UNIVERSIDAD DE CÓRDOBA
ESCUELA POLITÉCNICA SUPERIOR DE CÓRDOBA

COMPUTER SCIENCE ENGINEERING DEGREE
MENTION: COMPUTATION
FOURTH YEAR. FIRST QUADRIMESTER

INTRODUCTION TO COMPUTATIONAL MODELS

# Assignment 2: Multilayer Perceptron for classification problems

*Ventura Lucena Martínez*
31008689C
i72lumav@uco.es

Academic Year 2021-2022
Cordoba, November 7, 2021

# Contents

# List of Figures

# List of Tables

# List of Algorithms

**Abstract**

This lab assignment [1] serves as familiarisation with neural network computational models, in particular, with the multilayer perceptron. To do this, an implementation of the basic back-propagation algorithm for the multilayer perceptron has been carried out with an analysis checking the effect of the different parameters: network architecture or topology, moment factor ($\mu$), use of validation set, decrease of the learning rate ($\eta$) for each layer and so on). In special, this experiments will be tested in the off-line back-propagation algorithm, specific to this practice.

# 1 Architecture

The concept of **architecture** referred to neural networks makes mention not only of the number of neuronal layers or the number of neurons in each of them, but also the connection between neurons or layers, the type of neurons present and even the way in which they are trained.

## 1.1 Layers and Neurons

On the first hand, a **layer** is a set of neurons whose inputs come from a previous layer (or from the input data in the case of the first layer) and whose outputs are the input from a later layer. We will denote each layer as $l_i$.

On the second hand, the basic unit of the neural network is the **perceptron** or **neuron**. Each neuron has **inputs** with $x_i$ values, each one weighted with its corresponding **weight** $w_i$ to be optimized.

## 1.2 Connetions

The neural network has its **neurons fully connected**; this is, each neuron of the layer $i$ is fully connected with each neuron of the layer $i+1$, for any neuron from $i = 0$ to $i = n\_neurons$ - 1. In the experiments we will see the following cases:

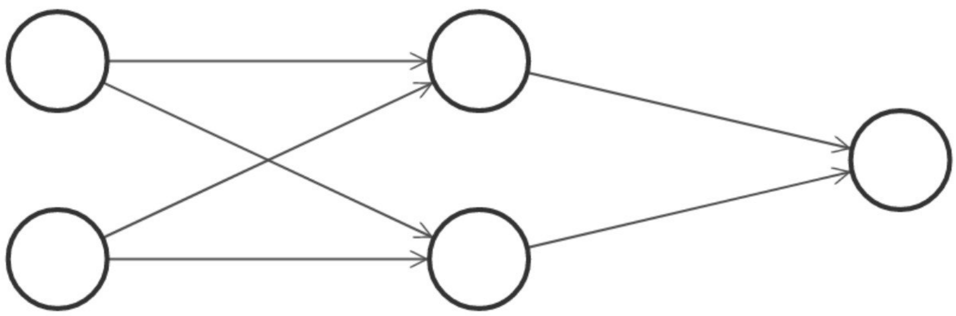


Figure 1: Topology based in 1 hidden layer and 2 neurons/each.

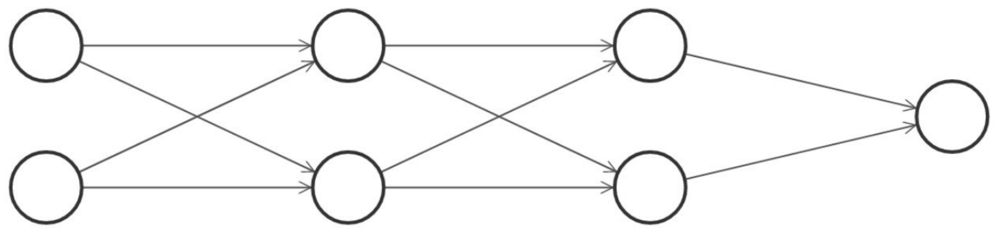


Figure 2: Topology based in 2 hidden layer and 2 neurons/each.

## 1.3 Type of neurons

We can see two types of neurons in the code:

### 1.3.1 Sigmoid activation function

As the previous lab assignment, we can find the on-line mode where the main types of neurons have a sigmoid activation function:

$$\sigma(x) = \frac{1}{1 + e^{-(x-t)}} \tag{1}$$

### 1.3.2 Softmax activation function

Furthermore, we have implemented the off-line mode, where we can find the softmax activation function. The network is configured to output N values, one for each class in the classification task, and the softmax function is used to normalize the outputs, converting them from weighted sum values into probabilities that sum to one. Each value in the output of the softmax function is interpreted as the probability of membership for each class:

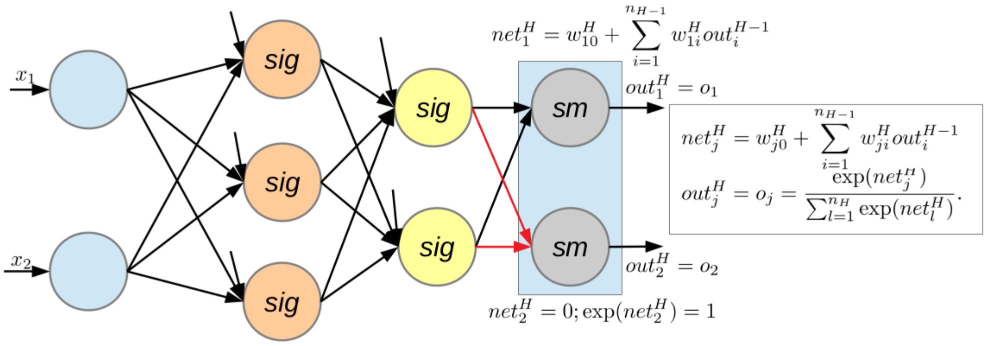



Figure 3: Softmax activation function. [2]

## 1.4 Training

Two types of error functions were used:

### 1.4.1 Minimum Square Error

It is an estimation method which minimizes the mean square error (MSE), which is a common measure of estimator quality, of the fitted values of a dependent variable. We recall that the MSE is defined as follows:

$$MSE = \frac{1}{N}\sum_{p=1}^{N}\left(\frac{1}{k}\sum_{o=1}^{k}(d_{po} - o_{po})^2\right) \tag{2}$$

### 1.4.2 Cross-Entropy

Cross-entropy is a measure from the field of information theory, building upon entropy and generally calculating the difference between two probability distributions.

$$L = \frac{1}{N}\sum_{p=1}^{N}\left(\frac{1}{k}\sum_{o=1}^{k}d_{po}ln(o_{po})\right) \tag{3}$$

- N: number of pattern in the considered dataset.
- $k$: number of outputs.
- $d_{po}$: target value for pattern $p$ and the output variable $o$.

- $o_{po}$: predicted value.

## 2 Back-propagation algorithm

The pseudo code that adheres to the *off-line* backpropagation algorithm is the following:

**Algorithm 1** Back-propagation

$i \leftarrow 1$
$w^h_{j,i} \leftarrow U[-1, 1]$ ▷ Random values between -1 and +1
**while** $StopCondition == false$ **do**
  **while** $i < nPatterns - 1$ **do**
    $\Delta w^h_{j,i} \leftarrow 0$
    $out^0_j \leftarrow x_j$
    $feedInputs()$;
    $forwardPropagate()$;
    $backpropagateError()$;
    $accumulateChange()$;
    $weightAdjustment()$;
  **end while**
**end while**

- *feedInput()*: feed the input neurons of the network with a vector passed as an argument.
- *forwardPropagate()*: calculate and propagate the outputs of the neurons, from the first layer until the last one.
- *backpropagateError()*: backpropagate the output error wrt a vector passed as an argument, from the last layer to the first one.
- *accumulateChange()*: accumulate the changes produced by one pattern and save them in $\Delta_w$.
- *weightAdjustment()*: update the network weights, from the first layer to the last one.

We have to consider some changes compared to the previous delivery:

1. *forwardPropagate()*: some changes were made in order to adapt the neurons of our neural network to a softmax function, so that, we have to pay attention at the last layer of the network, were we apply the softmax.

2. *backpropagateError()*: some changes were made related to the error function and the output function used:

   - Derivatives for sigmoid neurons:
     - Output layer:
       * MSE:

$$\delta_j^H \leftarrow -(d_j - out_j^H) \cdot out_j^H \cdot (1 - out_j^H) \tag{4}$$

       * Cross-entropy:

$$\delta_j^H \leftarrow -\left(\frac{d_j}{out_j^H}\right) \cdot out_j^H \cdot (1 - out_j^H) \tag{5}$$

     - Hidden layers:

$$\delta_j^h \leftarrow -\left(\sum_{i=1}^{n_{h+1}} w_{i,j}^{h+1} \delta_i^{h+1}\right) \cdot out_j^h \cdot (1 - out_j^h) \tag{6}$$

   - Derivatives for softmax functions (only output layer):
     - MSE:

$$\delta_j^H \leftarrow -\sum_{i=1}^{n_H} \Big((d_i - out_i^H) \cdot out_j^H (I(i == j) - out_i^H)\Big) \tag{7}$$

     - Cross-entropy:

$$\delta_j^H \leftarrow -\sum_{i=1}^{n_H} \left(\left(\frac{d_i}{out_i^H}\right) \cdot out_j^H (I(i == j) - out_i^H)\right) \tag{8}$$

3. *weightAdjustment()*: some modifications were made:

   - For each neuron from the first layer to layer $h$-1:

$$w_{j,i}^h \leftarrow w_{j,i}^h - \frac{\eta \Delta w_{j,i}^h}{N} - \frac{\mu\Big(\eta \Delta w_{j,i}^h(t-1)\Big)}{N} \tag{9}$$

   - For the bias:

$$w_{j,0}^h \leftarrow w_{j,0}^h - \frac{\eta \Delta w_{j,0}^h}{N} - \frac{\mu\Big(\eta \Delta w_{j,0}^h(t-1)\Big)}{N} \tag{10}$$

# 3 Experiments

We will test different neural network configurations and run each configuration with five seeds (1, 2, 3, 4 and 5) in the *off-line* mode of the algorithm. Based on the results obtained, the standard and mean deviation of the error will be obtained. Depending of the error function selected, the MSE error or Cross-entropy error will be calculated for both the training set and the test set.

To assess how the implemented algorithm works, we will use three datasets, in which we will use the following nomenclature:

- $n$: number of inputs.
- $h$: number of neurons in each hidden layer.
- $k$: number of outputs.
- $l$: number of hidden layers.

## 3.1 XOR problem

This dataset represents the problem of non-linear classification of the XOR. The same file will be used for train and test. We are using the best architecture obtained in the previous lab assignment:

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| {2 : 64 : 64 : 1} | 0.0353319 +- 0.0425446 | 0.0353319 +- 0.0425446 | 90 +- 12.2474 | 90 +- 12.2474 | 3.680225737 sec. |

Table 1: XOR test with $i$=1000, $l$=2, $h$=64, $e$=0.7, $m$=1, $f$=1 and $s$="true".

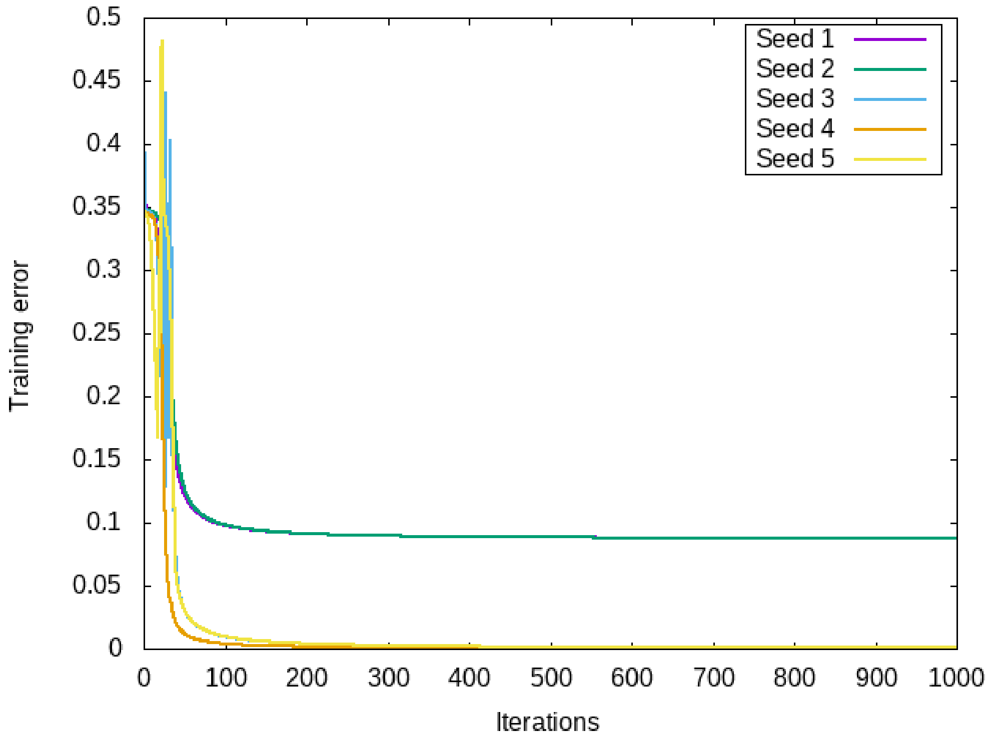



Figure 4: XOR test with $i$=1000, $l$=2, $h$=64, $e$=0.7, $m$=1, $f$=1 and $s$="true".

## 3.2 *Divorce* dataset [3]

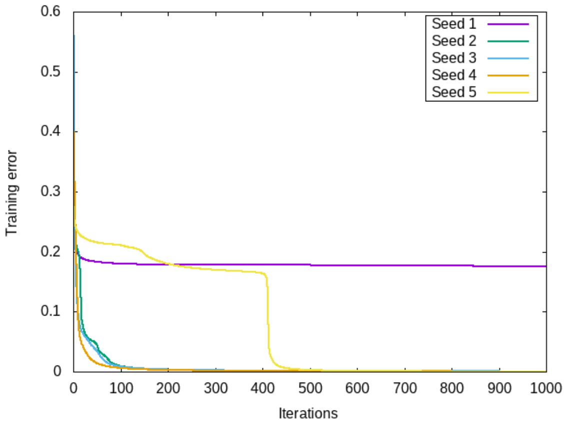
Contains 127 training patterns and 43 test patterns. The dataset contains the answer to a series of questions belonging to surveys, with the aim of predicting the divorce of a partner. The answers to the questions are provided in the Likert scale with values from 0 to 4. All the input variables are numerically considered. Two examples of questions are as follows:

- *I know my spouse's favourite food.*
- *I can tell you what kind of stress my spouse is facing in her/his life.*

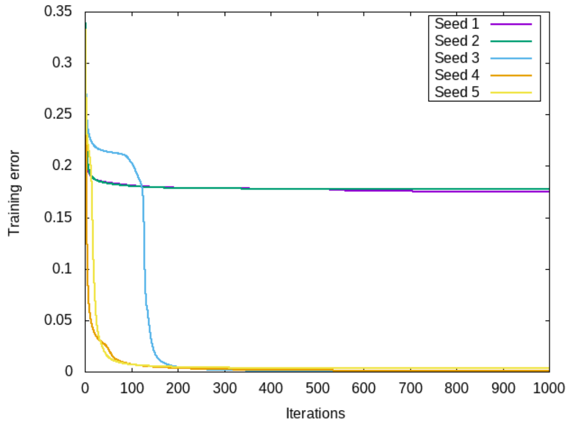
The dataset contains a total of 54 questions (therefore, 54 input variables) and two categories (0 if there is no divorce, 1 if there is a divorce).

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| {2 : 4 : 1} | 0.0355402 +- 0.0699472 | 0.0952655 +- 0.0455377 | 89.9213 +- 20.1575 | 87.907 +- 19.5349 | 4.359941884 sec. |
| {2 : 8 : 1} | 0.0712887 +- 0.0856362 | 0.114943 +- 0.0586943 | 79.685 +- 24.5609 | 78.1395 +- 23.9252 | 7.863889867 sec. |
| {2 : 16 : 1} | 0.0708392 +- 0.0851158 | 0.11342 +- 0.0603131 | 79.685 +- 24.5609 | 78.1395 +- 23.9252 | 14.556550625 sec. |
| {2 : 64 : 1} | 0.0712711 +- 0.0856006 | 0.11579 +- 0.0582648 | 79.685 +- 24.5609 | 78.1395 +- 23.9252 | 55.661486689 sec. |
| {2 : 4 : 4 : 1} | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.964434172 sec. |
| {2 : 8 : 8 : 1} | 0.000305938 +- 4.87469e-06 | 0.0796376 +- 0.0081996 | 100 +- 0 | 97.6744 +- 0 | 9.637060375 sec. |
| {2 : 16 : 16 : 1} | 0.0352078 +- 0.0698248 | 0.103608 +- 0.041106 | 89.9213 +- 20.1575 | 87.907 +- 19.5349 | 20.184049268 sec. |
| {2 : 64 : 64 : 1} | 0.000380435 +- 0.000174785 | 0.0830838 +- 0.00269926 | 100 +- 0 | 97.6744 +- 0 | 137.963661069 sec. |
| **Mean** | 0.035642928625 | 0.0978438625 | 89.8622 | 87.9069625 | 31.898884243625 sec. |

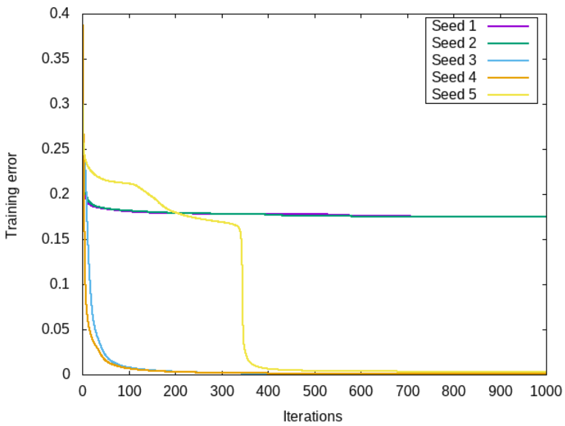
Table 2: *Divorce* dataset test with $i$=1000, $e$=0.7, $m$=1, $f$=1 and $s$="true".



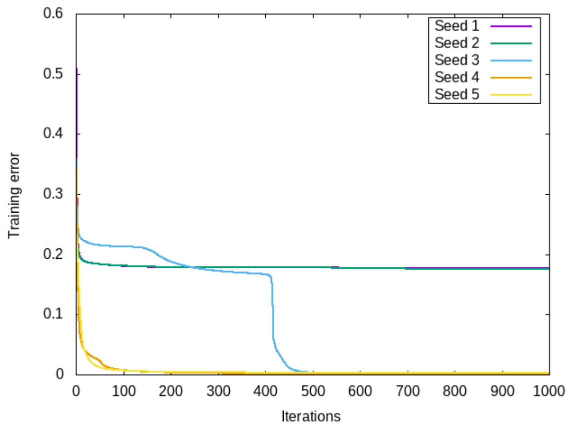

(a) $h$=4.




(b) $h$=8.

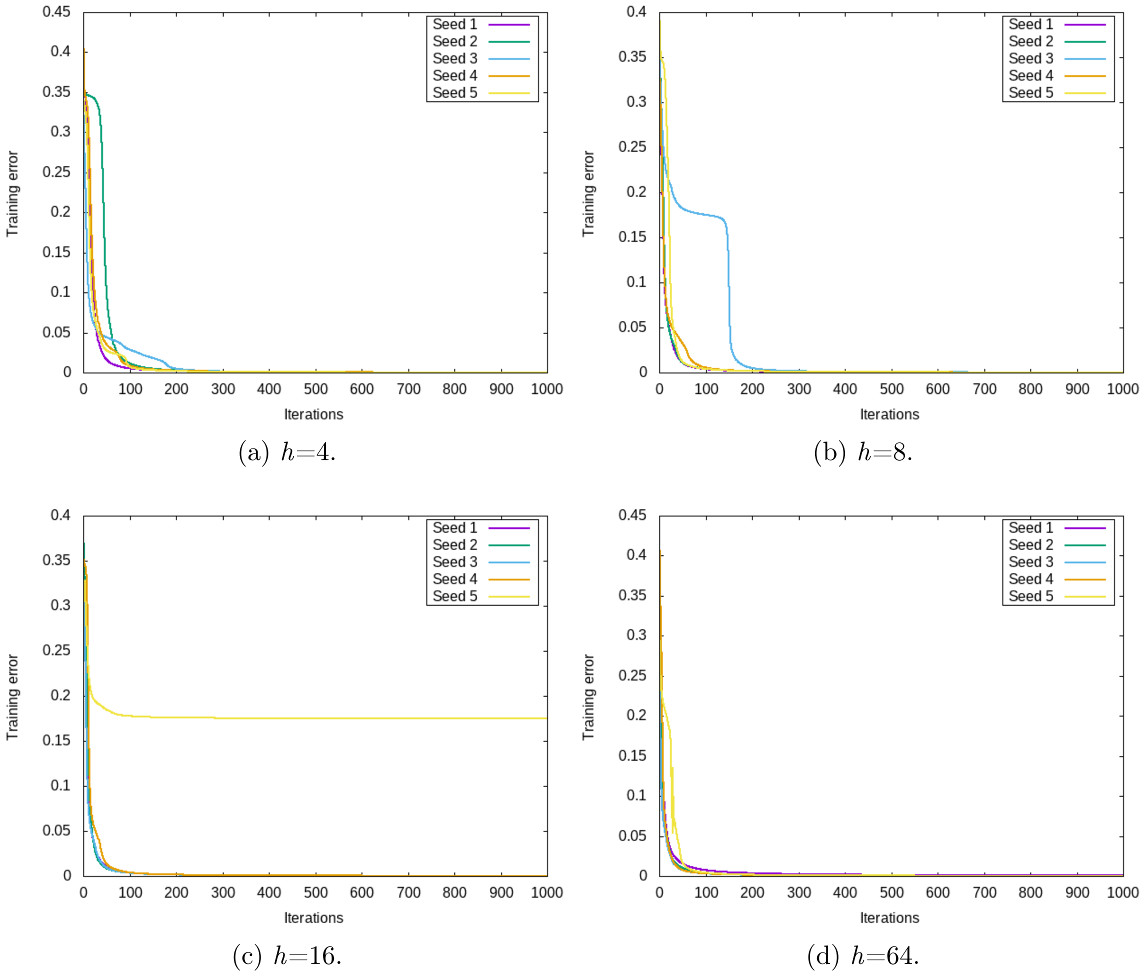



(c) $h$=16.




(d) $h$=64.

Figure 5: *Divorce* dataset test with $l$=1.




(a) $h$=4. (b) $h$=8.

(c) $h$=16. (d) $h$=64.

Figure 6: *Divorce* dataset test with $l$=2.

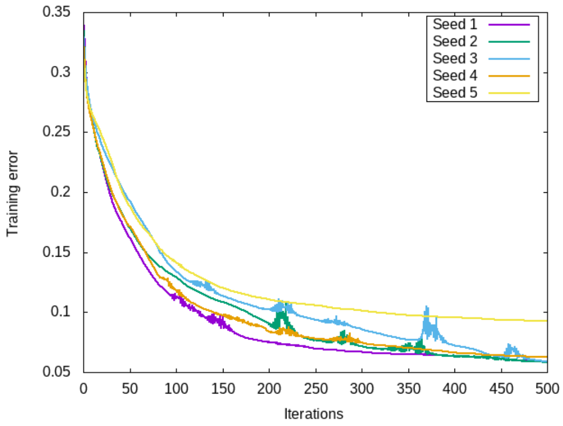
We can see that in most of cases, the error converge late. Only with 2 hidden layers and 4 or 64 neurons converge before ≈ 100-200 iterations.

### 3.3 *noMNIST* dataset [4]

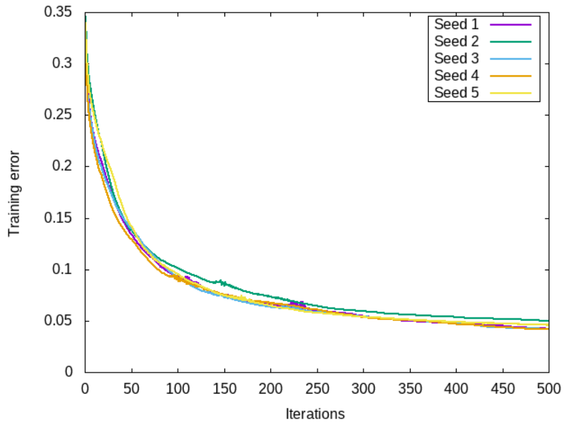
Originally, this dataset was composed by 200.000 training patterns and 10.000 test patterns, with a total of 10 classes. Nevertheless, for this lab assignment, the size of the dataset has been reduced in order to reduce the computational cost. In this sense, the dataset is composed by 900 training patterns and 300 test patterns. It includes a set of letters (from $a$ to $f$) written with different typologies or symbols. They are adjusted to a squared grid of $28 \times 28$ pixels. The images are in grey scale in the interval $[1.0; +1.0]$. Each of the pixels is an input variable (with a total of $28 \times 28 = 784$ input variables) and the class corresponds to a written letter ($a$, $b$, $c$, $d$, $e$ y $f$ , with a total of 6 classes).

| | **Train error** | **Test error** | **Train CRR** | **Test CRR** | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| {2 : 4 : 1} | 0.0663169 +- 0.0131514 | 0.126397 +- 0.0182808 | 87.9333 +- 3.92686 | 78.8667 +- 4.74506 | 200.045492339 sec. |
| {2 : 8 : 1} | 0.0443204 +- 0.0031206 | 0.110574 +- 0.0144564 | 92.4222 +- 0.741703 | 83.3333 +- 1.57762 | 398.660143182 sec. |
| {2 : 16 : 1} | 0.0472092 +- 0.00565299 | 0.1048 +- 0.00802431 | 91.2667 +- 1.55651 | 82.6667 +- 1.29957 | 750.885836373 sec. |
| {2 : 64 : 1} | 0.0460852 +- 0.00483626 | 0.113332 +- 0.0166279 | 92.0889 +- 0.735015 | 82.3333 +- 2.74064 | 2973.448589697 sec. |
| {2 : 4 : 4 : 1} | 0.0996585 +- 0.0186017 | 0.152566 +- 0.0199855 | 80.7333 +- 5.03597 | 71.5333 +- 4.29263 | 199.254751598 sec. |
| {2 : 8 : 8 : 1} | 0.0394151 +- 0.00973726 | 0.101302 +- 0.00771826 | 93.4889 +- 1.65939 | 83.2 +- 1.69444 | 383.415162230 sec. |
| {2 : 16 : 16 : 1} | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 721.434852521 sec. |
| {2 : 64 : 64 : 1} | 0.00393809 +- 0.000881529 | 0.0853702 +- 0.00459807 | 99.6444 +- 0.20367 | 87.6667 +- 0.918937 | 3201.359196764 sec. |
| **Mean** | 0.04527161125 | 0.1124449 | 91.933325 | 81.7916625 | 1103.563003088 sec. |

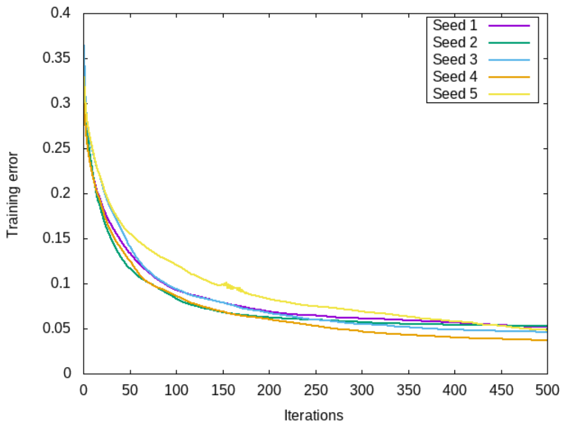
Table 3: *noMNIST* dataset test with $i$=1000, $e$=0.7, $m$=1, $f$=1 and $s$="true".



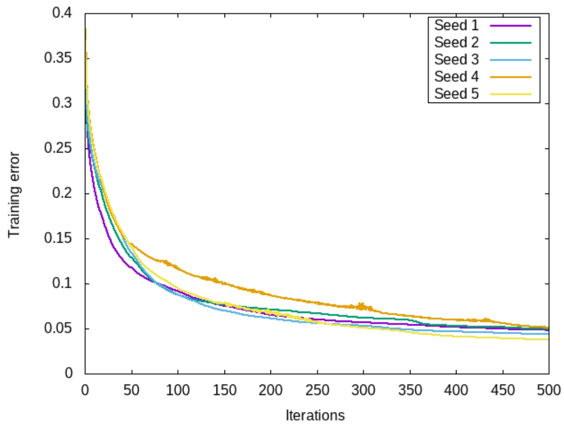

(a) $h$=4.




(b) $h$=8.

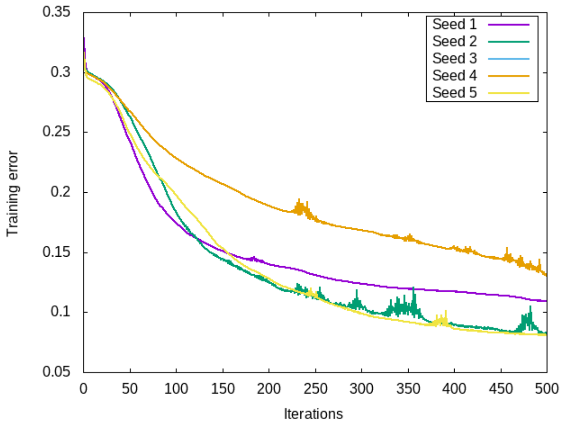



(c) $h$=16.

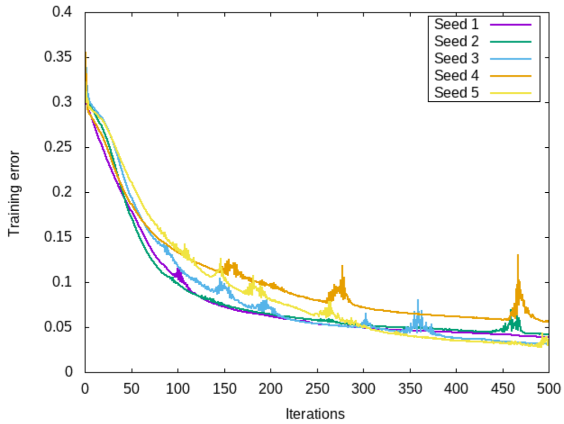



(d) $h$=64.

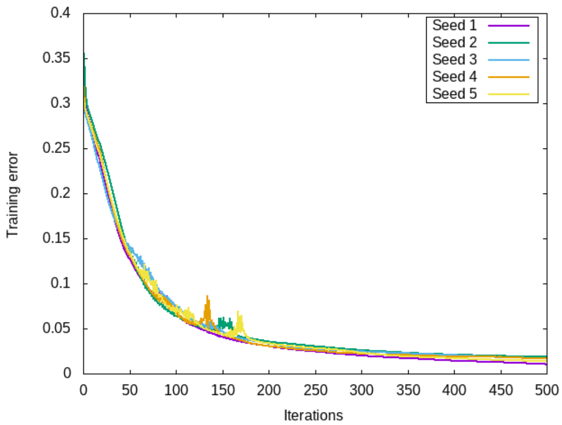
Figure 7: *noMNIST* dataset test with $l$=1.



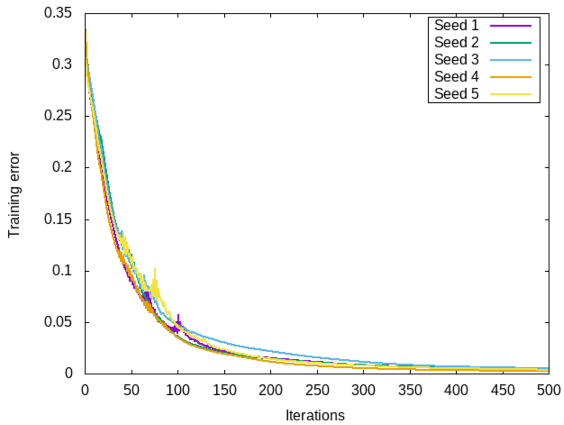

(a) $h$=4.




(b) $h$=8.




(c) $h$=16.




(d) $h$=64.

Figure 8: *noMNIST* dataset test with $l$=2.

Unlike the previous cases, we can see some noise in the error convergence. This means that prior to the fall, there are some iterations accepting an increase of the error.

# 4 Extra experiments

In order to select the best architectures for each dataset to test other combinations of the algorithm's parameters, we have calculated the mean of each measure as a simple filter. The chosen ones, with the best results below the mean, are as follows:

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| XOR $\equiv\{2:64:64:1\}$ | 0.0353319 +- 0.0425446 | 0.0353319 +- 0.0425446 | 90 +- 12.2474 | 90 +- 12.2474 | 3.680225737 sec. |
| *Divorce* $\equiv\{2:4:4:1\}$ | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.964434172 sec. |
| *noMNIST* $\equiv\{2:16:16:1\}$ | .0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 721.434852521 sec. |

Table 4: Selection of the best tests.

## 4.1 MSE error function and Sigmoidal activation function - *Off-line*

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| XOR $\equiv\{2:64:64:1\}$ | 8.29771e-06 +- 5.80867e-06 | 8.29771e-06 +- 5.80867e-06 | 50 +- 0 | 50 +- 0 | 3.839636259 sec. |
| *Divorce* $\equiv\{2:4:4:1\}$ | 0.000167934 +- 1.43591e-05 | 0.000168295 +- 1.45653e-05 | 64.5669 +- 18.4124 | 66.9767 +- 16.9879 | 4.724992648 sec. |
| *noMNIST* $\equiv\{2:16:16:1\}$ | 0.000130679 +- 9.37214e-06 | 0.000133133 +- 6.28972e-06 | 24.1111 +- 6.80087 | 22.2 +- 8.01831 | 708.656869003 sec. |

Table 5: MSE error function and Sigmoidal activation function - *Off-line*.

## 4.2 MSE error function and Softmax activation function - *Off-line*

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| XOR $\equiv\{2:64:64:1\}$ | 0.0353319 +- 0.0425446 | 0.0353319 +- 0.0425446 | 90 +- 12.2474 | 90 +- 12.2474 | 3.701127789 sec. |
| *Divorce* $\equiv\{2:4:4:1\}$ | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.856009135 sec. |
| *noMNIST* $\equiv\{2:16:16:1\}$ | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 705.583718563 sec. |

Table 6: MSE error function and Softmax activation function - *Off-line*.

## 4.3 Cross-Entropy error function and Softmax activation function - *Off-line*

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {n:h:k}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| XOR $\equiv\{2:64:64:1\}$ | 0.0353319 +- 0.0425446 | 0.0353319 +- 0.0425446 | 90 +- 12.2474 | 90 +- 12.2474 | 3.628350925 sec. |
| *Divorce* $\equiv\{2:4:4:1\}$ | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.913032513 sec. |
| *noMNIST* $\equiv\{2:16:16:1\}$ | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 708.648720247 sec. |

Table 7: Cross-Entropy error function and Softmax activation function - *Off-line*.

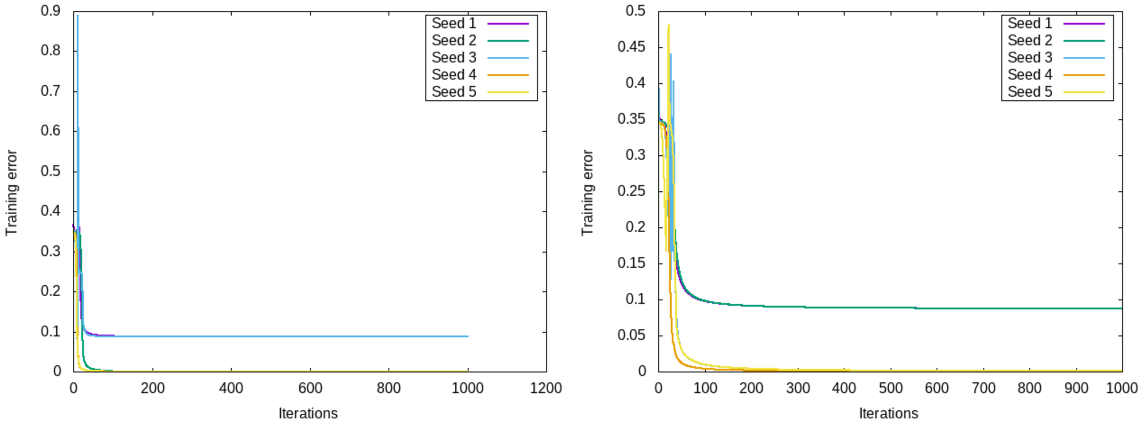
## 4.4 Performance comparison between *On-line* and best *Off-line* version

**Note:** The following tests have been carried out with Softmax activation function.

### 4.4.1 XOR problem

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {2 : 64 : 64 : 1}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| *On-line* and Cross-Entropy | 0.0303758 +- 0.0377676 | 0.0435455 +- 0.0550276 | 95 +- 10 | 95 +- 10 | 4.009749514 sec. |
| *Off-line* and Cross-Entropy | 0.0353319 +- 0.0425446 | 0.0353319 +- 0.0425446 | 90 +- 12.2474 | 90 +- 12.2474 | 3.628350925 sec. |

Table 8: Performance comparison between *On-line* and *Off-line* version: XOR problem.



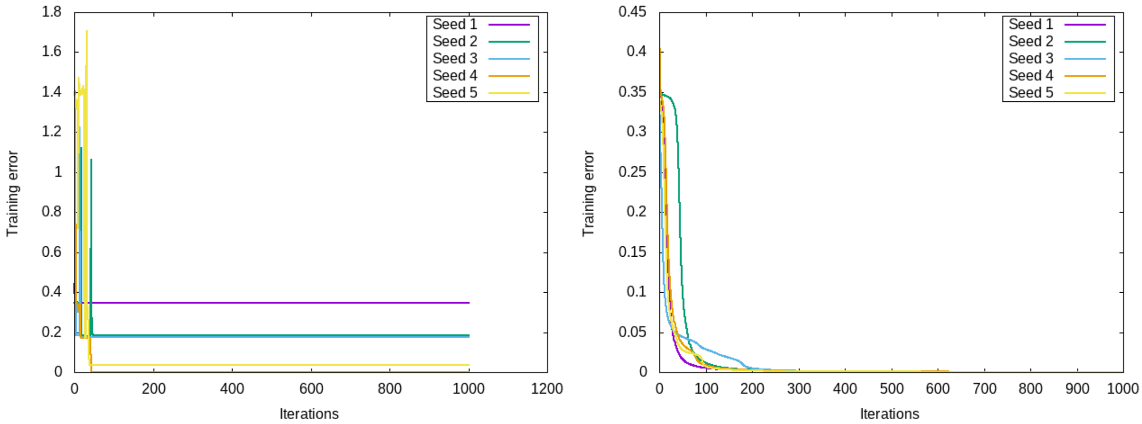

(a) *On-line* and Cross-Entropy (b) *Off-line* and Cross-Entropy

Figure 9: Training error performance comparison between *On-line* and *Off-line* version: XOR problem.

### 4.4.2 *Divorce* dataset

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {2 : 4 : 4 : 1}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| *On-line* and Cross-Entropy | 0.141893 +- 0.114517 | 0.604099 +- 0.848829 | 69.7638 +- 24.3693 | 68.8372 +- 23.5608 | 5.660838144 sec. |
| *Off-line* and MSE | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.856009135 sec. |

Table 9: Performance comparison between *On-line* and *Off-line* version: *Divorce* dataset.




(a) *On-line* and Cross-Entropy

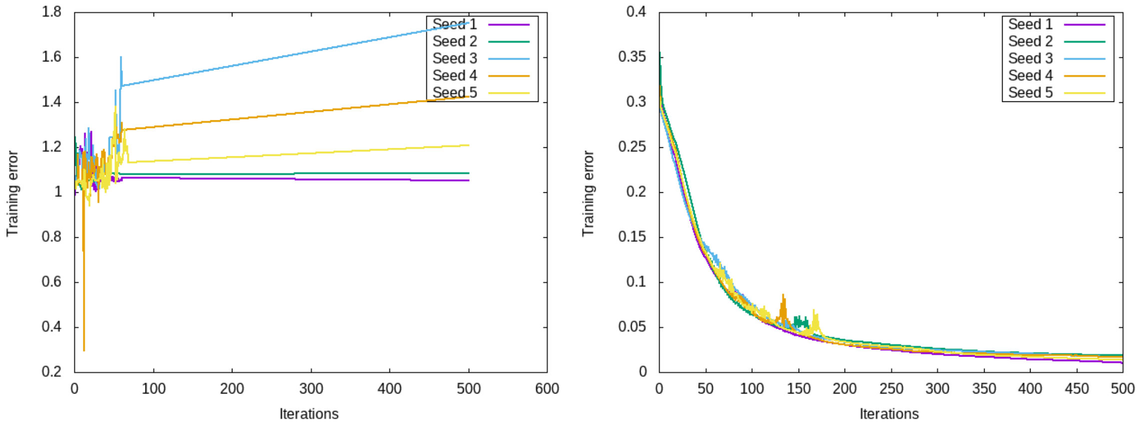
(b) *Off-line* and MSE

Figure 10: Training error performance comparison between *On-line* and *Off-line* version: *Divorce* dataset.

### 4.4.3 *noMNIST* dataset

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {2 : 16 : 16 : 1}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| *On-line* and Cross-Entropy | 0.827298 +- 0.269789 | 0.829695 +- 0.280602 | 18.5111 +- 3.99481 | 18.6 +- 3.95755 | 175.233547953 sec. |
| *Off-line* and MSE | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 705.583718563 sec. |

Table 10: Performance comparison between *On-line* and *Off-line* version: *noMNIST* dataset.




(a) *On-line* and Cross-Entropy (b) *Off-line* and MSE

Figure 11: Training error performance comparison between *On-line* and *Off-line* version: *Divorce* dataset.

In general, the *on-line* mode performs worse solutions (not valid at our discretion), except with the XOR problem. It is also remarkable the *on-line* mode performance with *noMNIST* dataset, where we can observe how the error behaves weird and rising.

## 4.5 Looking for a good configuration of parameters *v* and *F*

### 4.5.1 XOR problem

| | **Train error** | **Test error** | **Train CRR** | **Test CRR** | |
|---|---|---|---|---|---|
| **Architecture {2 : 64 : 64 : 1}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| v=0 / F=1 | 0.0353319 +- 0.0425446 | 0.0353319 +- 0.0425446 | 90 +- 12.2474 | 90 +- 12.2474 | 3.628350925 sec. |
| v=0.15 / F=1 | 0.0303758 +- 0.0377676 | 0.0435455 +- 0.0550276 | 95 +- 10 | 95 +- 10 | 4.175288289 sec. |
| v=0.25 / F=1 | 0.0303758 +- 0.0377676 | 0.0435455 +- 0.0550276 | 95 +- 10 | 95 +- 10 | 4.174899217 sec. |
| v=0 / F=2 | 0.0366147 +- 0.0729925 | 0.0367007 +- 0.0731643 | 95 +- 10 | 95 +- 10 | 4.081562668 sec. |
| v=0.15 / F=2 | 0.0366147 +- 0.0729925 | 0.0367007 +- 0.0731643 | 95 +- 10 | 95 +- 10 | 4.141878797 sec. |
| v=0.25 / F=2 | 0.0366147 +- 0.0729925 | 0.0367007 +- 0.0731643 | 95 +- 10 | 95 +- 10 | 4.205664541 sec. |

Table 11: Looking for a good configuration of parameters $v$ and $F$ - XOR problem.

The results obtained are good, independently of the configuration used. Of course, we got better train a test CRR, with *vneq* 0, but with a little increase of time, which is not disturbing.

### 4.5.2 *Divorce* dataset

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {2 : 64 : 64 : 1}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| v=0 / F=1 | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.856009135 sec. |
| v=0.15 / F=1 | 0.00734927 +- 0.0140673 | 0.070315 +- 0.0113459 | 99.685 +- 0.629921 | 97.6744 +- 0 | 4.213617629 sec. |
| v=0.25 / F=1 | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 5.179057563 sec. |
| v=0 / F=2 | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 4.966282229 sec. |
| v=0.15 / F=2 | 0.00734927 +- 0.0140673 | 0.070315 +- 0.0113459 | 99.685 +- 0.629921 | 97.6744 +- 0 | 4.244767281 sec. |
| v=0.25 / F=2 | 0.000310056 +- 1.3879e-05 | 0.077003 +- 0.00613462 | 100 +- 0 | 97.6744 +- 0 | 5.463856193 sec. |

Table 12: Looking for a good configuration of parameters $v$ and $F$ - *Divorce* test.

We obtained so similar results with the *divorce* dataset. No additional configuration provides more remarkable results.

### 4.5.3 *noMNIST* dataset

| | Train error | Test error | Train CRR | Test CRR | |
|---|---|---|---|---|---|
| **Architecture {2 : 64 : 64 : 1}** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Mean +- Std** | **Time** |
| v=0 / F=1 | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 705.583718563 sec. |
| v=0.15 / F=1 | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 745.388496170 sec. |
| v=0.25 / F=1 | 0.0152295 +- 0.00282579 | 0.105218 +- 0.0111421 | 97.8889 +- 0.616642 | 84.7333 +- 1.51144 | 775.102370716 sec. |
| v=0 / F=2 | 0.0202671 +- 0.00371521 | 0.104338 +- 0.00878891 | 97.1333 +- 0.802465 | 84.3333 +- 1.01105 | 719.653569782 sec. |
| v=0.15 / F=2 | 0.0202671 +- 0.00371521 | 0.104338 +- 0.00878891 | 97.1333 +- 0.802465 | 84.3333 +- 1.01105 | 738.140074976 sec. |
| v=0.25 / F=2 | 0.0202671 +- 0.00371521 | 0.104338 +- 0.00878891 | 97.1333 +- 0.802465 | 84.3333 +- 1.01105 | 762.742990588 sec. |

Table 13: Looking for a good configuration of parameters $v$ and $F$ - *noMNIST* test.

The last dataset has the most similar results. In this case, the best configuration strictly include an $F$=1.

# 5 Conclusion

In general, good results were obtained. In addition, we can observe how the *on-line* mode does not work very well with the datasets, although it actually do with the XOR problem. The differences among the results in each section are not very scattered, and without previous statistically study, we might suppose that are almost equal.

# References

[1] Pedro Antonio Gutiérrez Peña. *Lab assignment 2: Multilayer perceptron for classification problems.* URL: https://moodle.uco.es/m2122/mod/resource/view.php?id=49028.

[2] Pedro Antonio Gutiérrez Peña. *Lab assignment 2: additional contents.* URL: https://moodle.uco.es/m2122/mod/resource/view.php?id=49029.

[3] M. Yöntem, K. Adem, T. İlhan **and** S. Kılıçarslan. *Divorce prediction using correlation based feature selection and artificial neural networks.* Nevşehir Hacı Bektaş Veli University SBE Dergisi. 2019. URL: https://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set.

[4] *notMNIST dataset.* 2011. URL: http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html.