Number Recognition Using Artificial Neural Networks
CSA2171

Shawn Sammut

142489(M)

Definition of the Problem and Goal

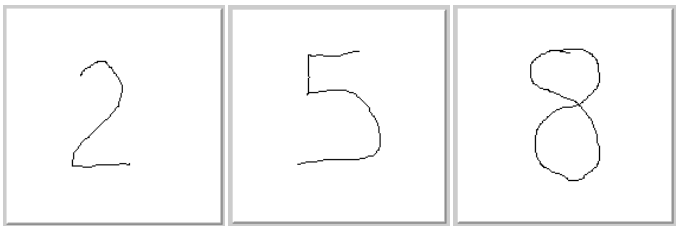
The problem at hand relates to the ability of an application implementing an Artificial Neural Network (aka ANN), which is able to recognize pictorial representations of digits, drawn by the user using a mouse. The application must be able to recognize digits from a range of zero to nine; moreover these pictorial representations must contain only one digit at a time. The ANN to be implemented must consist of the following:

- Input Layer, consisting of 64 nodes, thus it must receive an input of a 64-bit representation of the pictorial number inputted by the user.
- Hidden Layer, consisting of 37 nodes
- Output Layer, consisting of 10 nodes, reason being that the ANN must be able to recognize only numbers from zero to nine.
- A Back Prorogation Algorithm, used to change the weights of every node in the ANN accordingly.

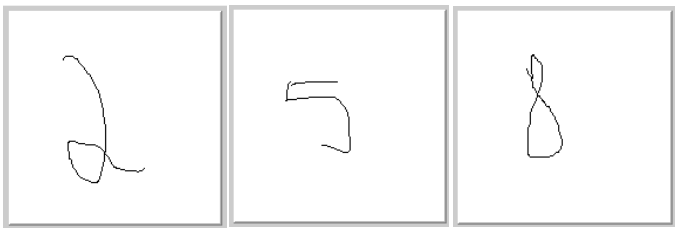
The ANN must "learn" on a test case created by the developer which consists of ten 64-bit representations for every digit from zero to nine. Thus this would result in a test case of hundred pictorial representations in the form of bits.

Statement of Completion

The application developed has achieved all the objectives it was designed for, i.e. the ability to recognize numbers represented by user input images, and outputting the equivalent number as an integer. The maximum and minimum recorded epochs, for the application to adjust all the weights such that the application would recognize the whole test case, was 210 and 43 respectively.
The application developed is able to recognize the majority of the numbers which were correctly inputted by the user. By correctly we mean pictorial representations as shown below:



On the other hand the application would not recognize pictorial representations of numbers inputted by the user either carelessly or in an unconventional manner. Few examples are given below:

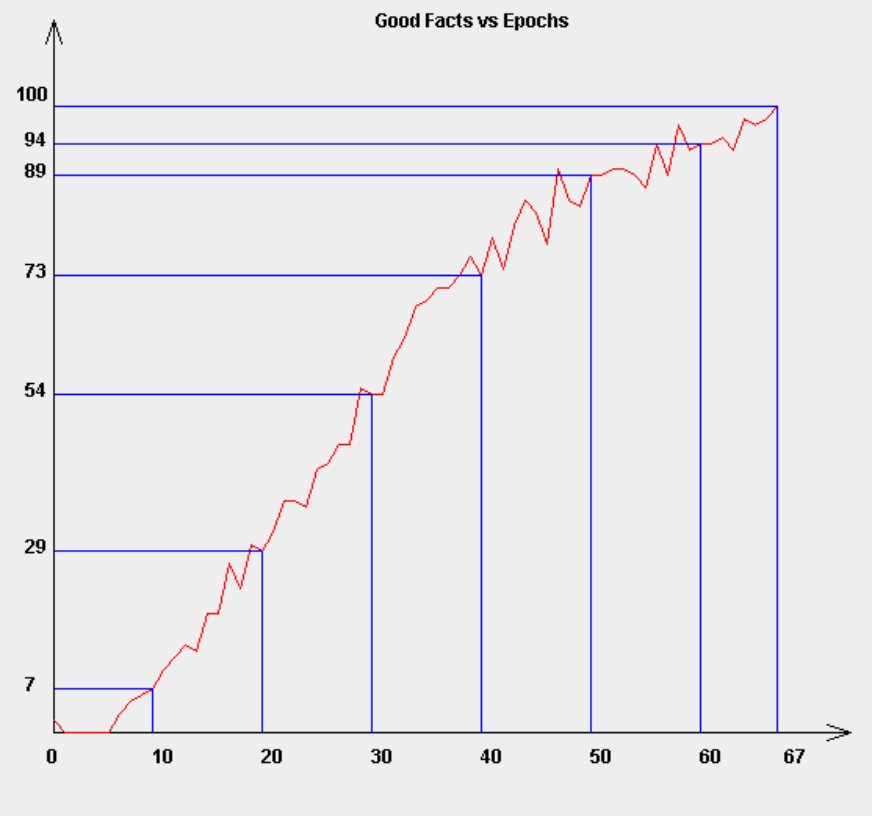


As a future development, the application would consist of an Artificial Neural Network with an Input Layer of 256 neurons, a Hidden Layer of 128 neurons and an Output Layer of 10 neurons. This would result in a 16x16 grid, thus pictorial representations of numbers would be more defined. Another improvement would be increasing the number of test cases which is retrieved by the ANN.

If the above improvements were to be implemented, the application would increase the level of number recognition drawn by the user, although one should note that this would also increase the amount of time taken for the application to adjust the weights accordingly in order to achieve an epoch consisting of 100 good facts.

Experimentation

The graph below is an example of how the ANN reacted till it reached its maximum number of good facts, which is 100. This graph is plotted automatically by the application after that the user has clicked the 'Teach Machine' button provided in the application. In the sample graph below, the ANN took 67 epochs in order to reach a 100 Good Facts.



The number of epochs taken to reach a 100 Good Facts is affected by several initial parameters which are the following:

- The values assigned to the weights, as the application is initialized, are never the same since the weights have randomized values assigned to them.
- The ANN's Threshold and Learning Rate value, where in this case they are set to 0.2 and 0.25 respectively.
- The number of test cases created for each digit which will be recognized by the application.

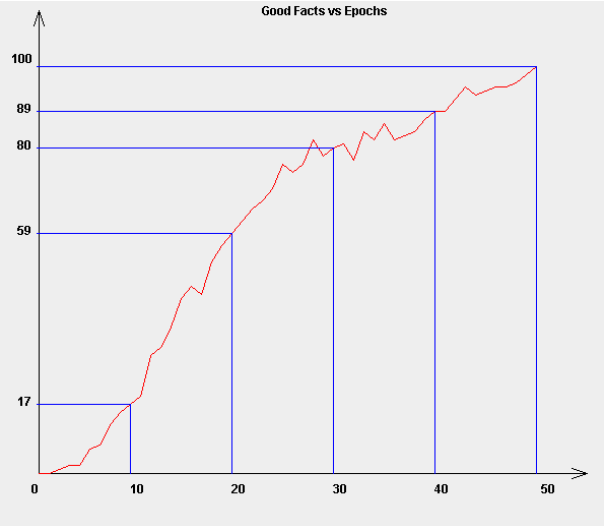
Decreasing/Increasing the Threshold value to 0.1/0.3

By decreasing/increasing the threshold value to 0.1/0.3, we are decreasing/increasing our marginal error allowed by the ANN. Thus if the difference between the neurons' output and the expected neurons output is greater than 0.1/0.3, the ANN will keep on looping epochs to the ANN until the expected marginal error is satisfied. Thus this has significantly increased/decreased the number of epochs executed in order to achieve 100 good facts. Therefore it can be deducted that:

$$Threshold \propto {}^{1}/_{Total\ Epochs}$$

Threshold Value: 0.1

Threshold Value: 0.3

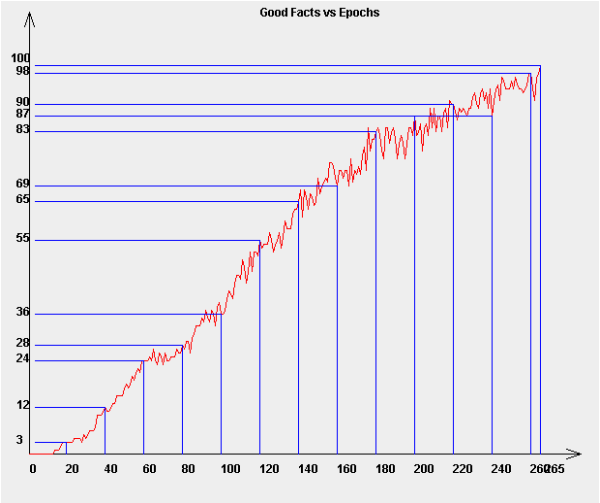


Decreasing/Increasing the Learning Rate value to 0.06/0.4

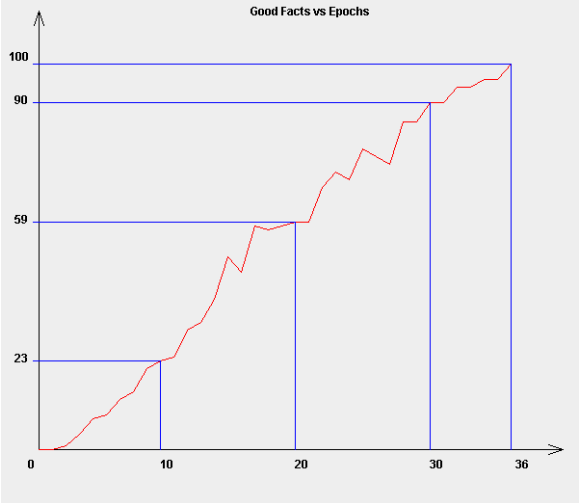
By decreasing/increasing the threshold value to 0.06/0.4, we are decreasing/increasing by how much the ANN changes the neurons weights. This has significantly increased/decreased the number of epochs executed in order to achieve 100 good facts. Therefore it can be deducted that:

$$Learning\ Rate \propto {}^{1}/_{Total\ Epochs}$$

Learning Rate Value: 0.06



Learning Rate Value: 0.4

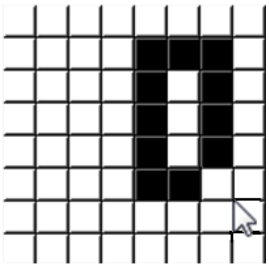


Extra Functionalities

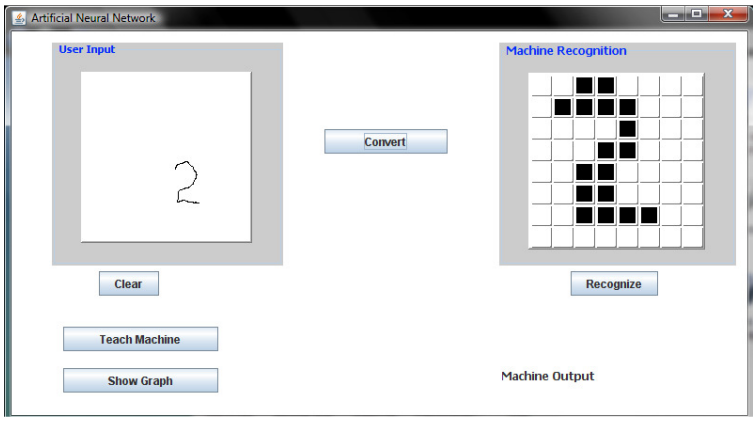
In order to increase the efficiency at which the application recognizes the pictorial representation of digits inputted by the user, the following has been implemented.

The user is able to draw a number onto screen by dragging the mouse onto a specified panel, rather than limiting to user to clicking onto large pixels which will represent the user's number.

The difference



The downside of this feature is that there is no restriction on the size of the users input drawing, thus this would give a hard time to the ANN. Thus the application developed is capable of focusing onto the drawn number on the panel and crops the image accordingly. After that the latter has been executed, the application enlarges the image to a size close to size that of what should have been drawn. An example is given below.



Thus, when creating the test cases, there was no need of considering transformation of images, meaning that all the numbers are centred into the panel. This has significantly reduced the number of epochs involved to make the ANN adjust the weights accordingly.

References

www.java2s.com