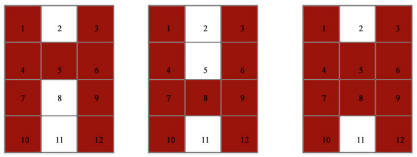
CSc11300 Programming Languages Final Project

Fall 2019

December 8, 2019

Instructor: Ahmet C. Yuksel *Deadline: Dec 25, 2019*

Python3 Software Project for creating a simple Python3 list (array) based (weightless) neural network. The project will have a dataset of 600 4X3 pixelated, black and white images for characters either L or H, that we can represent as arrays that contains 1 for black pixel, 0 for white pixel. 400 of those images will be the training set, and 200 of those images will be the testing set. To create such dataset, you only need to create arrays, that belongs to either Class L or Class H. For instance:



these different versions of character H, that belongs to Class H, can be represented as these Python3 lists respectively;

[1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1]
[1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1]
[1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1]

First part of the project is generating these arrays(not the actual images!), 300 for Class H, and 300 for Class L. Note that your image can have one or more pixel corrupted as the noise in the data. But we assume that the images are centered, normalized. A neural network works as training the system by giving some portion of your data and giving their classes for the training set and makes the neural network to be able to decide between classes of the testing set. After we train our neural network by providing 400 images as the training set, our network must be able to tell the class of 200 images from the testing set of which, classes are initially unknown to the neural network. Our neural network will decide which class, the image we provide from the testing set, belongs to.

An Array-Based (n-tuple, weightless) neural network works as the following;
Generate the index set for the arrays that have fixed size $N = 12$.

$I = \{1, ..., N\}$ representing all the indexes of the arrays (pixel positions of the images).

For example; the index set for all images for this project, including three sample images above, the index set will be $I = \{1, ..., 12\}$ since we are considering 4X3 images only. Also this does not have to be a set in Python, you can use a 1D list to represent that set.

Generate the list R, of the values that we store in those indexes. For instance;

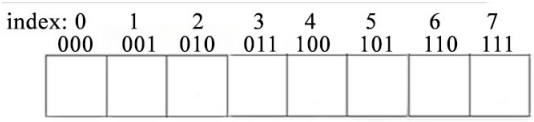
$R_{H1} = [1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1]$ representing all the values we store in those indexes for the first H character above. (Basically the array representation of the image)

Determine a tuple size n for sampling from the images. For instance, $n = 3$ we will sample 3 randomly selected points from the image. You can set the tuple size any value that is less than the number N. This must be something you can edit easily, because it affects the performance of your neural network. Recommended tuple size is $n = 3$.

Next, we create smaller, randomly generated index sets $J \subset I$ based on the tuple size we create. For instance if tuple size $n = 3$ we can have $J_1 = \{1, 5, 9\}$, $J_2 = \{2, 4, 8\}$, $J_3 = \{3, 6, 7\}$ and $J_4 = \{10, 11, 12\}$, note that the number of these sets are also a variable you should be able to change, but recommended number for those, $m = 4$

Similarly we create smaller lists S_m based on the smaller index sets we randomly create. For instance smaller lists for the first character H above are; $S_1 = [1, 1, 1]$, $S_2 = [0, 1, 0]$, $S_3 = [1, 1, 1]$ and $S_4 = [1, 0, 1]$. All yields to the values we store in indexes that are grouped above as index sets J

After that we create $m = 4$ empty arrays, for each class that will perform the machine learning for us. For instance, the arrays that are initially empty (you should have 0 as the initial values to be able to increment those values) for Class H will be as following; $T_{1H}, T_{2H}, T_{3H}, T_{4H}$, and for Class L will be as following; $T_{1L}, T_{2L}, T_{3L}, T_{4L}$. Each array will look like:



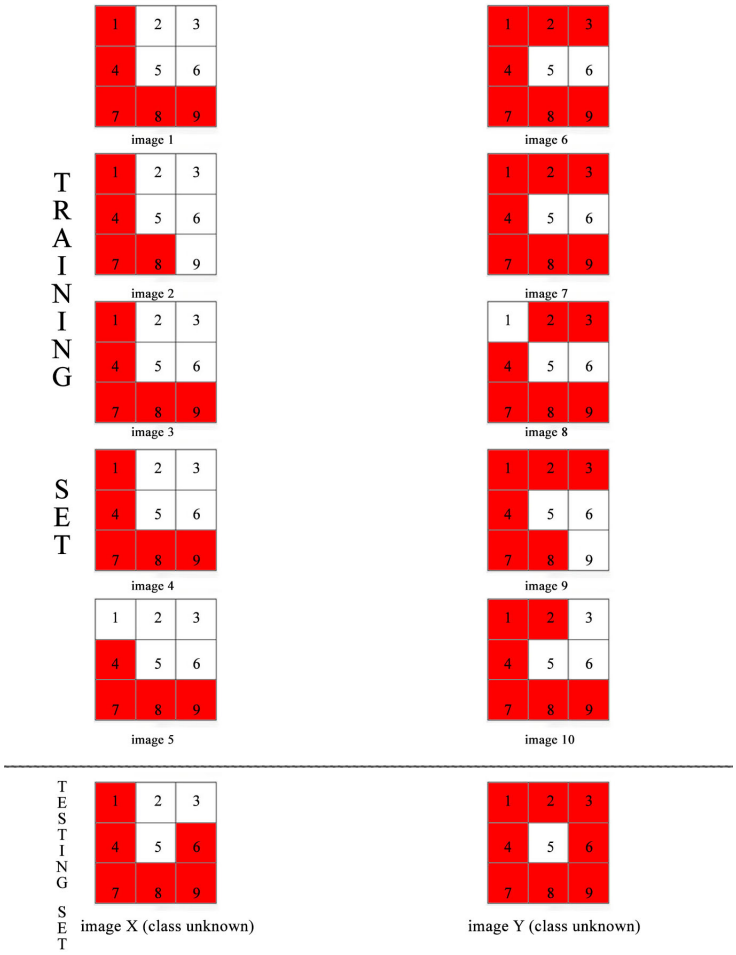
During the training of our neural network we will increment the occurrences of each n=3 tuples by 1, each time they appear as the value of smaller list S_m that belongs to that specific class. We will increment the index equals to the the tuple value. So if tuple 010 occurs, we increment the corresponding index (binary 010 yields to index number 2) by 1. By the end of this process you will have 8 arrays that counts the occurrences of the sampled tuples 4 arrays for Class H, 4 arrays for Class L, After doing so your neural network will gain the ability to decide, intelligence, to discriminate between classes. To see how well your program does that we should do some testing by using our testing set. "For the training, we provide the image as well as its class to our neural network."

During the testing we will use the same smaller index sets J_m for sampling, we will sample a same small portions of the image (array) and let computer figure out its class. For instance, tuple of the image from an unknown class, $S_1 = [1, 1, 1]$, comes up, so we check the value in that index and add it to a *sum* value, after we do this for all of 4 tuples for each class arrays, we will have a sum value from Class H arrays, and a sum value from Class L arrays. The image from the unknown class will belong to the class that gives the higher sum. Your program must print the arrays from the testing set, and its actual class, and the class that is predicted by the computer. If those classes are the same, add element "True" to the list. And after that an accuracy percentage which is calculated by the ratio of correctly classified classes.

–30 POINTS BONUS–:

While testing, if the predicted class is the correct class, increment the array values for that image (tuple occurrence) by adding some variable $+dn$ to each tuple occurrence and if the predicted class is not the actual class, decrement array values by adding some variable $-dn$ to each tuple occurrence. Adjust that value for n iterations (increase/decrease) based on the performance. So we increment and decrement the values in class arrays not necessarily by 1 but some variable dn which is up to you to determine the initial value. And change the value for this variable after each attempt of prediction. (Recommendation: Start with an initial value $0 < d < 1$) And observe how this small modification affects your accuracy.

Following pages contain an example of how this model of array based neural nets works with 3x3 images that belong to either Class L or Class C.



–TRAINING–

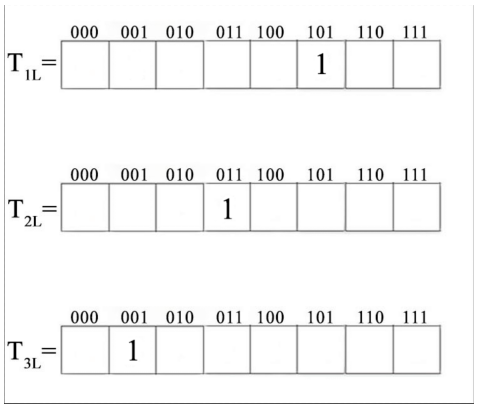
Number of classes=2, tuple size $n = 3$, number of $3-tuples = 3$(number of sets J), and the randomly generated sets Js are;

$J_1 = \{1, 5, 9\}$, $J_2 = \{2, 4, 8\}$, $J_3 = \{3, 6, 7\}$

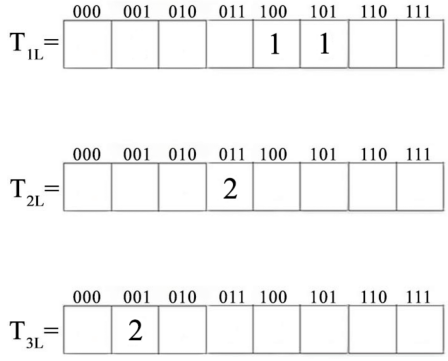
And the corresponding values to these indexes are;

For image 1: $S_1 = [1, 0, 1]$ (index for the array 1 of Class L), $S_2 = [0, 1, 1]$ (index for the array 2 of Class L), $S_3 = [0, 0, 1]$ (index for the array 3 of Class L)

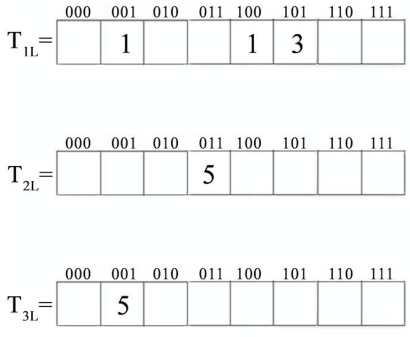
So we increment the following indexes in the 3 arrays of Class L by 1.
After training the neural network by using "image 1", the arrays of the Class L look like:



After training the neural network by using "image 2" with $J_1 = \{1, 5, 9\}$, $J_2 = \{2, 4, 8\}$, $J_3 = \{3, 6, 7\}$, and $S_1 = [1, 0, 0]$, $S_2 = [0, 1, 1]$, $S_3 = [0, 0, 1]$, the arrays of the Class L look like:



After the training the neural network by using all images that the classes are known to be L, the arrays of the Class L look like:



And after the training the neural network by using all images that the classes are known to be C, the arrays of the Class C look like:

T_{1C}=

000	001	010	011	100	101	110	111
	1				4		

T_{2C}=

000	001	010	011	100	101	110	111
							5

T_{3C}=

000	001	010	011	100	101	110	111
	1				4		

–TESTING–

Now an image (in your case an array) comes with an unknown class. Consider Image X, above. We use the same index set J to do sampling on that image. Thus, $J_1 = \{1, 5, 9\}$, $J_2 = \{2, 4, 8\}$, $J_3 = \{3, 6, 7\}$ yields to $S_1 = [1, 0, 1]$, $S_2 = [0, 1, 1]$, $S_3 = [0, 1, 1]$.

At this point we do not increment our array values for any of the classes but we will sum the values corresponding to those indexes: $S_1 = [1, 0, 1]$, $S_2 = [0, 1, 1]$, $S_3 = [0, 1, 1]$ for each 2 classes. And will decide that the image X belongs to the class that its arrays giving us the highest sum.

T_{1L}=

000	001	010	011	100	101	110	111
0	1	0	0	1	3	0	0

T_{1C}=

000	001	010	011	100	101	110	111
0	1	0	0	0	4	0	0

T_{2L}=

000	001	010	011	100	101	110	111
0	0	0	5	0	0	0	0

T_{2C}=

000	001	010	011	100	101	110	111
0	0	0	0	0	0	0	5

T_{3L}=

000	001	010	011	100	101	110	111
0	5	0	0	0	0	0	0

T_{3C}=

000	001	010	011	100	101	110	111
0	1	0	0	0	4	0	0

So this is Class L having score: 3+5+0=8 vs. Class C score: 4+0+0=4. Therefore image X belongs to Class L. Our program outputs:

[[1,0,0,1,0,1,1,1,1],"Actual Class:","L","Predicted Class:","L",True]
Accuracy 100%