^{10 \times 784}$ randomly.

- **2)** Initialize `epoch` $= 0$.
- **3)** Initialize `errors`(`epoch`) $= 0$, for `epoch` $= 0, 1, \ldots$.
- **3.1)** Do
  - **3.1.1)** for $i = 1$ to n do (this loop is where we count the misclassification errors)
    - **3.1.1.1)** Calculate the induced local fields with the current training sample and weights: $\mathbf{v} = \mathbf{W}\mathbf{x}_i$, where $\mathbf{x}_i \in \mathbb{R}^{784\times 1}$ is the ith training sample (the vectorized version of the 28×28 image from the `training set images`).
    - **3.1.1.2)** The given input image may result in multiple 1s on different output neurons (or no 1s at all). For such scenarios, only for the purpose of calculating the misclassification errors, we choose the output neuron with the largest induced local field. In other words, we find the largest component of $\mathbf{v} = [v_0\ v_1\ \cdots\ v_9]^T$. Now, suppose that the largest component of $\mathbf{v}$ is v_j, where $j \in \{0, \ldots, 9\}$. Correspondingly, our network decides that the input image $\mathbf{x}_i$ corresponds to the digit j.
    - **3.1.1.3)** If j is not the same as the input label (which is obtained from the `training set labels`), then `errors`(`epoch`) $\leftarrow$ `errors`(`epoch`) $+ 1$.
  - **3.1.2)** `epoch` $\leftarrow$ `epoch` $+ 1$.
  - **3.1.3)** for $i = 1$ to n do (this loop is where we update the weights)
    - **3.1.3.1)** $\mathbf{W} \leftarrow \mathbf{W} + \eta(\mathbf{d}(\mathbf{x}_i) - u(\mathbf{W}\mathbf{x}_i))\mathbf{x}_i^T$, where the step function $u(\cdot)$ is applied component-wise, and $\mathbf{d}(\mathbf{x}_i) \in \mathbb{R}^{10\times 1}$ is the desired output for training sample $\mathbf{x}_i$ (which is obtained from the `training set labels`). For example, if the label for $\mathbf{x}_i$ is 3, then $\mathbf{d}(\mathbf{x}_i) = [0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0]^T$.
- **3.2)** Loop to **3.1)** if `errors`(`epoch` $- 1)/n > \epsilon$.

(e) We now have some (hopefully) good weights that we have obtained via the multicategory PTA above. We now test the corresponding network on the `test set images` and `test set labels`. All we have to do is to use the loop **3.1.1)** in the training algorithm:
- **0)** Given $\mathbf{W}$ obtained from the multicategory PTA.
- **1)** Initialize `errors` $= 0$.
- **2)** for $i = 1$ to 10000 (note that there are 10000 test images)
  - **2.1)** Calculate the induced local fields with the current test sample and weights: $\mathbf{v}' = \mathbf{W}\mathbf{x}_i'$, where $\mathbf{x}_i' \in \mathbb{R}^{784\times 1}$ is the ith test sample (the vectorized version of the 28×28 image from the `test set images`).
  - **2.2)** Find the largest component of $\mathbf{v}' = [v_0'\ v_1'\ \cdots\ v_9']^T$. Suppose that the largest component of $\mathbf{v}'$ is $v_{j'}$, where $j' \in \{0, \ldots, 9\}$.
  - **2.3)** If j' is not the same as the input label (which is obtained from the `test set labels`), then `errors` $\leftarrow$ `errors` $+ 1$.

(f) Run Steps (d) and (e) for $n = 50$, $\eta = 1$, and some very small ϵ ($\epsilon = 0$ should also work). You should observe that step (d) terminates with 0 errors eventually. So, we have 0% error according to our training samples. Plot the epoch number vs. the number of misclassification errors (including epoch 0). Now, run Step (e) and record the percentage of misclassified test samples (over all 10000 test samples). Explain the discrepancy (if they are different why? if they are the same why?) between the percentages of errors obtained through the training and test samples.

(g) Run Steps (d) and (e) for $n = 1000$, $\eta = 1$, and some very small ϵ ($\epsilon = 0$ should also work). Again, you should observe that step (d) terminates with 0 errors eventually. Repeat the same tasks as in Step (f). Compare what you obtain here with what you have obtained in Step (f).

(h) Run Step (d) for $n = 60000$ and $\epsilon = 0$. Make note of (i.e., plot) the errors as the number of epochs grow large, and note that the algorithm may not converge. Comment on the results.

(i) Using your observations in the previous step, pick some appropriate value for ϵ (such that your algorithm in (d) will eventually terminate). Repeat the following two subitems three times with different initial weights and comment on the results:
- Run Step (d) for $n = 60000$, some η of your choice and the ϵ you picked.
- Run Step (e) to with the $\mathbf{W}$ you obtained in the previous step.