FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE
COIMBRA

Universidade de Coimbra

Assignment 2a

Optical Character Recognition

November 28, 2021

Author	*Student ID*
Francisco Fernandes	2018278239
João Marcelino	2018279700

Contents

1 Introduction

This report aims to explain the work done in Assignment 2a of Machine Learning course. The main goal of this assignment was to create different neural networks capable of recognizing handwritten digits. Each number is represented in a 16x16 binary matrix.

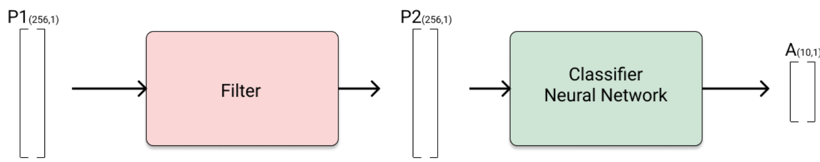


Figure 1: Optical Character Recognition system composed by the filter and classifier.

After being drawn in a 16x16 matrix, each digit is flattened into a 256x1 array. The array is then fed to a filter which, hopefully, smooths the digit strokes and turns it into a perfect representation. The goal of this phase is to make the following classification task easier in the sense that the strokes composing digits of the same class should possess very small differences.

The final stage conducted by the classifier is described as associating each image sample to an individual integer class ($[0, 1, 2, 3, 43, 5, 6, 7, 8, 9]$) with a certain degree of confidence.

Given this overview of the system, this report seeks to answer the following questions:

- **Which system components arrangement is best suited for the task at hand:**

  Filter + Classifier

  Only Classifier
- **Which filter architecture produces the best results:**

  Associative Memory

  Binary Perceptron
- **Which number of layers for the classifier produces the best results (1 or 2)**
- **Best activation function for the classifier:**

  Linear

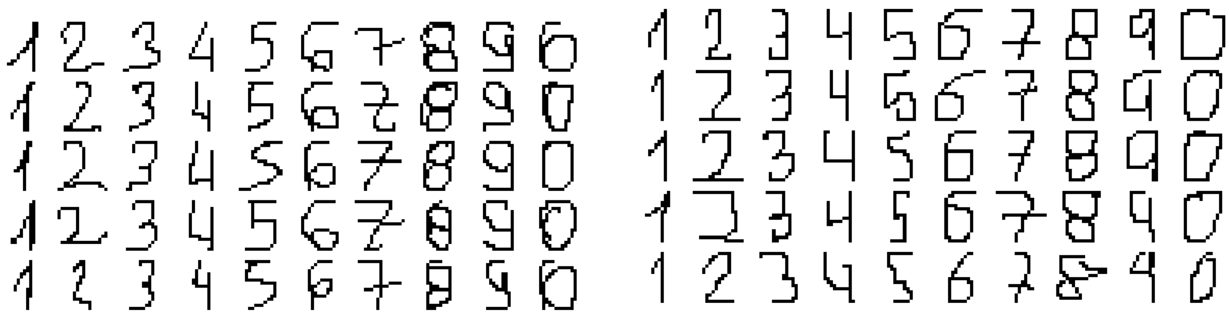
  Hard Limit

  Sigmoidal Funcion (Logistic, Hyperbolic Tangent, etc)

2 Data set

The first task of the assignment was the creation of the dataset. The main concerns in this task were class balancing and sample diversity. Each class should be represented in the dataset in equal amount of samples and each class should be composed of samples slightly different from each other (simulating different handwriting styles, for example) in order to allow the system to have a greater degree of generalization and robustness.

Using the provided matlab function *mpaper*, each element of the group created 500 different samples amounting to a total of 1000 samples (100 for each class). Since the dataset was relatively large it was decided to randomly subdivide it into 3 sets (train, validation and test sets) with the following proportions: 75%, 15%, 15%.



(a) (b)

Figure 2: Some examples of the handwritten digits created by different students.

3 Architectures

3.1 Filter

For this component, two different architectures were tested. Both are a simple network composed by a single layer of 256 neurons fully connected to 256 input sensors and 256 outputs.

- **Associative memory**

  - Linear activation function ($P_2 = W_P * P_1$);

  - No bias;

  - No backpropagation. The weight matrix $W_P(255, 25)$ is computed by $W_P = T * pinv(P1)$.

- **Binary Perceptron**

  - Hard limit activation function;

  - Perceptron training rule.

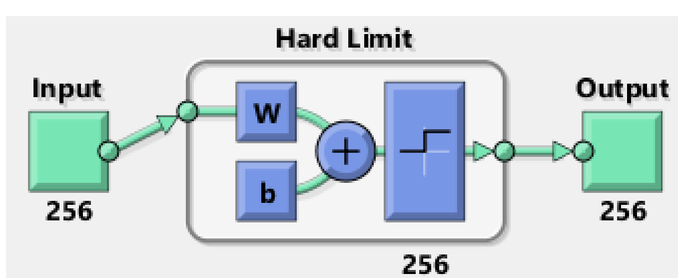
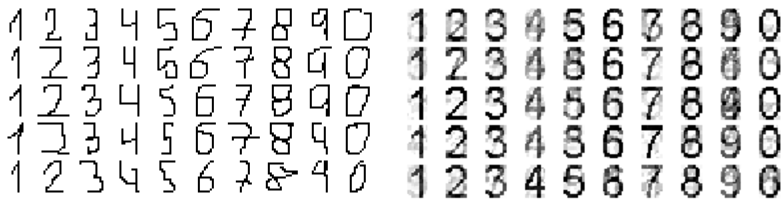
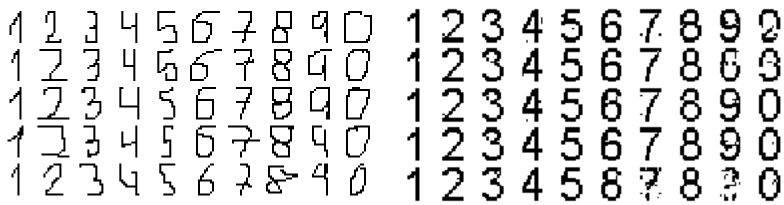


Figure 3: Perceptron filter architecture diagram.



(a) (b)

Figure 4: Input given to associative memory filter on the left (a) and output generated on the right (b).



(a) (b)

Figure 5: Input given to perceptron filter on the left (a) and output generated on the right (b).

As seen in figures 4 and 5, the outputs of the second network seem to approximate better the perfect digits provided beforehand that served as the training targets to both filters. Further results regarding the filter are referenced later in this report when it is joined together with the classifier and its accuracy is computed.

3.2 Classifier

As stated previously, the classifier should also be a simple feed forward neural network that accepts as input a 256x1 array containing an handwritten digit (or a near perfect representation in case a filter is applied beforehand) and outputs a 10x1 array of probabilities corresponding to the prediction confidence for each label class. Two choices about the composition of this network had to be made: number of layers (1 or 2) and the activation function to be used (hard limit, linear or sigmoid).

To understand which activation function was best suited for this problem, the accuracy of recognition was computed for three networks incorporating the activation functions aforementioned and with 5 different weight initialization values each. The number of epochs was fixed to 200, and no filter was applied to input. With each function, a grid search was conducted to find the best learning rate (from 0 to 1 with 0.1 increments).

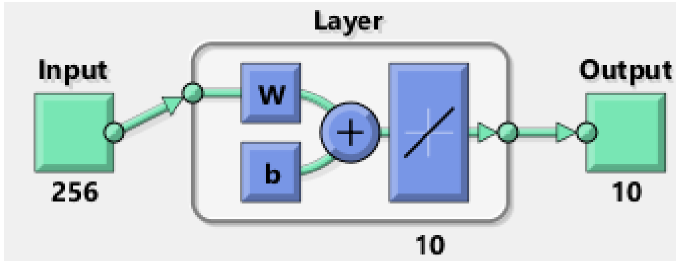


Figure 6: Single layer classifier with linear activation. During the first experiment only the activation function was changed.

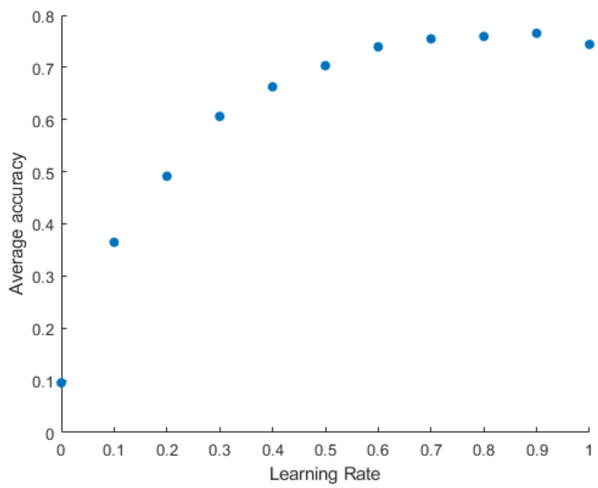


Figure 7: Example of accuracy evolution throughout learning rate search space. These specific values correspond to a 2 layer network with tangent sigmoid activation function in both layers.

Dataset \ Function	**Hard Limit**	**Linear**	**Sigmoid**
Test Set	**0.9633**	0.9307	0.8420
Training Set	**0.9651**	0.9317	0.8560

Figure 8: Average accuracy computed on test and train sets with each activation function. 5 distinct weight and bias initialization were considered. Learning rate was varied to find the value that produced the best outcome.

Dataset \ Function	**Linear**	**Sigmoid**	**Tangent Sigmoid**	**Sigmoid + Linear**
Test Set	0.1167	0.1193	**0.7653**	0.5253
Training Set	0.1014	0.1689	**0.8894**	0.5253

Figure 9: Average accuracy for various 2-layer networks. The same methodology was applied for networks of 1 layer and 2 layers.

As observed in Figure 8, the best activation function for 1-layer classifiers is the Hard Limit, having the best accuracy, in both the training and test sets. It's also worth noticing the Linear activation function, also with a good accuracy in the two data sets.

The results obtained in Figure 9 reveal that the best activation function for 2-layer classifiers is the Tangent Sigmoid, having a significant better training set than a test set. Although the Linear and Sigmoid activation functions had a good performance with 1-layer classifiers, they didn't achieve a good accuracy with the 2-layer classifier.

3.3 Filter + Classifier

In this part of the assignment we grouped the previous subsections and decided to test neural networks with data sets passing through a filter before training. In order to accomplish this this test, we used the same seed to all the networks used. The neural network used has 1-layer with sigmoidal activation function.

Testing \ Training	Non Filtered	Associative Memory	Perceptron
Non Filtered	**0.9590**	0.4080	0.3630
Associative Memory	0.5690	**0.9790**	0.9640
Perceptron	0.6300	0.9650	**0.9900**

Figure 10: Accuracy values of the testing datasets (Non Filtered, Filtered with Associative Memory and Filtered with Perceptron) in a neural network that trained with the same datasets. The network has 1-layer with sigmoidal activation.

After analyzing the table in figure 10, we can see that when the same data set is used for training the neural network and testing, the accuracy values are comparatively higher than when only one of the filter is only used in either the training or testing datasets. The accuracy values when one dataset is filtered with associative memory and the other is filtered with binary perceptron are just a slightly shorter than when the same dataset is used in both training and testing.

3.4 Softmax

The Softmax function is used as the last activation function of a neural network, in order to normalize the output of the network to a probability distribution over predicted output classes. Prior to the application of Softmax, the output values could be negative, greater than one, and might not sum to 1. After the utilization of this function, each value will be in the interval [0,1], and the values will add up to 1, so they can be interpreted as probabilities.It helps the interpretability of the outcomes of the neural network by translating the outcome to normalized confidence levels in the predictions made. This activation function was used in our project due to the number of labels in the data set.

4 Conclusion

In conclusion, the group was able to successfully build networks capable of recognizing handwritten digits, yielding accuracy values above 95% in some architectures. In general, the accuracy observed on the training set was maintained going into the test set, indicating a good generalization capability.

One key aspect noted in the results is the fact that the hard limit function outperformed the remaining functions when only 1 layer is considered for the classifier and, overall, networks with 2 layers performed notably worse than their counterparts. In spite of the project members carefully studding the *matlab* documentation related to the *network()* function, some of these performances, notably the ones observed for the 2-layer linear network, may be attributed to the misconfiguration of the network parameters.