UNIVERSITÀ DI PISA

MASTER DEGREE IN EMBEDDED COMPUTING SYSTEMS

Computational Intelligence course

A neural network based "Smart Color Comparator"

Assignment report

Student:
Gabriele Serra

Professor:
Prof. Beatrice Lazzerini

Academic Year 2017/2018

Contents

List of Figures

1. Introduction

1.1 Printing in 2000s

Printing is a process that aim to reproduce text and images using a master template. The earliest known form of printing as applied to paper was woodblock printing, which appeared in China before 220 A.D. [1] Printing technology played a key role in the human history, mostly during the Renaissance and the scientific revolution. Of course these time are passed by and nowadays printing process has acquired another meaning in humans lives. Furthermore the evolution of technology spread the possibility to print also at own home. Of course this is not valid for each kind of printing process. In some areas of industry, printing it's still a very complex task to accomplish. This is true, for instance, for Macs Tech SRL, a company active in industrial printing of ceramic. Dott. Antonio Maccari is the CEO of Macs Tech and presided a lecture in University of Pisa on the importance of technology in this field.

1.2 Reproducing an image

1.2.1 Scanning

In order to have the most precise digital representation of an image, is necessary to use some advanced technique that scan the original image or the original material. This result can be reached by using specific instrument such as a *spectrometer*.

1.2.2 Printing

Furthermore, latest technologies helps in solving problem related to printers. For instance the transformation from digital image to workflow, namely the file formatted in a way that printer can interpret. Usually, software like photoshop are used to perform the majority of steps required.

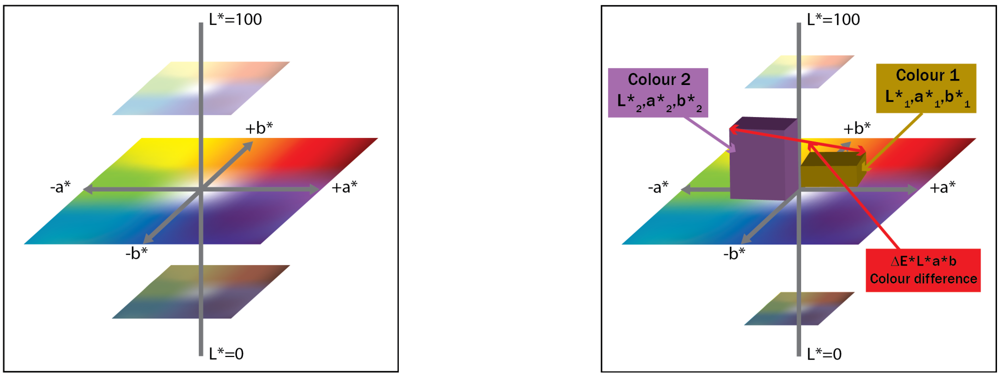
1.2.3 Conclusion

The point is that the problem is not present in scanning phase nor in printing phase. The problem in industrial painting is mainly related to the different representation of colors in digital environment with respect to the representation in the material in which they are printed. Often industrial processes are not able to generate the same reproduction over and over, and it is needed a workflow that is able to adjust "manually" colors based on subjective evaluation.

1.3 Solving the problem

1.3.1 Old-fashioned way

A first reproduction is built. When the process is completed, with a mask it is possible to check if the color has been reproduced correctly. The comparison is made pixel by pixel using a concept called ΔE. ΔE is calculated translating colors in a color-space called L*A*B*.



(a) *The L*a*b* color space describes mathematically all perceivable colors in the three dimensions L for lightness and a and b for the color opponents green–red and blue–yellow.*

(b) *The CIE calls their distance metric ΔE where delta is used to denote difference, and E stands for Empfindung; German for "sensation".*

Figure 1.1: *The L*a*b* color space and the ΔE as defined by CIE in 1976*

LAB space offer a very interesting features, in fact the distance of two colors in this space is related to human perception. Using this feature is possible to calculate color difference ΔE using the euclidean distance. Given two colors 1 and 2 in LAB color space, ΔE can be calculated as:

$$\Delta E = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2}$$

Anyway, the LAB color space was defined in 1941 and some error were introduced due to lack of knowledge in past. Hence the ΔE concept is something very unreliable. An effective real color comparison is not still discovered with optimal, because the human brain senses differences between colors depending not only to single pixels, but to overall adjacent colors and so on. Also if this kind of factors are not consider, ΔE is still not optimal [2].

1.3.2 Our proposal

The solution provided in this work instead try to take advantage of neural network to solve this problem, without using the ΔE concept. The developed tool takes as first input the entire set of reflected spectral power distribution of the first color (the color to be reproduced) sampled with a spectrometer and calculated using the standard illuminant D65[1]. As second the system takes the entire set of reflected spectral power distribution of the printed patch (the reproduction). At this point, the neural network will calculate the color difference based on a ground truth provided by some observers. Of course, greater is the number of observers/samples, greater is the ability of the network to generalize the result. For each input, the output of the network is a boolean value that indicate if the reproduction exactly[2] match the original patch.

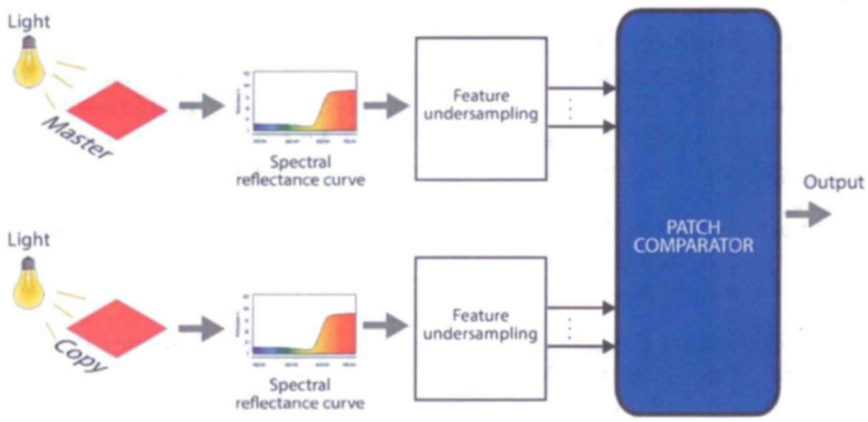


Figure 1.2: *System architecture overview*

[1]Commonly used standard illuminant defined by the International Commission on Illumination (CIE)

[2]Note that here "exactly" means that it is impossible to distinguish the two colors for a person.

2. Dataset and copies

2.1 Starting dataset

The starting dataset is made up of the reflectance spectra of 1269 matt color patches. This 1269 color patches represent our masters, namely the color to be reproduced by the industrial printing process. Reflectance spectra are measured with 1 nm step, ranging from 380 nm to 800 nm. This results into 421 samples for each reflectance curve [1]. Starting from spectra, XYZ color coordinates were calculated using D65 light source.

2.2 Generating copies

Considering that the master patches are represented using reflectance spectral curve, copies of a master color patch are generated through a perturbation that adds gaussian white noise to the curve.

From an operational point of view, we have considered the visible spectrum and divided it into wavelength ranges of 42 nm. In fact, approximately, the entire number of shades of a color are contained in 40 nm. Then we have added a fixed gaussian white noise. Unfortunately a great number of generated copies were too much similar if compared. Varying the entity of noise randomly in each interval produced a very good result by the point of view of variety, but the number of copies was exploding.

In order to get rid of the great number of copies generated and to get advantage of variety produced, the two approach had been fused into one, namely a variable intensity of gaussian white noise were applied on the entire interval of visible spectra. In particular 4 different intensity of noise were applied.

To apply noise, a function was developed. The MATLAB *awgn* is the basis of the developed function. Awgn accept two parameters: the signal to perturbate and the SNR after the

[1] Actually, only 1232 color patches have been used because was not possible to convert 37 patches to RGB.

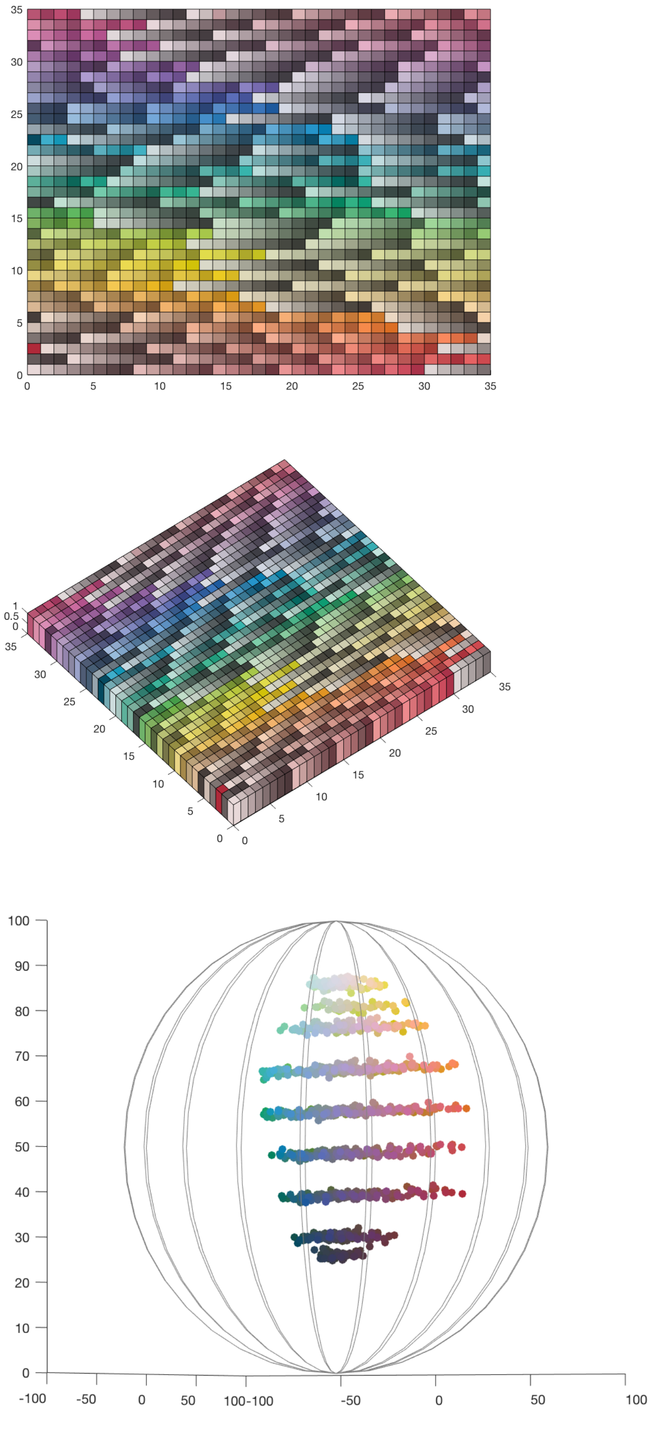


Figure 2.1: *The starting dataset in 2D, 3D and in the LAB space. Note that only 1225 patches are displayed for visualization reason*

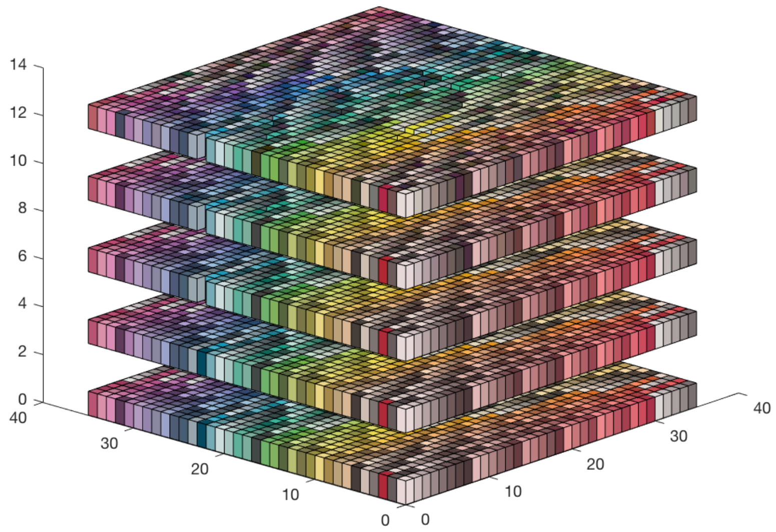


Figure 2.2: *The first plane is composed by the original dataset and the other are the copies in order of fidelity to the original. Where are present little holes, it means that the disturb ruined the RGB representation of the patch*

perturbation in dB.

$$SNR = 10 \cdot \log_{10} \frac{S}{N}$$

In our case the assumption is that the noise amount is surely less than the signal amount, in fact we hope that the industrial process print a color at least similar to the original. Hence, to obtain reasonable value of SNR to vary from the number of copy, an exponential function has been used.

$$SNR = f(x) = a \cdot e^{b \cdot x}$$

$$a = 63.97, b = -0.3284$$

Substituting 1, 2, 3, 4 we obtain four SNR value, from 45 db to 15 db

$$SNR = 10 \cdot \log_{10} \frac{S}{N} = 45 \rightarrow \frac{S}{N} \simeq 3 \cdot 10^4$$

$$SNR = 10 \cdot \log_{10} \frac{S}{N} = 15 \rightarrow \frac{S}{N} \simeq 30$$

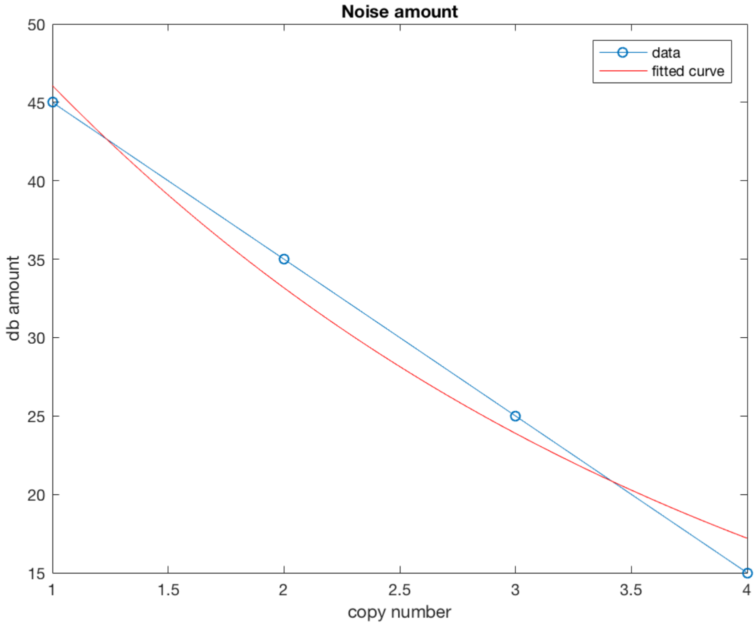


Figure 2.3: *Function used to generate db amount varying the copy number*

In conclusion, the intensity of the disturb after the application of awgn will be vary from 1/30000 and 1/30 compared to the original signal.

It is fundamental also to say that the *awgn* add the perturbation in random way, but can be forced to reproduce the same type of perturbation using the same seed for the random number generator. Thus, we have created three group of copies: the first group with the same pattern of perturbation on all samples, the second group with 50 different patterns of perturbation and the third group with 200 patterns of perturbation.

2.3 Dataset definition

The system is based on a neural network, so it need a great quantity of samples on which it will be train. For each master patch, 4 copies had been generated for each group. This means a total of possible $1232 \cdot 4 \cdot 3 \simeq 15000$ couples. Considering approximately 1 second for each evaluation, this means a total of approximately 11 hours. Of course is not reasonable for a person to spent 11 hours evaluating color differences. Thus the dataset has been divided in 3, considering only the 35% of the entire dataset, namely a total number of 1800 copies per group. Clearly is not possible to randomly choose the 35% of dataset, but it is necessary to carefully select a set of masters that maximize the coverage in color space. Hence the selection of masters was made using a clustering algorithm.

2.4 Master selection

The selection of masters has been made using unsupervised learning paradigma. No a priori knowledge is available, the network, which is trained without a teacher, learns by detecting the similarity between the input patterns. In particular, the used architecture is the SOM namely a Self-Organized Map. The SOM consists of a two dimensional array of identical neurons. The input vector is broadcast in parallel to all these neurons. In this work, the SOM was trained using 225 neurons. In fact, a fewer number of neurons was not good to create a reliable distribution of color.

There was the possibility to operate the selection using both a SOM created using as input LAB representation of the color or using as input the spectral reflectance curve of the masters. As we can see in the images, the spectral curve SOMs better represent the distribution of color shades. Once the SOM was built, the selection of masters (namely 450 patches) was made selecting two patch per neuron and randomly the missing ones. This was possible using a simple algorithm:

```
master = [];

// select approx. 450 patches
repeat(2 times) {
    for each (neuron in map) {
        add(master, neuron.linkedpatch)
        remove(neuron.linkedpatch)
    }
}

// select randomly the missing ones
while(length(master) < 450) {
    i = rand()
    add(master, neuron[i].linkedpatch)
    remove(neuron[i].linkedpatch)
}
```

2.5 Sets definition

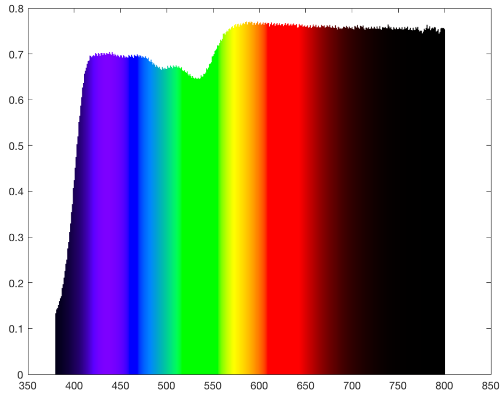
Finally it remains to discuss how were selected the three sets that were provided to the network. The definition of training set, testing set and validating set is a fundamental step. In order to reach good result, each of this sets must include all shades of color possibly distributed in uniform way. If these three sets are well-composed, the network will be trained, validated and tested on the same set of color shades. The subdivision was made, as common, 70% for training set and 15% for the other two. The masters selected were been 450 hence the total dimension of dataset consists in 450 master $\cdot$ 4 copies = 1800 couples of patch per group.

In particular 70% of total considered copies (315 master $\cdot$ 4 copies = 1260 couples of patch) was composed selecting in interleaved way a master and its copies. For the validating and for the testing set the remaining masters were equally subdivided in two groups of 67/68 patch and also them were subdivided in the two sets using the interleaved way. Of course selecting masters in random way was feasible but maybe the result could be affected in negative.

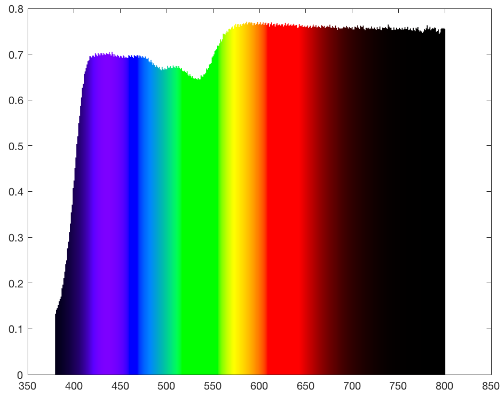
2.6 Target definitions

It is fundamental to explain how the target of the network has been created. The tagging phase has been completed using a simple tool developed in order to fastly compare two colors and decide if one is similar or different with respect to the other.

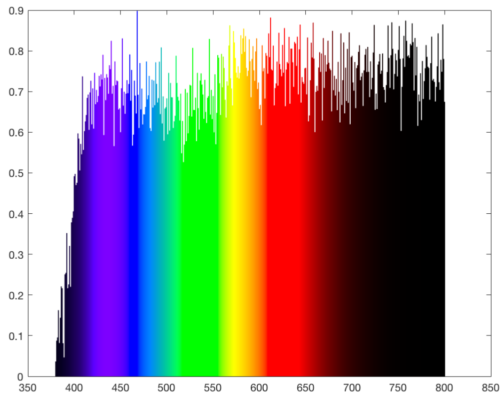
The functionality of this GUI are very simple, it displays two colors, the master and the copy. The user must decide if they are equal or not, and click next to continue. There is the possibility also to show the ΔE value between the two patches. The GUI is equipped also with keyboard callbacks that permit to interact with the GUI without select option with mouse. This can increment the number of tagged color per minute.



(a) *Spectral reflectance curve of the first patch.*

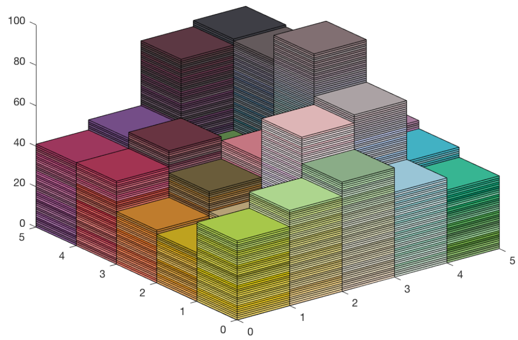


(b) *Spectral reflectance curve of the first patch after adding white gaussian noise (45 db SNR).*

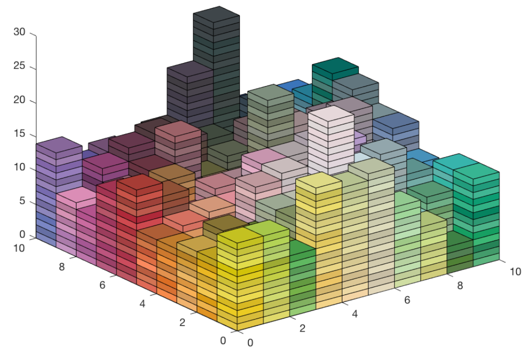


(c) *Spectral reflectance curve of the first patch after adding white gaussian noise (15 db SNR).*

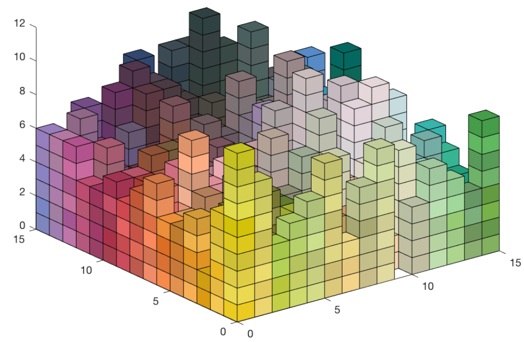
Figure 2.4: *The spectral reflectance curves of the first patch in its original form and two of its copies*



(a) *SOM built using LAB of each patch as input and 25 neurons.*

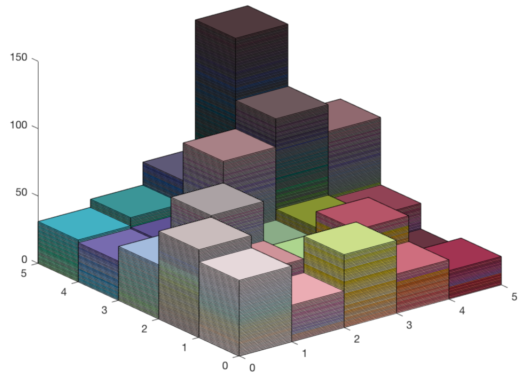


(b) *SOM built using LAB of each patch as input and 100 neurons.*

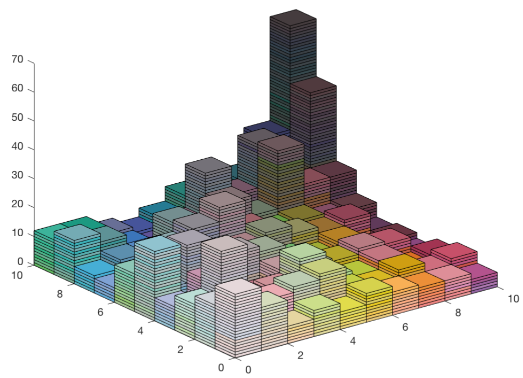


(c) *SOM built using LAB of each patch as input and 225 neurons.*

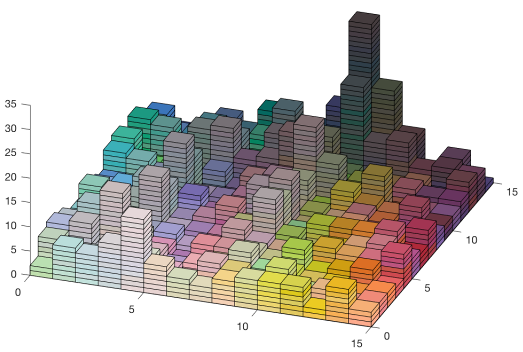
Figure 2.5: *SOMs built using LAB of each patch as network input*



(a) *SOM built using reflectance spectral curve of each patch as input and 25 neurons.*



(b) *SOM built using reflectance spectral curve of each patch as input and 100 neurons.*



(c) *SOM built using reflectance spectral curve of each patch as input and 225 neurons.*

Figure 2.6: *SOMs built with reflectance spectral curves of each patch as network input*

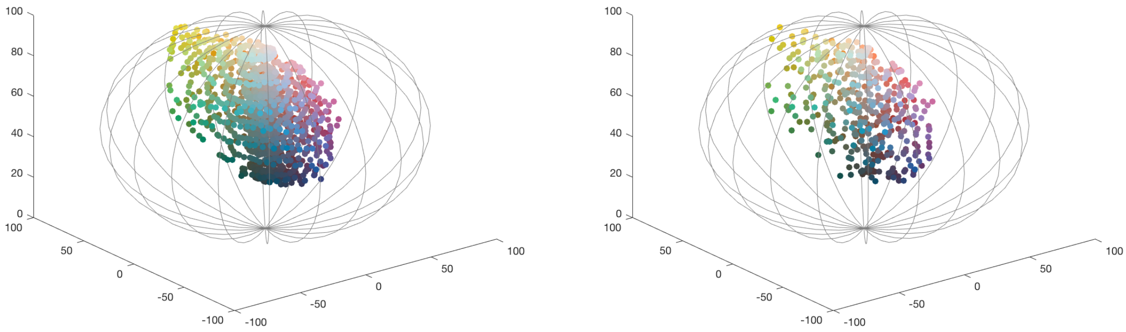


Figure 2.7: *The starting dataset LABs compared to selected master LABs in front view*

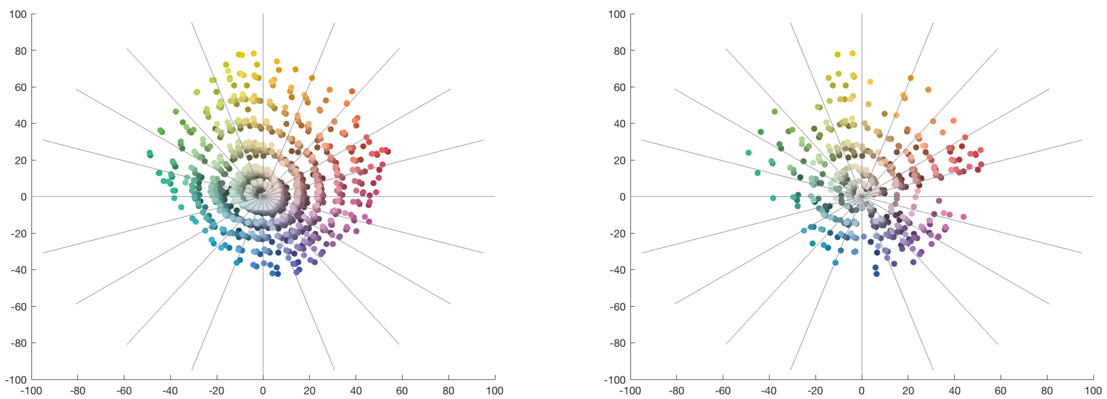
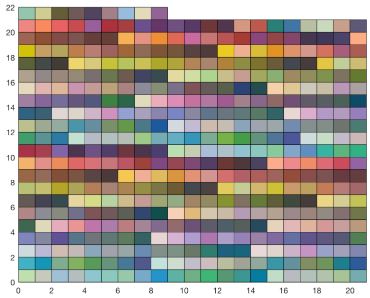
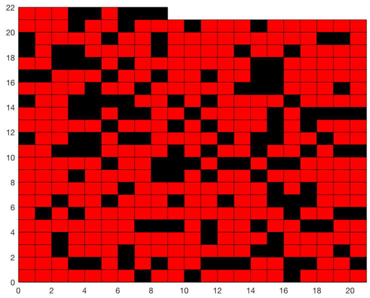


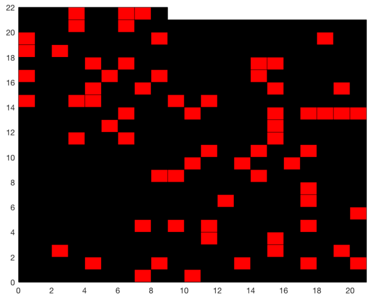
Figure 2.8: *The starting dataset LABs compared to selected master LABs in top view*



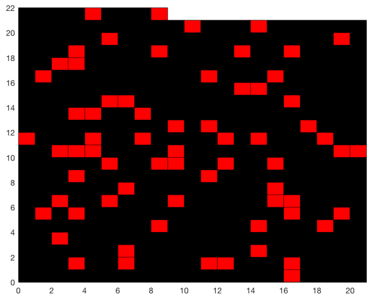
(a) *A visualization of selected masters as they are organized in the array.*



(b) *Masters selected as training-set in random mode.*

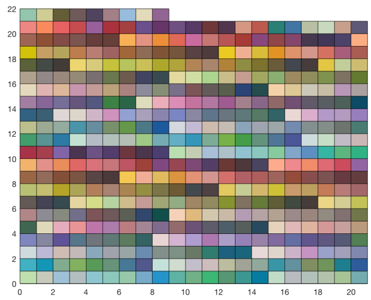


(c) *Masters selected as validating-set in random mode.*

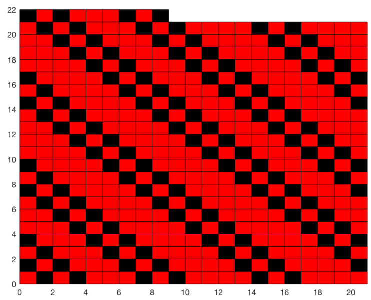


(d) *Masters selected as testing-set in random mode.*

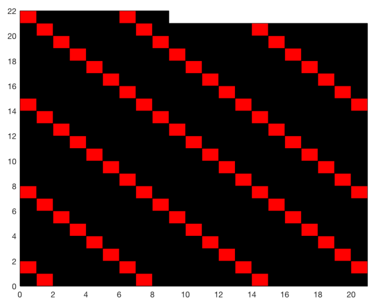
Figure 2.9: *Masters as they are organized in the array and subdivision in training, validating and testing sets using random mode*



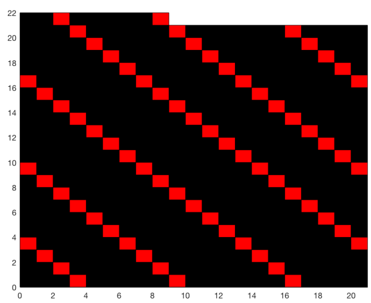
(a) *A visualization of selected masters as they are organized in the array.*



(b) *Masters selected as training-set in interleaved mode.*



(c) *Masters selected as validating-set in interleaved mode.*



(d) *Masters selected as testing-set in interleaved mode.*

Figure 2.10: *Masters as they are organized in the array and subdivision in training, validating and testing sets using interleaved mode*

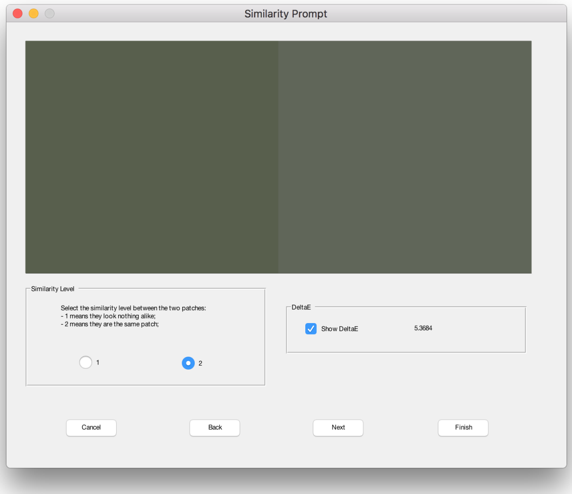


Figure 2.11: *Similarity GUI has been used to manually classify colors and it was developed using MATLAB GUIDE*

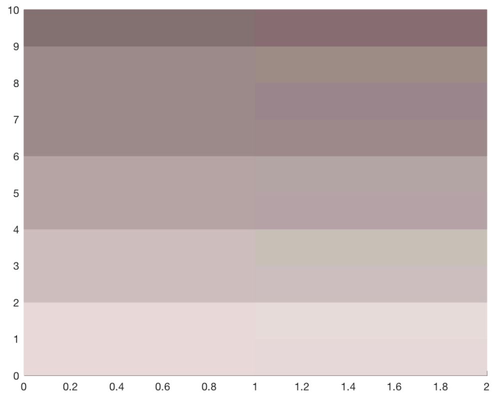


Figure 2.12: *An example of patches already compared using the similarity GUI. Of course they are all different.*

3. Networks

In order to maximize the quality of output, networks with different architectures has been used. At first, a multilayer perceptron (MLP) network has been used. Subsequently a radial basis functions network (RBF) was tried.

3.1 Multilayer perceptron

Multilayer perceptron is a neural network architecture that tries to map a set of input data in a set of output relying on the knowledge learned after a training phase. MLP networks are good at fitting functions. In fact, there is proof that a fairly simple neural network can fit any practical function. In addition to function fitting, MLP neural networks are also good at recognizing patterns. In practice, MATLAB offer two different architectures, *fitnet* and *patternet*

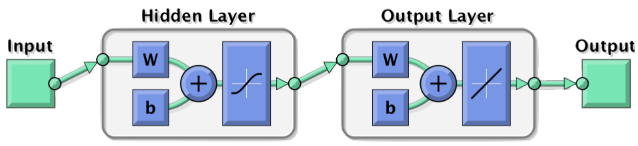
- Fitnet: a two-layer feed-forward network with sigmoid hidden neurons and linear output neurons.

- Patternet: a two-layer feed-forward network, with sigmoid hidden and softmax output neurons.

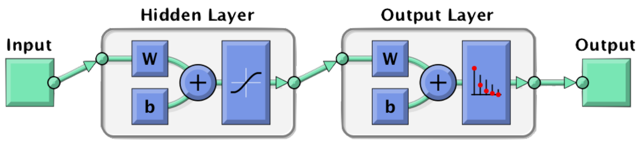
Both networks admit some parameters, such as the training algorithm and the number of neurons in the hidden layer. For memory/time reasons, the training algorithm used in all cases is *scaled conjugate gradient.* Thus, the only parameter that change in this work is the number of neurons in the hidden layer. Networks are trained 10 times and both the average regression value and the mean square error are calculated. The best network is chosen accordingly.

3.2 Radial basis function networks

Radial basis networks consist of two layers: a hidden radial basis layer of radbas neurons, and an output linear layer of linear neurons. In practice, MATLAB offer two way to create a RBF



(a) *Architecture overview of fit net.*



(b) *Architecture overview of pattern net.*

Figure 3.1: *MLP architectures overview*

network, *newrbe* and *newrb*

- Exact design (newrbe): creates as many radbas neurons as there are input vectors.
- Efficient design (newrb): iteratively creates a radial basis network adding one radbas neuron at a time.

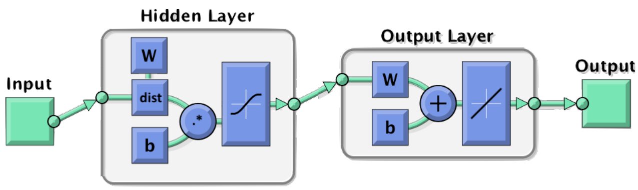


Figure 3.2: *Architecture overview of rbf net.*

For memory/time reason the first solution, namely the exact RBF design, was not take into account. In fact the great number of input (1800 input per group) make the solution unfeasible by the point of view of computation. The second solution, instead, was used to built a RBF network. In this work, both the spread and the max number of neuron are used as parameters. In fact it is important that the spread parameter be large enough that the radbas neurons respond to overlapping regions of the input space, but not so large that all the neurons respond in essentially the same manner. Thus the spread varies between the lowest distance between two adjacent vectors and the greatest distance between all vectors. In conclusion, all RBF networks are trained until a goal of 0.05 has reached and with a maximum number of neuron of 500.

3.3 RBF vs MLP

Radial basis networks, even when designed efficiently with newrb, tend to have many times more neurons than a comparable feedforward network. This is because sigmoid neurons can have outputs over a large region of the input space, while radbas neurons only respond to relatively small regions of the input space. The result is that the larger the input space (in terms of number of inputs, and the ranges those inputs vary over) the more radbas neurons required. On the other hand, designing a radial basis network often takes much less time than training a sigmoid/linear network, and can sometimes result in fewer neurons' being used. In the next chapter it is possible to see the advantage of each architecture and to derive the best for this kind of problem.

4. Experiments

4.1 Premise

Experiments and result evaluation are clearly related to the disturb (gaussian noise) that has been added to master patches. Of course is really intuitive to say that in presence of a lower number of disturb pattern[1] for the network is easier to generalize. But the important thing is to focus on how it is, for the network, easy to generalize and mostly when the network loses its capacity to get the job done. Hence, in next paragraph the result for each type of network (mlp fitnet, mlp patternet and rbf) will be analyzed in the context of perturbation pattern number, ending with a total number of 9 different results.

4.2 Simplest case: 1 pattern of noise

4.2.1 MLP fitnet

The best network accordingly to the mean square error value was the one with 30 neurons. Regression plot shows a very good capacity of the network to fit provided data. There are a considerable number of points that are a little far from the expected target but, as shown by the confusion matrix, the overall classification is done with an high degree of accuracy. By the way, the high variance of the regression value shows that a fitnet maybe is not the most suitable network for this kind of problem.

4.2.2 MLP patternet

The best network accordingly to the mean square error value was the one with 140 neurons. Here the regression plot show that, apart from isolate case, the classification was done in almost perfect way. The comparison between the confusion matrix of fitnet with the confusion

[1] Note that here we are not talking about noise intensity but noise patterns.

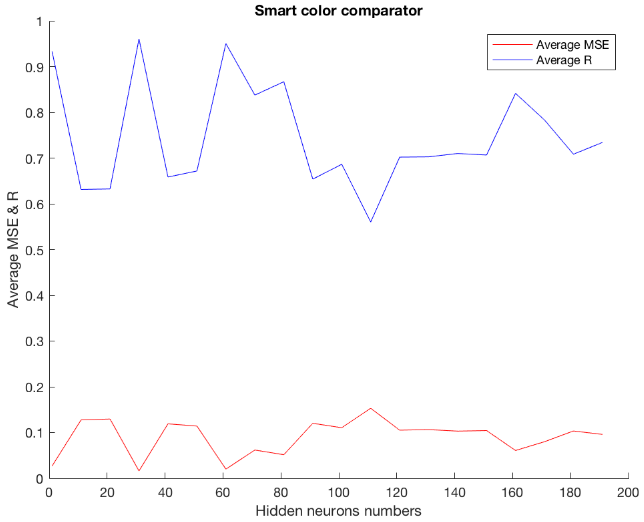


Figure 4.1: *R and MSE of MLP fitnet with 1 pattern of noise*

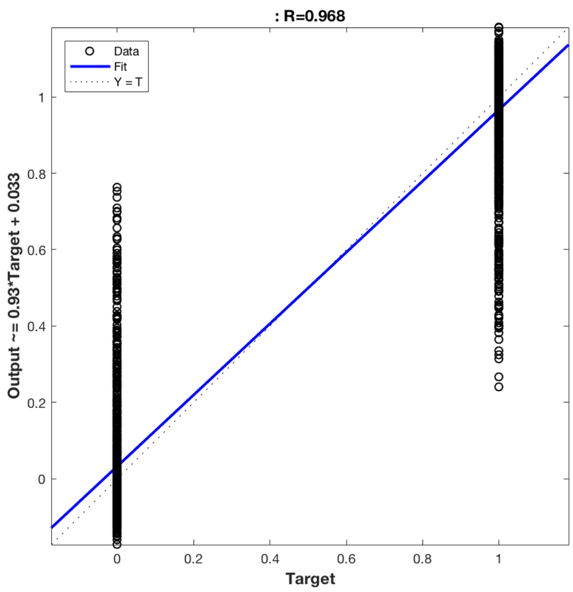


Figure 4.2: *Regression of best fitnet with 1 pattern of noise*

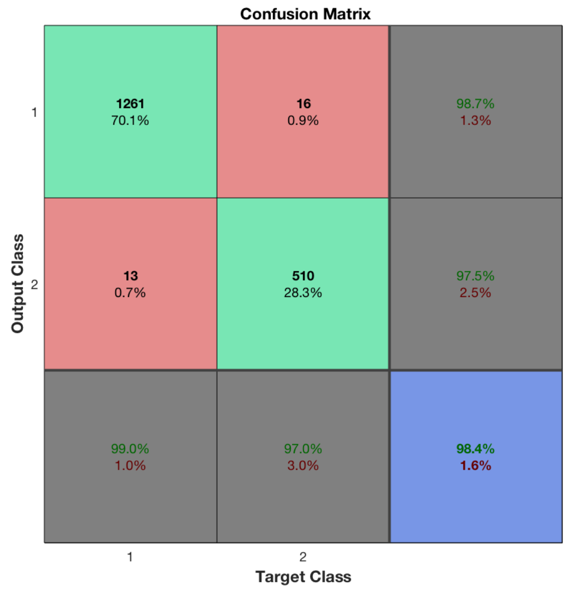


Figure 4.3: *Confusion matrix of the best fitnet with 1 pattern of noise*

matrix of patternet does not say something important but the low variance of regression value indicate that maybe this kind of network is the most suitable for this problem.

4.2.3 RBF

The best network accordingly to the mean square error value was the one with 50 neurons and a spread of 1100. The regression plot shows that the classification was done in good way also if there are some values that are little far from the expected target. Anyway the comparison between the confusion matrix of this network and the matrices of MLP networks reveal that also RBF is very good to solve this problem. In the figure it is possible to see the average MSE and the average Regression in function of the spread value. Spreads from 400 to 1100 are the most suitable to reach the goal with the lowest number of neurons.

4.3 A bit harder case: 50 pattern of noise

4.3.1 MLP fitnet

The best network accordingly to the mean square error value was the one with 20 neurons. Regression plot shows that the network tries to fit data and the majority of the time it do

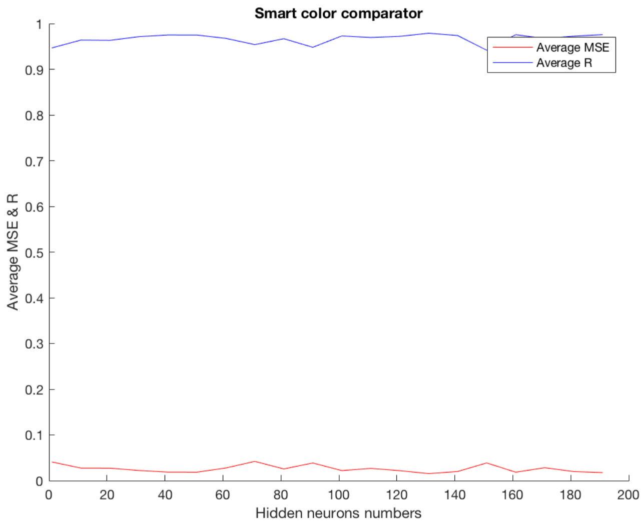


Figure 4.4: *R and MSE of MLP patternet with 1 pattern of noise*

well. Of course, with respect to MLP with 1 pattern of noise, the result is poor, but still acceptable. Anyway, if we consider how the points are classified, the percentage of correct classified points exceed 88%.

4.3.2 MLP patternet

The best network accordingly to the mean square error value was the one with 190 neurons. Here the regression plot show that the classification was done in approximately way. As in the previous case, the situation is acceptable considering the confusion matrix. The comparison between the confusion matrix of fitnet with the confusion matrix of patternet does not say something important as in the previous case but on contrary, when the number of pattern noise rise, a fitting network provide better result. A little remark, there is a great set of points in the upper-left part of the regression plot. This means that the network sometimes output '1' (different patches) instead of '0' (same patches, namely the expected target). This can depends by the little imbalance of the dataset provided to the network.

4.3.3 RBF

In this situation, was not possible to reach the goal of 0.05, hence the maximum number of neurons was reached in all trials. The best network accordingly to the mean square error value was the one with 500 neurons and spread 20. Regression plot shows a very poor capacity of the network to classify provided data. Most of time the network produce a value that seems

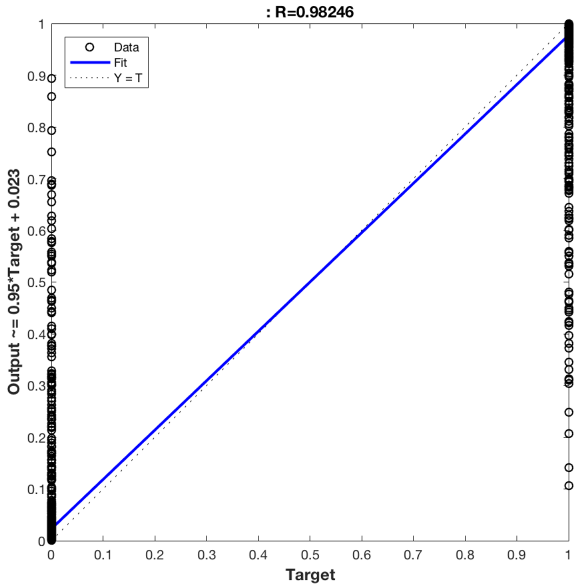


Figure 4.5: *Regression of best patternet with 1 pattern of noise*

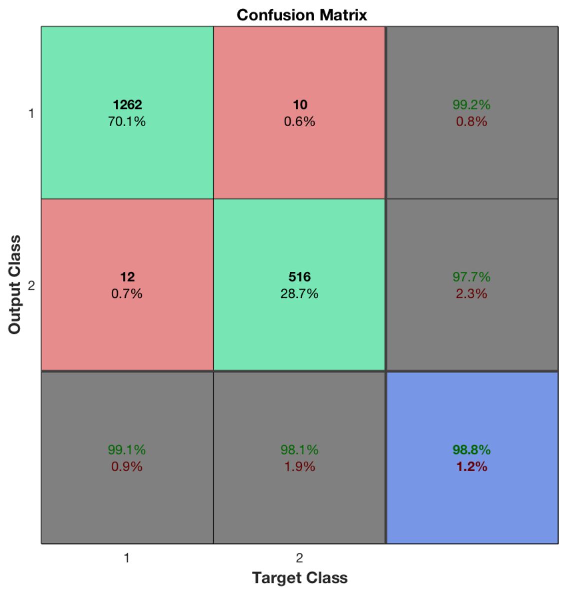


Figure 4.6: *Confusion matrix of the best patternet with 1 pattern of noise*

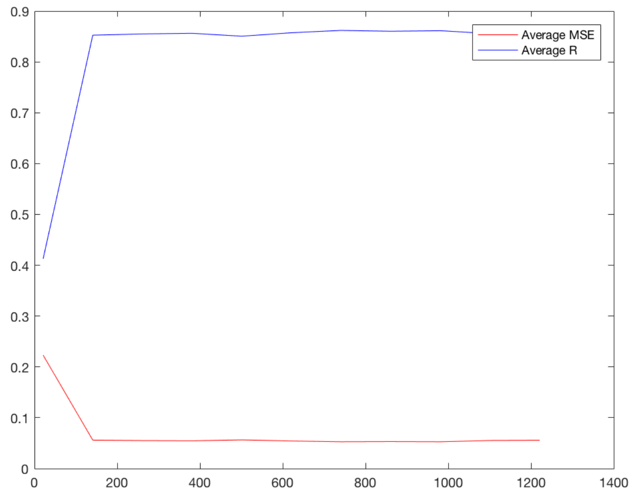


Figure 4.7: *R and MSE of RBF with 1 pattern of noise*

completely random, without reflecting the expected target. A regression value of 0.28 indicate a network that is not capable to output a reliable result. Is not possible to solve this problem using only 500 neurons, in fact the confusion matrix reveal that only 63% of samples are correctly classified.

4.4 Hardest case: 200 pattern of noise

4.4.1 MLP fitnet

The best network accordingly to the mean square error value was the one with 200 neurons. Regression plot shows a very poor capacity of the network to fit provided data. Most of time the network produce a value that oscillate around 0.5, without reflecting the expected target. A regression value of 0.62 indicate a network that is not always capable to output a reliable result. The situation maybe could result better if the dataset was balanced.

4.4.2 MLP patternet

The best network accordingly to the mean square error value was the one with 190 neurons. Also here the regression plot shows that the classification was done in approximately way. As in the previous case, the situation is acceptable considering the confusion matrix. The consideration of the previous case is valid also here. In fact the comparison between MSE/Regression of fitnet/patternet shows that when the number of pattern noise rise, a fitting network

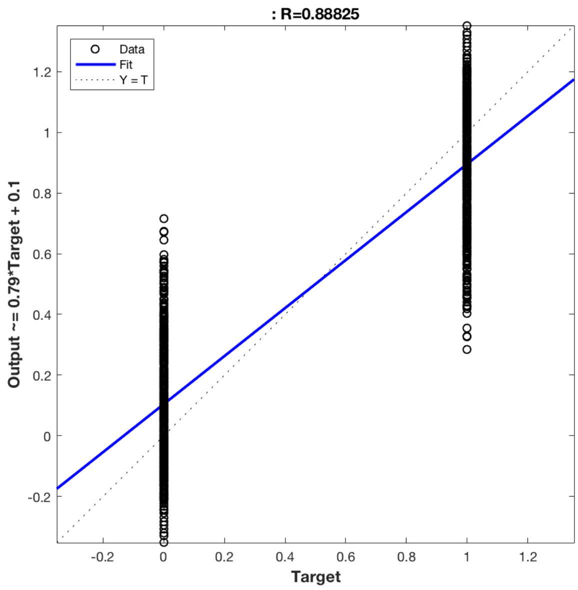


Figure 4.8: *Regression of best RBF with 1 pattern of noise*

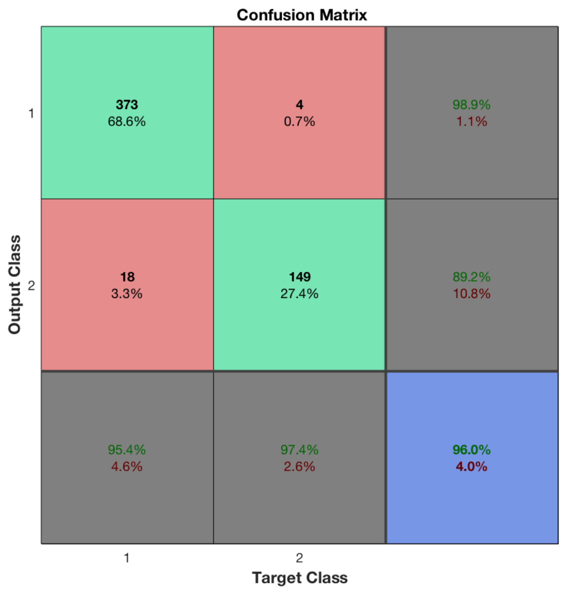


Figure 4.9: *Confusion matrix of the best RBF with 1 pattern of noise*

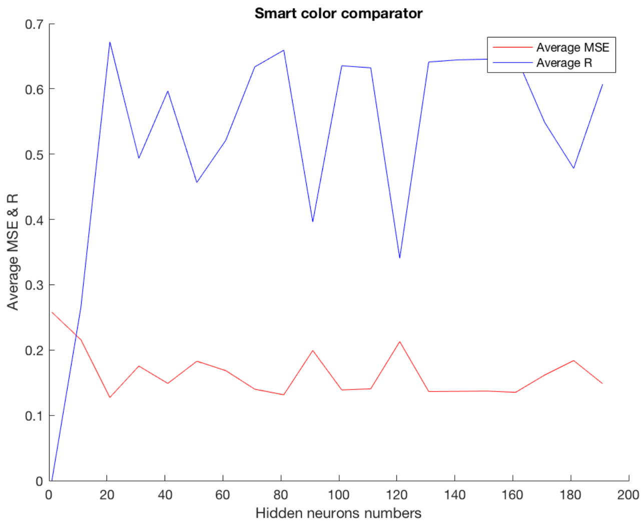


Figure 4.10: *R and MSE of MLP fitnet with 50 pattern of noise*

provide better result.

4.4.3 RBF

Also in this situation, was not possible to reach the goal of 0.05, hence the maximum number of neurons was reached in all trials. The best network accordingly to the mean square error value was the one with 500 neurons and spread 20. Regression plot shows that the network is not able at all to classify provided data.

4.5 Comparison

The comparison among the networks is done using ROC curves. Performance degrade is due to a greater number of pattern of noise. ROC curves shows that for the simplest problem, all the network get the job done, while when the number of pattern of noise increments, maybe the fitnet network is the most balanced. RBF network is not good to solve the problem when the number of pattern of noise increments over a certain threshold.

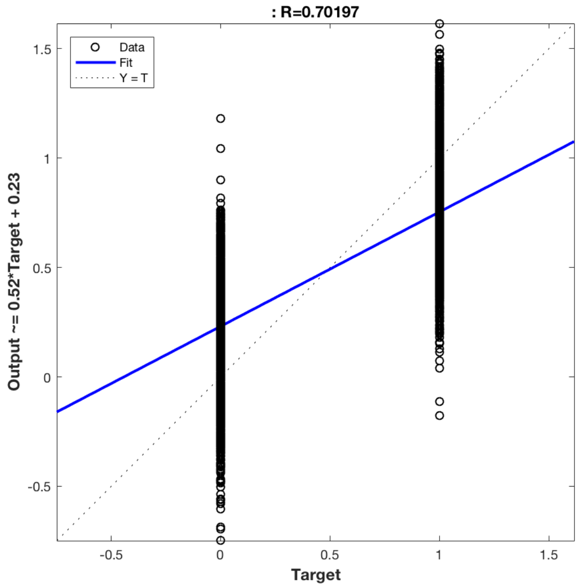


Figure 4.11: *Regression of best fitnet with 50 pattern of noise*

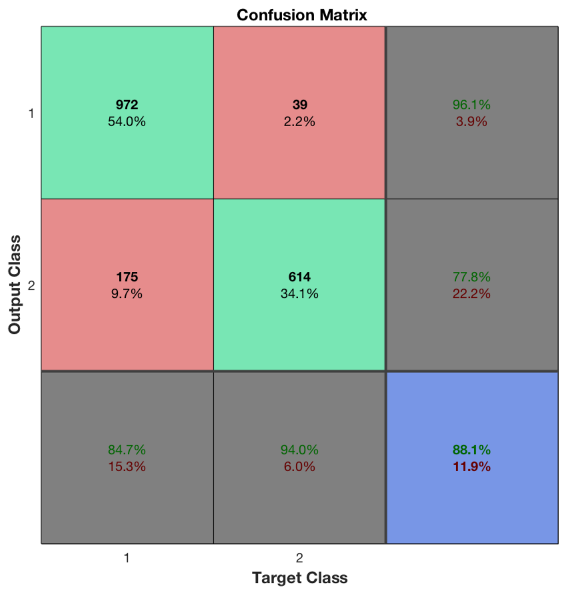


Figure 4.12: *Confusion matrix of the best fitnet with 50 pattern of noise*

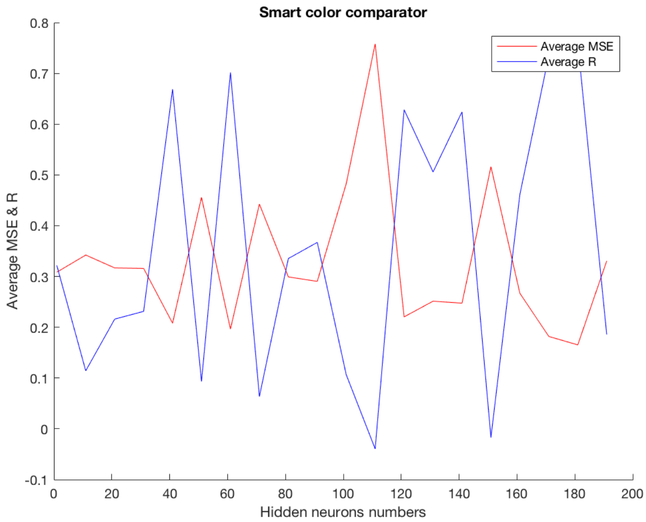


Figure 4.13: *R and MSE of MLP patternet with 50 pattern of noise*

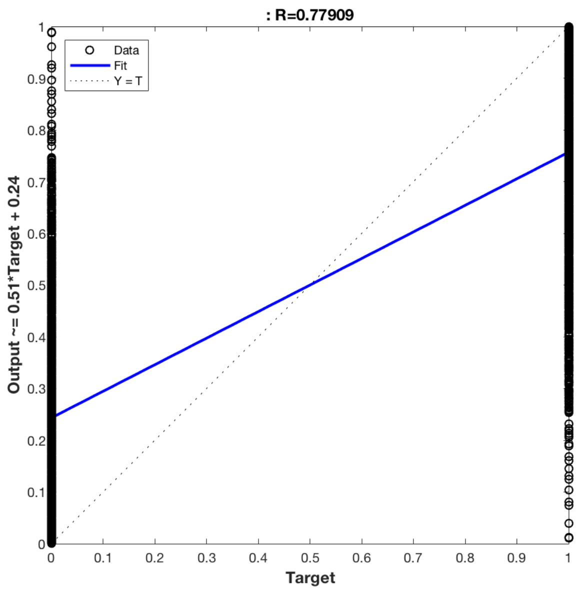


Figure 4.14: *Regression of best patternet with 50 pattern of noise*

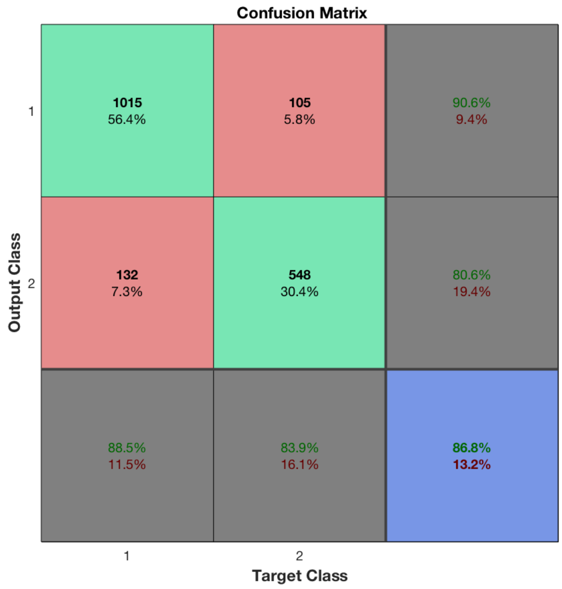


Figure 4.15: *Confusion matrix of the best patternet with 50 pattern of noise*

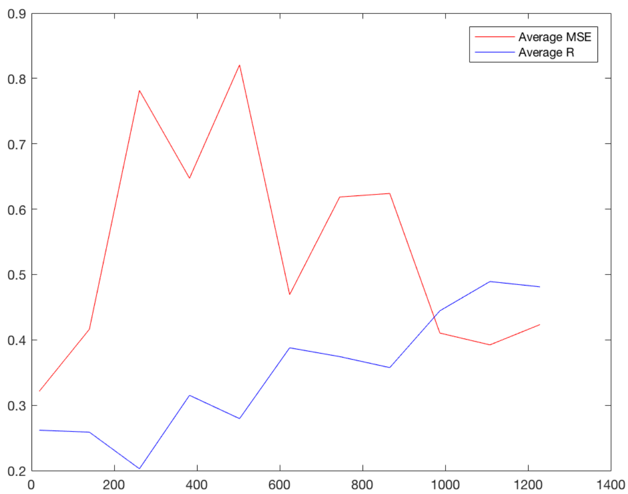


Figure 4.16: *R and MSE of RBF with 50 pattern of noise*

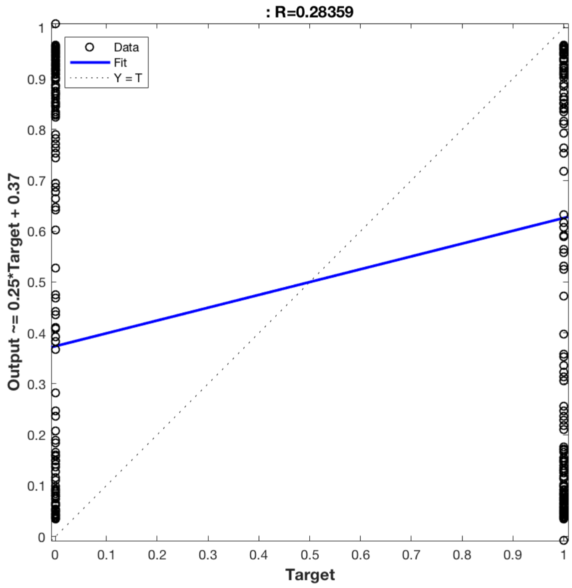


Figure 4.17: *Regression of best RBF with 50 pattern of noise*

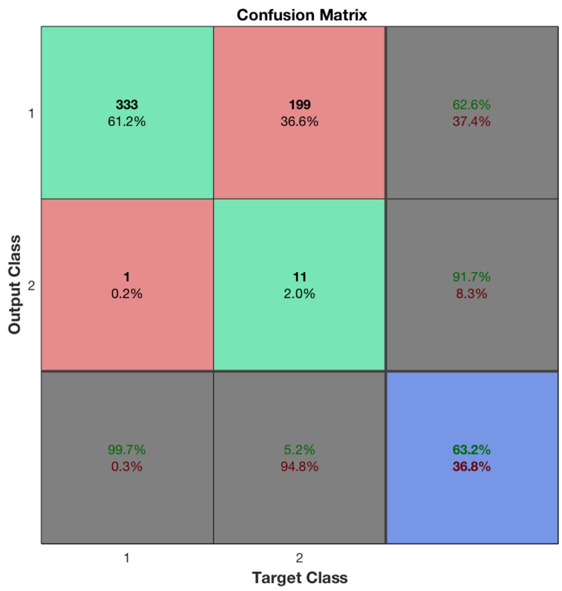


Figure 4.18: *Confusion matrix of the best RBF with 50 pattern of noise*

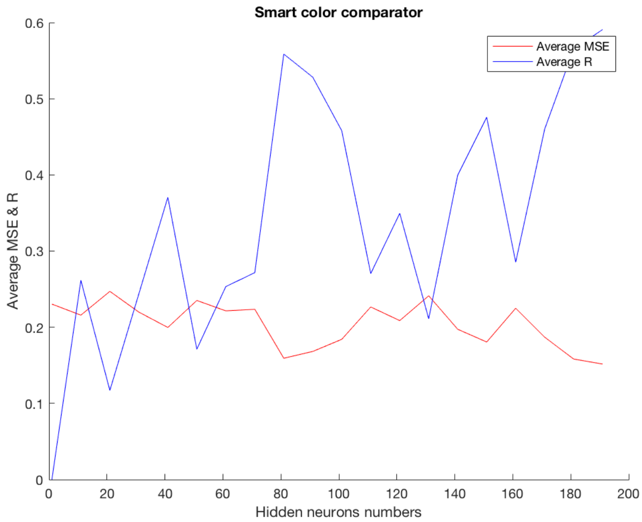


Figure 4.19: *R and MSE of MLP fitnet with 200 pattern of noise*

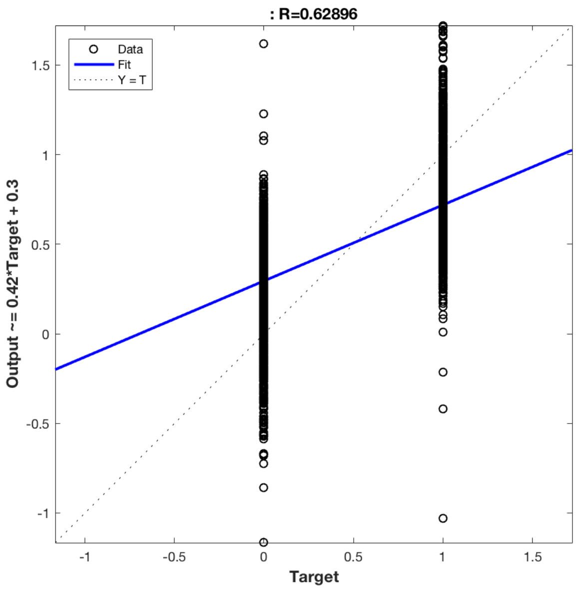


Figure 4.20: *Regression of best fitnet with 200 pattern of noise*

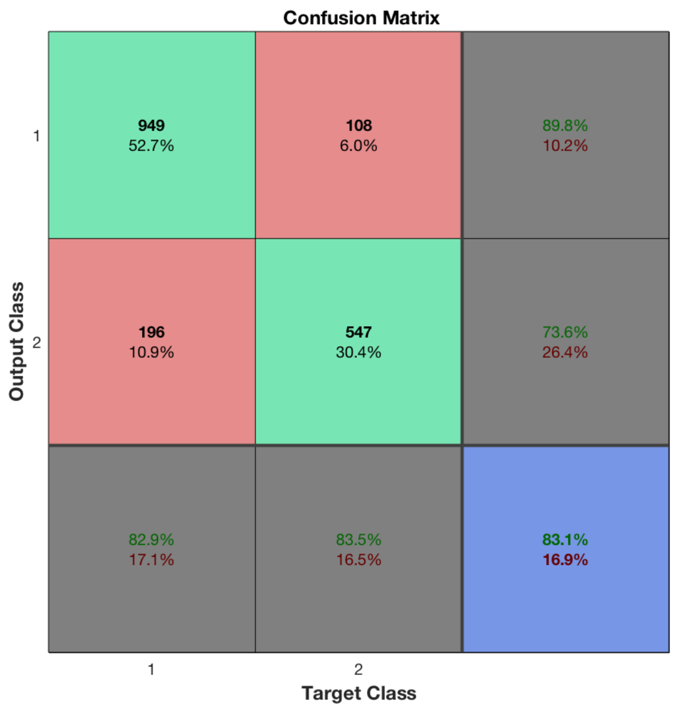


Figure 4.21: *Confusion matrix of the best fitnet with 200 pattern of noise*

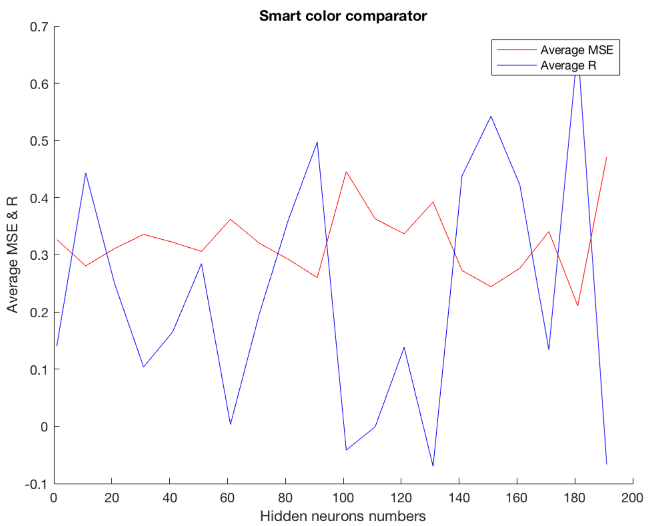


Figure 4.22: *R and MSE of MLP patternet with 200 pattern of noise*

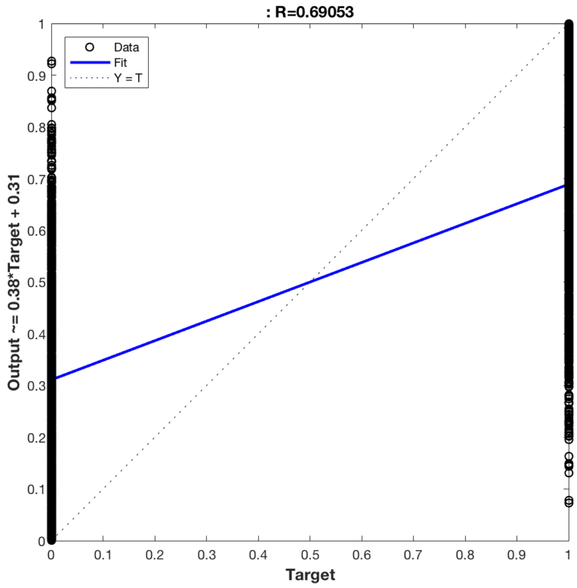


Figure 4.23: *Regression of best patternet with 200 pattern of noise*

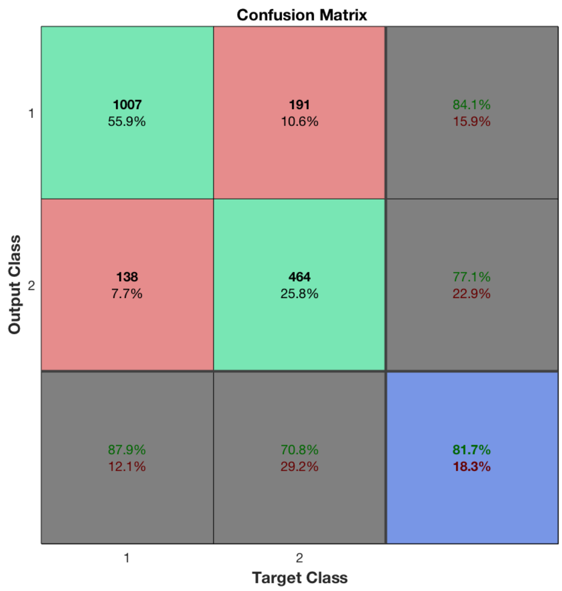


Figure 4.24: *Confusion matrix of the best patternet with 200 pattern of noise*

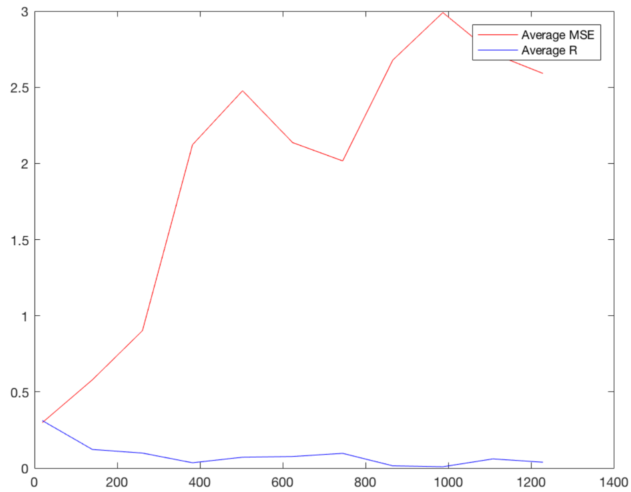


Figure 4.25: *R and MSE of RBF with 200 pattern of noise*

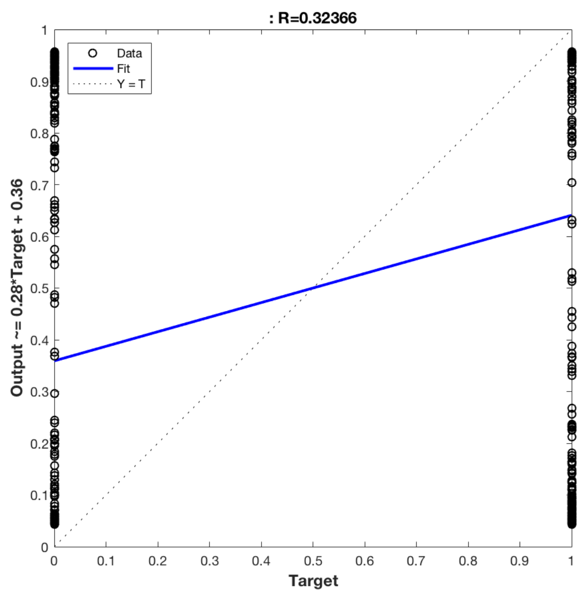


Figure 4.26: *Regression of best RBF with 200 pattern of noise*

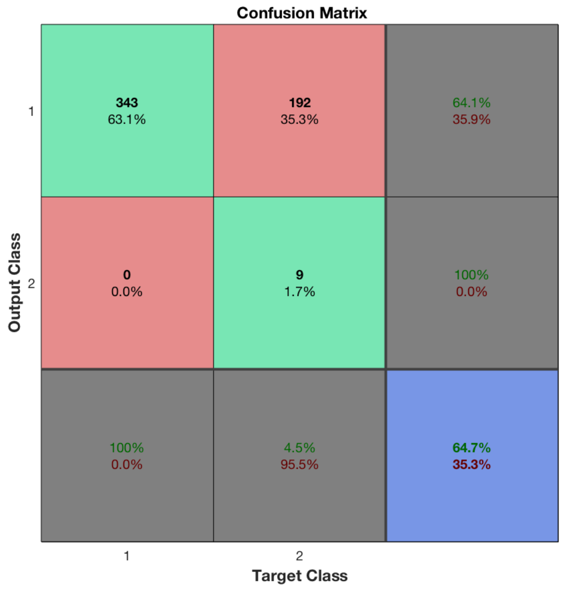


Figure 4.27: *Confusion matrix of the best RBF with 200 pattern of noise*

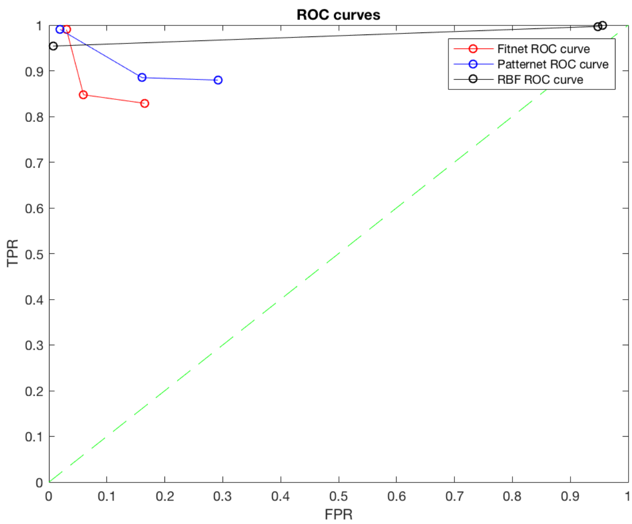


Figure 4.28: *Comparison of all network using ROC curves.*

5. Conclusion

It is not easy to conclude this work because there are so many insights. At first is fundamental to remark the results derived by the previous chapter. As expected the capacity of the network to give a good output after the training phase depends mainly by the number of pattern of noise applied to the copies. Another important result is that, for this kind of problem, the MLP networks are faster to train and results are better, maybe because the RBF it is not able to generalize well. The third conclusion is that a fitnet give better result in presence of more pattern of noise while, with only 1 pattern, the patternet will give result almost perfect with a relative small number of samples. This work leaves open some question that maybe will be analyzed in future works. For instance, it will be interesting to analyze the noise of a real industrial printer to understand if actually the noise is a gaussian white noise and maybe with a Self-Organized Map to check how many pattern of noise the printer will generate. Subsequently, will be interesting to understand at what point the network does not recognize anymore the difference between two colors, estimating the greatest number of pattern of noise related to the number of data provided to the network.

Bibliography

[1] S. Huges, *Caves of the Thousand Buddhas*, 1990.

[2] A. Farrer, *A guide to Understanding Color Tolerancing*, 1998.