

Detecting anomalies with random ferns

Dissertation presented by
Loïc WARSCOTTE

for obtaining the Master's degree in
Mathematical Engineering

Supervisor(s)
Christophe DE VLEESCHOUWER

Reader(s)
Anne-Sophie COLLIN, John LEE

Academic year 2018-2019

Acknowledgements

I would like to thank Mr. De Vleeschouwer for his devotion, for his advice and for the precious help I was given throughout the entire work. I would also like to thank Mrs Collin and Mr Lee for the time they have spent in reading this master thesis.

I especially thank my family and my friends for their encouragement and their support during my studies.

Contents

1	Introduction	4
2	Data Set	6
2.1	Construction of the labels	6
3	Preliminaries of anomaly detection	9
3.1	Classification function	9
3.2	Computation of the performance	9
3.3	The computation in practice	11
4	Basic Random Ferns	12
4.1	Basic Principle	12
4.2	Performance of basic Random Ferns	15
4.2.1	Analysis of S	16
4.2.2	Analysis of M	16
4.2.3	Fix M and S	17
5	Improved Tests	20
5.1	Generalization of the principle	20
5.2	Gradient Test	21
5.3	Cut Test	23
5.4	Distribution of the absolute values	25
5.5	Histogram Test	25
6	Mathematical Morphology	28
6.1	Binary Morphology	28
6.2	Gray-scale Morphology	30
6.3	MM before tests	30
7	Optimization method for the adaptive random ferns	33
7.1	Spreading of a frequency histogram and the discriminating capacity	33
7.2	Optimization problem	35
7.2.1	Global optimization	37
7.2.2	Local optimization	37
7.3	Function of narrowness	38
7.3.1	The entropy of the histogram	38
7.3.2	The value of the maximum bin	39
7.3.3	The first difference of bins	39

8	Validation of the method	40
8.1	Parameters	40
8.1.1	Single cut optimization	40
8.1.2	Choice of g	40
8.1.3	Multi-cuts optimization	44
8.2	Decision of mathematical morphology	45
8.3	Decision of the test	46
8.4	Opening on the supervised case	48
8.4.1	Defective and non-defective data available	49
8.4.2	Artificial Defective and non-defective data available	50
8.4.3	Comparison with unsupervised performance	51
9	Groups optimization	54
9.1	Mutual information	54
9.2	Combinatorial optimization problem	55
9.3	Greedy local search	56
10	Conclusion	58

Chapter 1

Introduction

More and more processes in industries are automated. The industrial vision, that can be defined as all the technologies and the methods used for inspection of industrial processes, is a part of it. This lets the industry to get some advantages. In a presence of well developed algorithms, the industry can avoid the human errors and becomes more performing, in terms of errors, but also in term of speed and financial profitability.

The purpose of this master thesis is trying to contribute to the automating of the industry in the context of the computer vision. So this work will explain some improvements on a pre-existent algorithm that can be brought.

In a general way, the production of an industry is inspected by a camera, that sends the images to a computer. Then specific algorithms on this computer analyze the images, and must warn the workers in the company if the objects are defective. If this is the case, the production line or a worker can decide to put it in the trash.

The purpose of these algorithms is being able to highlight the anomalies in the production. It is easy to understand this job is very specific : the computation must be fast and accurate, in order to follow the production in live and to not waste correct materials or keep defective ones.

In order to save time, the task may require using unsupervised learning. Indeed, the task of saying to the computer if the image is defective or not can be very laborious for thousand of objects, because a human must take the images one by one and identify if the image has a problem or not. So instead of getting an algorithm that must be fed with some well-known examples, i.e in knowing if the images are defective or not, we want to have an algorithm that can understand alone the global representation of a normal object, and that is able to separate them from the objects that it judges as defective. We will see during this work that this assumption is very strong, and it drastically reduces the performance of well-known algorithms. The final goal is then to produce an algorithm that is able to get closer to the performance of a supervised method.

For this purpose, we need some data set in order to test, train and validate our methods. Some real data sets are provided. The opportunity to work on real ones allows to be faced with concrete anomalies and to bring some new ideas of improvement. In particular, the data sets used to develop the algorithm is composed of several pharmaceutical pills. As we will see on the chapter dedicated to the presentation of the data sets, the several sets presents very different kinds of structures, which is important in order to test our method in multiple types of contexts. This allows us to find algorithms which work on a very large number of different data sets.

The procedure will be to start with a quite well-known algorithm and we will try to improve it in order to get better performance of separation between good and defective objects. With

these specifications, a good starting point can be the random ferns. These ones are based on assumptions that make them very fast, and offer good results in supervised learning and in the context of classification ([13]). It will be seen that anomaly detection in unsupervised learning gives bad results with this method, and this highlights some possibilities of improvement.

After that, we will see that the feature that is used in the basic algorithm is not suitable for anomaly detection. Then, some new ones will be introduced in order to try to improve the performance of the classification. In particular, we will notice that a test can work better on a certain type of data.

Then an important point is the choice of which feature the algorithm must apply with regard to the data that we propose to it. This question is a try of making the detection algorithm adaptive. The difficulty of this part is the unsupervised assumption : which test must be used if we do not know its performance during the training phase?

This question has a big impact on the usage of the program, since it allows the user to not be an expert in anomaly detection in order to use the good test.

The final purpose of this master thesis is to develop an algorithm based on the random ferns, but that can be applied to the anomaly detection. This extended algorithm allows us to work well with very different type of data sets, and has the big advantage of being independent, i.e there is no need of expertise to use this algorithm in the best way. We will see that the unsupervised adaptation gets closer the the supervised case in terms of performance.

From a more practical point of view, the programming language during the whole work is C++ with the library OpenCV. This library provides powerful and pre-programmed tools that will be very useful. The documentation on this library comes from [1].

Chapter 2

Data Set

As explained in the introduction, the used data set is composed of three kinds of pharmaceutical pills : magnesium, ginseng and mint. There are three sub-types of states per pill : good ones, contaminated ones and cracked ones. All the development of the algorithm is based on these ones, because as it can be seen in the figures from 2.1 to 2.9, the data sets hold very different kinds of anomalies, and it can be considered as sufficient to succeed on these ones, in order to hope good results in other data sets.

Through the whole report, images are considered as intensity images, i.e. in the grey-scale form because of several reasons. The first one is the fact that all the industrial vision does not have 3-colours vision system as X-rays detection, for example. The second one is the fact that the algorithm in grey-scale mode is easily switched for coloured images, it is sufficient to take the intensities of the 3 colours independently and running the algorithm on each colour, for example.

The first thing to do before the running of the detection algorithm is the creation of the sub-data set. By the way, the program changes the coloured image to a grey-scale image.

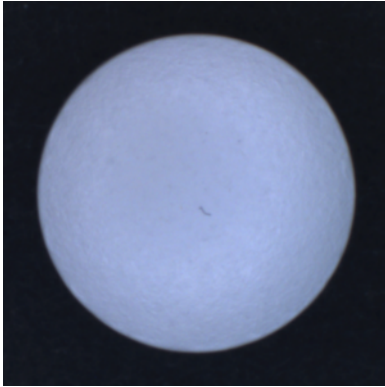


Figure 2.1

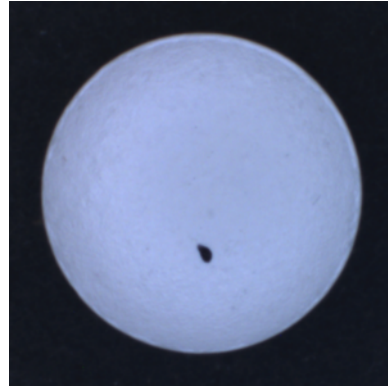


Figure 2.2

2.1 Construction of the labels

In order to know when the detection method fails or succeeds, we need to point out the anomalies manually in the images. A possible and simple way to achieve this is creating label images. A label of an image is an image of the same dimension, in which each pixel is equal to 1 if this pixel corresponds to a pixel of a anomaly, and 0 otherwise. We can see the transformation from the figure 2.10 to 2.11. We will see in the next chapter how these label images are used in order to compute the performance of our algorithm.

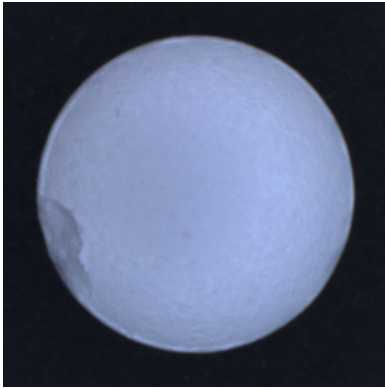


Figure 2.3

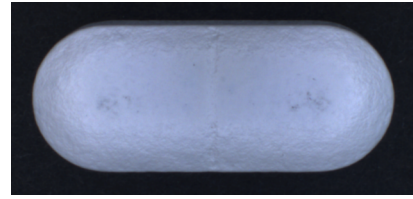


Figure 2.4

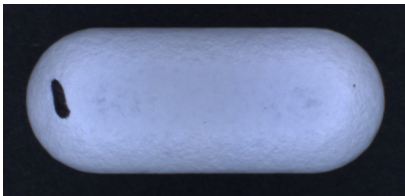


Figure 2.5

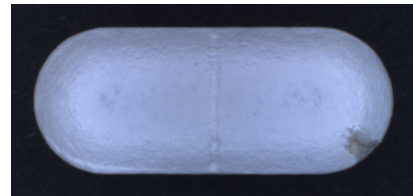


Figure 2.6

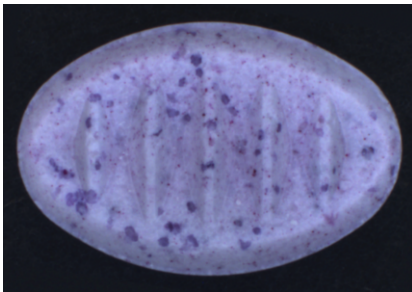


Figure 2.7



Figure 2.8

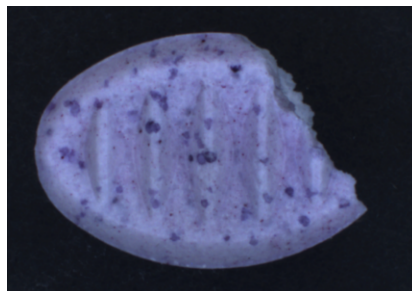


Figure 2.9

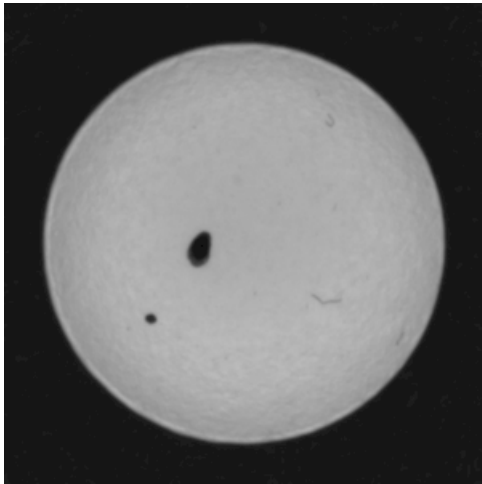


Figure 2.10



Figure 2.11

Chapter 3

Preliminaries of anomaly detection

In this chapter, we get to the heart of the subject, which is the anomaly detection. In this part, we will define mathematically the intuition of the state "anomaly" and show how the performance of a detection anomaly algorithm can be quantified.

3.1 Classification function

From a mathematical point of view, we want a function that is able to map an image x inside a set of possible images X to a binary output b , which is equal to 1 if x is an anomaly, and 0 otherwise. In practice, this function is decomposed in two steps. The first step is to define a feature, a function that turns an image x into an integer which is the output y , or the realization y of the feature, from a probabilistic point of view. We can see the visual representation in the figure 3.1. The second step is computing the frequency of the encountered realizations. If the probability of the given realization is under a threshold τ , then the image x is probably an anomaly.

3.2 Computation of the performance

During the whole report, the performance of a method is analyzed thanks to its *ROC curve*. As it was said just above, an event is considered as anomaly if the probability of this event is less than a fixed probability, let's say τ . The behaviour of a method highly depends of this fixed value, and a method 1 can have better results than method 2 for small τ , and it can be the other way around for big τ . So no single value must be considered for the comparison of results, but several τ 's.

Good criteria in order to compare anomaly detection algorithms are false positive rate (FPR) and true positive rate (TPR). A false positive is a non-defective object that is considered as anomaly for the the algorithm, and a true positive is a defective object that is detected as an anomaly by the method. On the other hand, A false negative is an defective object that is not considered as anomaly for the the algorithm, and a true negative is a non-defective object that is not detected as an anomaly by the method. So the ROC curve is the parametric curve for the parameter τ , of the true positive rate as a function of the false positive rate. From [2], the true and false positive rates can be computed as :

$$FPR = \frac{FP}{FP + TN}$$

$$TPR = \frac{TP}{TP + FN}$$

where each acronym is made of two letters, the first one describes if it is true or false, and the second one says if it is negative or positive. Therefore, the FPR can be seen as the proportion

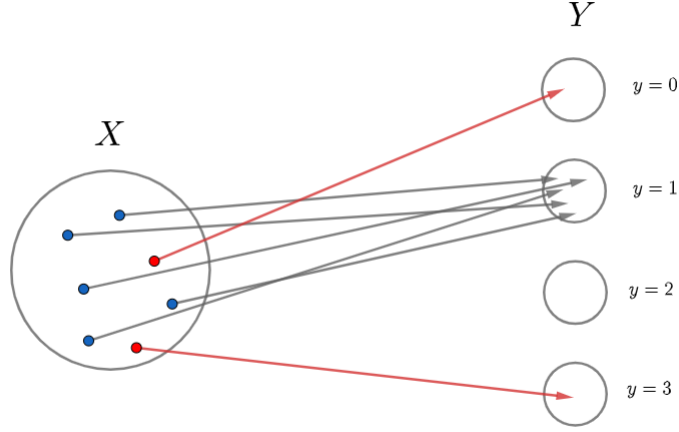


Figure 3.1

of the non-defective images that is detected as anomalies, and the TPR is the proportion of defective objects which is detected as defective by the algorithm. In order to have a single number of comparison, we can consider the AUC criterion (area under the curve) which is the computation of the integral of the ROC curve. In our case, we do not pay attention to this AUC criterion for two reasons. The first one is that it is useless in practice when we have the chart of the performance. Secondly, the AUC criterion does not take into account the evolution of the curve. As we will see later, some methods can have a really good TPR for very small FPR, but the TPR can remain constant when the FPR increases. For other methods, it can be the other way around. The method can be very poor for small FPR, but we can get quickly a TPR close to 1 with a very low FPR. The result of this is the methods will not be chosen for the same purpose. This is why we prefer to plot the roc-curve rather than computing the AUC, because it gives much more information on the performance.

Typically, for big τ , the method will be very strict : the real anomalies will be detected, but also the non-defective objects. So the FPR will be very high, and the TPR will also be very high, since almost all the defective objects will be detected. Especially, the point (1,1) belongs to the curve.

For small τ , the method is very lax, so few good objects will be detected as anomalies, but the relaxation can make the algorithm forgets some errors. This is why for small values of τ , both the FPR and the TPR will be low. Especially the point (0,0) belongs to the curve.

Indeed, a perfect method has a curve like a step function : the point (0,0) belongs to the chart, then when we decrease the value of τ , the number of detected anomalies increases, so the TPR increases, but the perfect method does not increase the number of false positives, so the ideal method keeps the FPR to 0. So the perfect algorithm increases the TPR to 1 while the FPR stays to 0, so the point (0,1) belongs to the chart too. An example can be seen in the figure 3.2. In practice, this never happens as well.

The figure 3.3 shows a common roc-curve that we can expect.

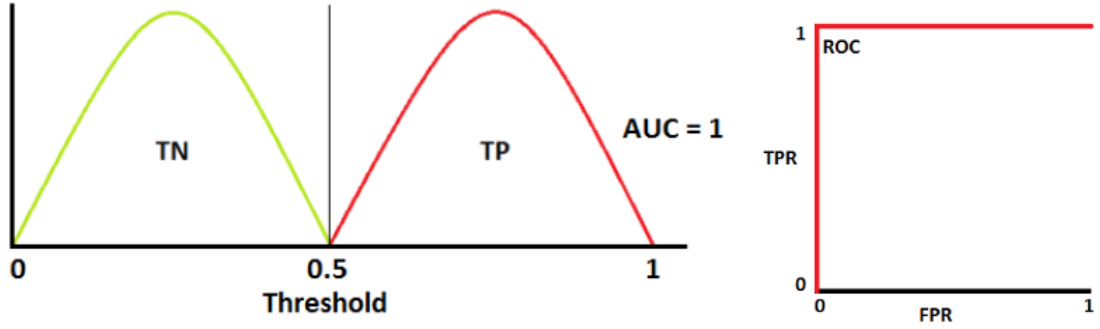


Figure 3.2: [2]

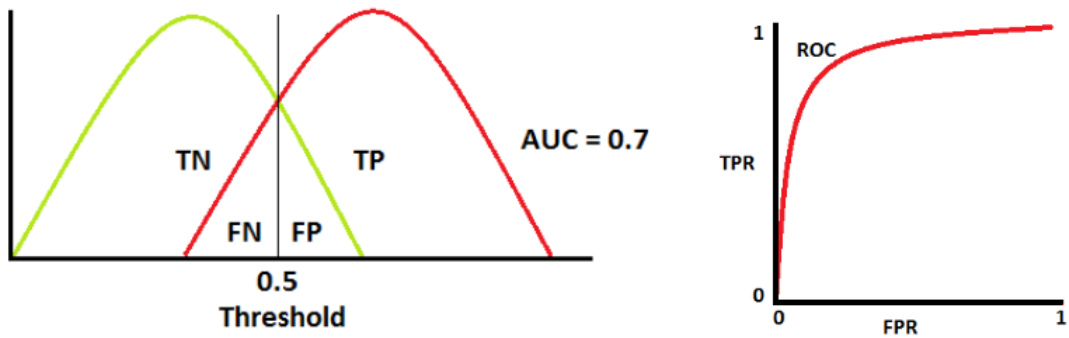


Figure 3.3: [2]

3.3 The computation in practice

In order to compute the different rates and then the roc-curve, we need to know where are the anomalies in the images. In order to do that, we can use the label images that we introduced in the previous chapter.

The decision of the algorithm is compared with presence or not of a white pixel in the label image. The practical computation of the 4 quantities is explained below. We define two boolean values : *sayError* which is equal to 1 if the algorithm decided to classify the image as an anomaly and 0 otherwise, and *isError* which is equal to 1 if there is at least a white pixel for the current image. This can be efficiently computed with the openCV method `minMaxLoc`. We start with the 4 values FP, TN, TP, FN at 0. Then, for each testing image, we increment these 4 values such that :

```

FP += sayError & & !isError
TP += sayError & & isError
FN += !sayError & & isError
TN += !sayError & & !isError

```

Then, the FPR and the TPR are computed as described at the beginning of this section.

Chapter 4

Basic Random Ferns

In this section the basic principle of the random ferns will be explained. The assumptions that we need for the construction of this method are explained. We also notice the limitations of the random ferns and the possible explanation of these phenomena. This chapter is mainly inspired from [13], [7] and [9].

As said in the paper [13], the random ferns are a very useful alternative to some other more well-known algorithms, as random trees. Its particularity is their reliability to environment variation, as such as a change of brightness during the capture of the image, for example. With most of other tools, these types of variation must be handled with pre-processing. This pre-processing must not only be applied during the training, but also during the classification of new images, i.e. in live. For our case, this is a very important point. Now, we will introduce the mathematical background of the random ferns' algorithm.

4.1 Basic Principle

Firstly we must describe the main component of them : the tests. A test is a random variable T in order to characterize an image. A test is a probabilistic realization of a *feature*. Here we speak about random variables, because each value of a test in an image can be seen as a realization for this random variable. In our case, these tests are binary, because any test depends on a comparison of 2 pixels of the image, such that each realization of a test is either 0 or 1, according to the result of the comparison. More precisely, one can write the test in this way :

$$T = \begin{cases} 1 & \text{if } I_1 < I_2 \\ 0 & \text{otherwise} \end{cases} \quad (4.1)$$

Where I_1 and I_2 are the intensities of the 2 compared pixels. Typically the choice of which pixel to choose is totally random, during a phase before the training phase, which is called initialization phase in this work. It is important to emphasize that this phase is completely independent from the training phase and the test phase.

Then, we can describe what a fern is. A fern is simply a set of these binary tests. The number of tests in a fern will be denoted as the number S in this work. So one can see the realization of a fern as a sequence of S binary numbers : $F = \{0, 1\}^S$. However, in this report F will be expressed in the decimal base instead of the binary base.

The bijection is performed by a simple computation :

$$f = \sum_{i=0}^{S-1} f_b[i] \times 2^{S-1-i}$$

where f is the value of the fern in the decimal base, $f_b[i]$ is the value of the binary sequence f_b at the 0-based index i . Moreover it can be easily verified that $f \in \{0, \dots, 2^S - 1\}$.

K can be defined as the number of different realizations that F can take, i.e $K = 2^S$. The benefit of the integer representation is mostly practical and algorithmic : firstly it is more suitable to write an integer than a sequence of binary, and secondly the value of a fern is used as index of an array in the code. One can observe that F is also a random variable for the same reasons as T .

During the learning phase, the random variable F is realized once per image, thanks to the tests inside this fern. So, each test in the fern will be evaluated for each couple of pixels at each image and the fern F is assigned a value between 0 and $K - 1$, which is represented by f as the realization of the fern.

Then, at each image, the value of F is put in a frequency histogram h of size K , such that each entry is incremented of +1 at each time the value of the histogram is realized. We can emphasize that the value of the i^{th} entry of histogram h is denoted as $h[i]$, in the 0-indexed base. So, when the realization f is encountered, we do $h[f] += 1$.

As we will see later, the shape of the histogram is a key indication of the behaviour of the performance, and it is often the better way to improve or to justify the behaviour of the algorithm.

When the training phase is done and the histogram is complete, a new image (a testing image) can be presented to the algorithm and it will compute the realization of the same fern that in the training phase. Intuitively, if the algorithm does not see the value of the realization very often during the training phase, there is a high probability of being an anomaly. Rigorously, the algorithm computes the probability of appearance of that realizations as :

$$\hat{P}_{\text{naive}}[F = f] = \frac{h[f]}{N} \quad (4.2)$$

where $h[f]$ is the value of the histogram at that realization and N is total number of training images.

However, this approximation can be very problematic. Indeed, if we imagine the case where few training images are presented for example, the probability can be 0 very often. It is therefore more appropriate to slightly add a bias to this estimation like this :

$$\hat{P}[F = f] = \frac{h[f] + 1}{N + K}$$

It can be observed that it slightly changes the value of the probability, but no probability is put at 0 in this way. The interest will be very clear after the following lines.

In theory, the precision can be improved by increasing S : more tests are used, and more pixels are compared. However, there are 2 big problems which come from that the number of possible realizations is exponential compared to S , since $K = 2^S$. The first one is due to the curse of dimensionality : the number of possible values can overpass the number of training images (or simply be too big compared to it) and the histogram can become very non-representative. The second problem is the data complexity : the number of values to store is therefore exponential compared to the number of tests and it can be a problem for a system which can be limited, for example.

The idea to fix the problem is to take several ferns of S tests. This number will be denoted as M . So instead of generating S tests, $M \times S$ tests are created at random, and we independently apply the tests of each fern on the images. So each fern will give a vote, which is the decision of the fern to say that an image is an anomaly or not. Finally, we combine the different votes of the M ferns in order to give a global decision D .

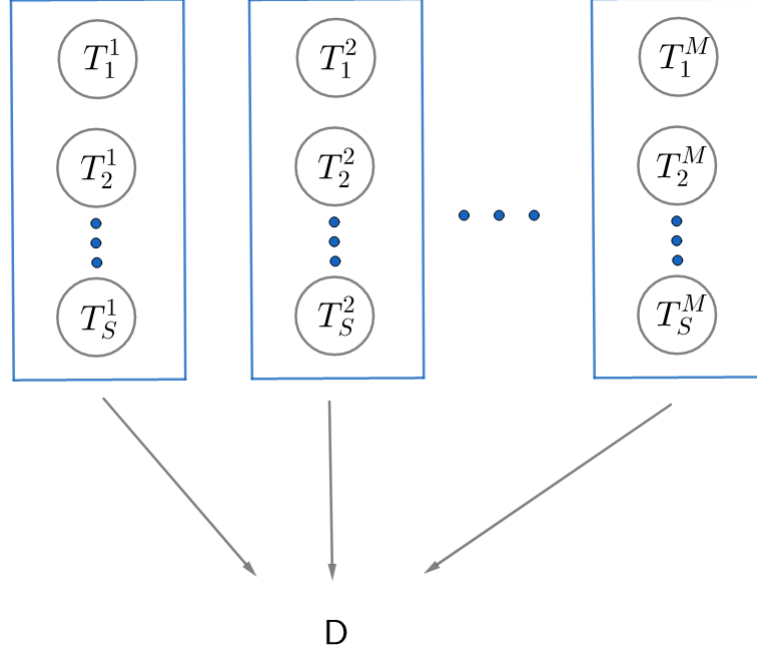


Figure 4.1

We can extend the definition of the random test in the equation (4.1) with T_i^m , which is the test i in the fern m . We can see the representation of the M ferns in the figure 4.1, where D is the jointed decision of the M ferns.

Now, the current question is the way to combine the M votes. The idea is to consider the assumption that the ferns are completely independent, that which is not really true in practice. So, instead of constructing one big histogram, M small histograms h_m are computed independently. F_m can be defined as the random variable for the m^{th} fern. Therefore we can express the probability for each fern :

$$\hat{P}[F_m = f] = \frac{h_m[f] + 1}{N + K}$$

Where f is the realization of the m^{th} fern.

Then, each probability of realization of each fern is computed, and thanks to the naive bayesian rule, the estimator of the probability of an image is the product of the estimators of each probability on each fern, such that :

$$Pr[F_1 = f_1, \dots, F_M = f_M] = \prod_{m=1}^M \hat{P}[F_m = f_m]$$

Now we can see why the added bias is very important. If we define the joint probability like this, only one fern can make the global probability at 0 without bias. This is very bad, because it is the same that giving an infinite weight to this fern, and the decision D is not global anymore over the M ferns.

Having several ferns involves the decrease of the data complexity from $2^{M \times S}$ to $M \times 2^S$.

We can see the random ferns as a relaxed version of random tree. The big difference here is the type of the used structure : in the first case, the features are organized as a tree while in our case, the features are considered as non-hierarchical way.

As explained in [13], the ferns have a local behaviour, i.e the frequency histograms must refer to a small region of an image. A sub-image is called a *local area*, and each sample of the image at a local area is defined as a *patch*. In practice, the images are therefore split into patches, it is more interesting to make them overlapping, in order to have patches which are as independent as possible compared to the cutting of the image. In practice, the splitting is done for each of the two dimensions of an image. The number of patches is noted P . The algorithm is summarized below. So, we do not have M frequency histograms h_m , but $M \times P$ frequency histograms h_m^j for the local area j and for the fern m .

Now, we are able to write the computation of the estimation of the probability for each local area :

$$Pr[F_1^j = f_1^j, \dots, F_M^j = f_M^j] = \prod_{m=1}^M \hat{P}[F_m^j = f_m^j]$$

Where :

$$\hat{P}[F_m^j = f] = \frac{h_m^j[f] + 1}{N + K}$$

Then, when we encountered a new image, its patch j is detected as an anomaly if :

$$Pr[F_1^j = f_1^j, \dots, F_M^j = f_M^j] \leq \tau$$

We can summarize the different notations that we saw until now, in order to make it more clear for the next chapters.

Abbreviation	signification
S	Number of test per fern
M	Number of ferns
$K = 2^S$	Number of possible realizations for a fern
P	Number of patches in an image
N	Number of training images
T_i^m	Random variable of the test i for the fern m
F_m^j	Random variable of the fern m for the zone j
h_m^j	Frequency histogram of the m^{th} fern for the j^{th} zone
τ	Probability threshold of being an anomaly

4.2 Performance of basic Random Ferns

In this section the performance of the very basic algorithm is analyzed. This chart can be drawn for the basic random ferns, and the behaviour of the methods can be discussed compared to its parameters : number of ferns, number of tests per fern. The basic parameters are :

- $S = 6$
- $M = 10$
- $P = 25$

Algorithm 4.1 Basic Random Ferns : Summary

Require: M, S, τ, P, N Generate randomly M ferns of S tests**Learning Phase :**

```
for ( $j = 1; j \leq P; j++$ ) do
  for ( $m = 1; m \leq M; m++$ ) do
    Construct  $h_m^j$ 
  end for
end for
```

Testing Phase : F_m^j the fern variable for the new image for the fern m and the zone j

```
for ( $j = 1; j \leq P; j++$ ) do
   $Pr[F_1^j = f_1^j, \dots, F_M^j = f_M^j] \leftarrow 1$ 
  for ( $m = 1; m \leq M; m++$ ) do
    Compute  $f$  such that  $F_m^j = f$ 
    Compute  $Pr[F_m^j = f]$  with  $h_m^j$ 
     $Pr[F_1^j = f_1^j, \dots, F_M^j = f_M^j] \leftarrow Pr[F_1^j = f_1^j, \dots, F_M^j = f_M^j] \times Pr[F_m^j = f]$ 
  end for
  if  $Pr[F_1^j = f_1^j, \dots, F_M^j = f_M^j] \leq \tau$  then
    patch  $j \leftarrow$  anomaly
  end if
end for
```

4.2.1 Analysis of S

As reminder, S is the parameter that corresponds to the number of tests that a patch contains. In the three figures below, we can see the roc-curves for each pill, and for three different values of S . The conclusion that we can draw for the parameter S is that the bigger is the parameters S , the better is the anomaly detection. However, this improvement is quite small, and there is quickly a saturation phenomenon, as we see it in the figures from 4.2 to 4.4. In fact, this phenomenon is quite intuitive. The number of tests directly refers to the accuracy of the binary representation of an event, i.e it is the number of bits that an event can use to be represented. So, in a data set of quite good quality, i.e the non-defective data set has more or less the same binary representation, taking a big number of tests is useless, because we add non-used possible binary representation. In fact, this can even become really dangerous for not really good data set, as very noisy one, for example. The reason is that each patch through the images can have a different binary representation, and making the distribution of the histogram flat, such that in a uniform distribution. So, one must have a S big enough to characterize correctly each patch, but the number must not be too big in order to not erase the quantization of the tests. We can see the consequences on the increasing of S on the frequency histograms in the figures 4.5 and 4.6.

4.2.2 Analysis of M

The conclusion of the change in M is quite similar than for S . There is a slightly improvement when M is increasing for the 3 data sets, but the performance saturates for big values of M as we can see in the figure 4.7 to 4.9. This is due to the randomness of the tests : since the tests are selected in a uniform way in the space, after a certain M , adding votes, i.e ferns, is not interesting anymore, because the gain of information is minimal.

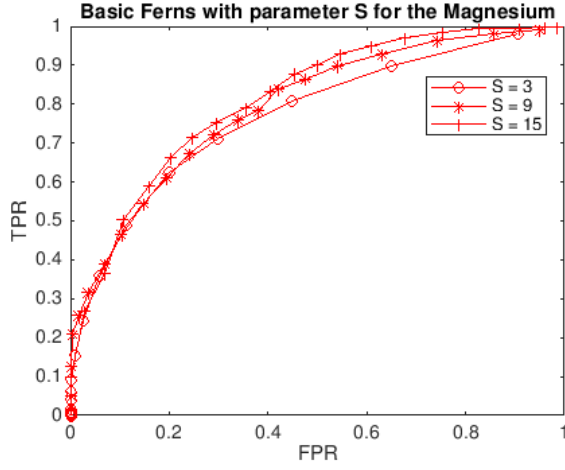


Figure 4.2

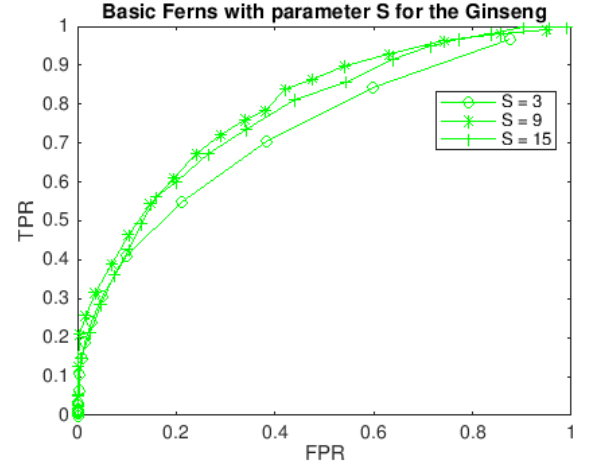


Figure 4.3

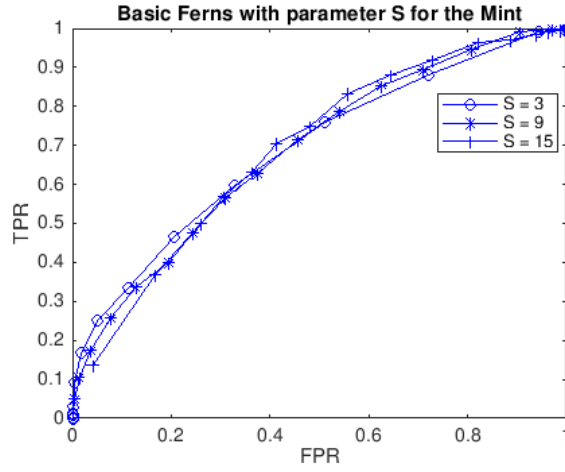


Figure 4.4

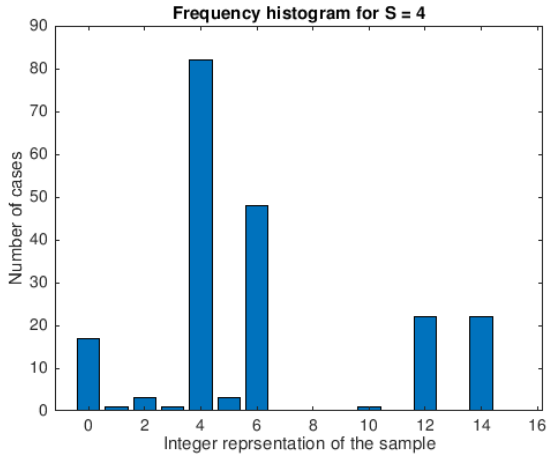


Figure 4.5

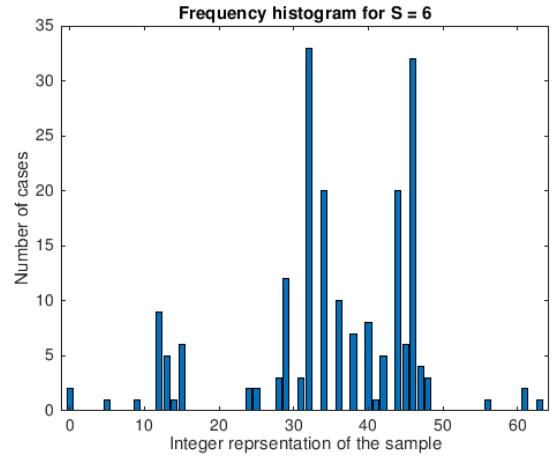


Figure 4.6

4.2.3 Fix M and S

In order to compare the methods and test them, it is necessary to keep the same S and M during the whole report. The choice is $S = 8$ and $M = 16$.

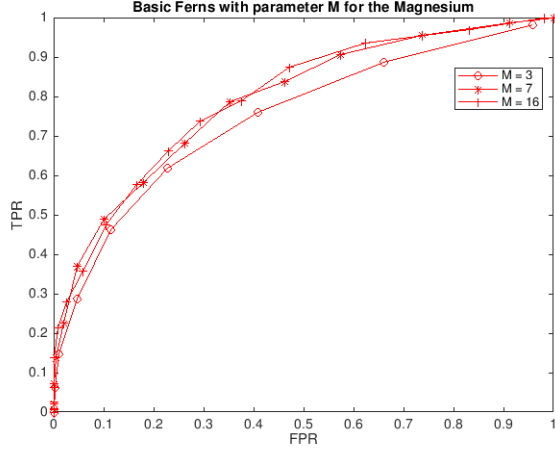


Figure 4.7

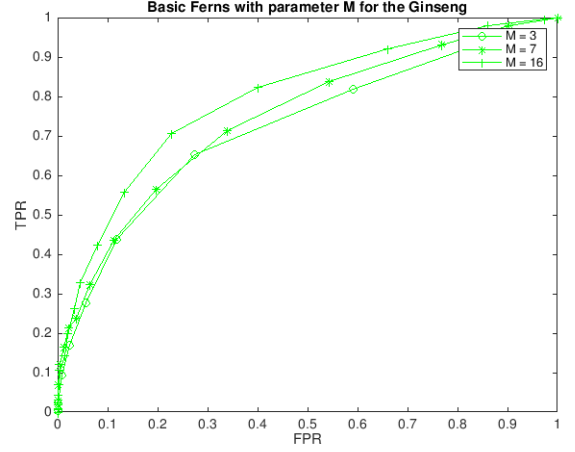


Figure 4.8

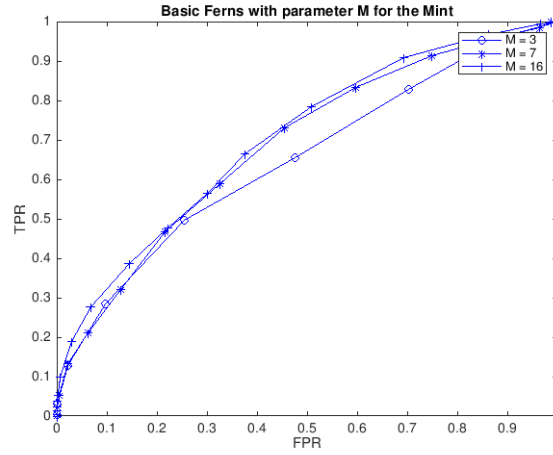


Figure 4.9

We can see the figure 4.10 of the roc-curve for basic ferns, and for each data set. These roc-curves will then be used to compare the initial algorithm with new methods.

As a conclusion of this first analysis of the performance for the basic ferns, one saw that the bigger are the number of tests and the number of ferns, the better is the performance of the ferns. However, this improvement saturates quite quickly and it is not sufficient to increase them in order to have satisfying performance. Moreover, increasing S and M increases also the data complexity. The next chapter speaks about the solution that can be brought.

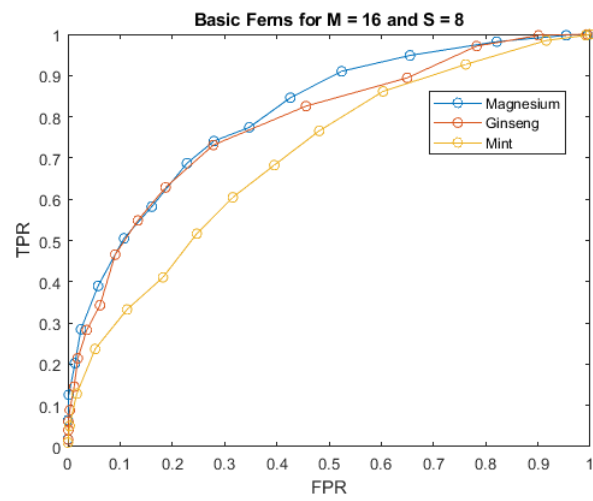


Figure 4.10

Chapter 5

Improved Tests

As it can be observed in the previous chapter, the basic algorithm of random ferns does not give very good results, mostly for the third type of data set : the mint. The reason was explained and very clear : the small purple clusters, which are randomly distributed in the pill, completely break the underlying statistics of the object. In fact, this is exactly what defines a good or a bad feature for a data set. Therefore, the goal of this chapter is introducing alternative tests, which are already well-known as the histogram of oriented gradients, or constructed in the context of this work as the cut test. Though some tests are already used for a long time, we need to slightly modify them in order to include them in the context of the randoms ferns.

The reason of investigating other tests is to find tests that can discriminate in a efficient way the anomalies from the good objects.

We will see that some tests seem to be efficient and other tests no more useful than the basic comparison test, in certain contexts. It stays important to keep them, because they can be an example of bad test, and some characteristics of these tests can be highlighted.

The data set for the mint must not be taken lightly, because it can be the behaviour of many multi-component objects. The purpose of this chapter is then generalizing the principle of random fern, in order to look for improvements of the basic algorithm. Moreover, we will see that it is especially the kind of features that is guilty, and we will try to find better features in order to improve the roc-curve of the random ferns.

Moreover, some features can be interesting or not according to the data that is used. We will try to capture the specificities of the data set. After introducing a generalization of the random ferns to make them compatible with other tests, we introduce new tests and the specificity of their behaviour.

5.1 Generalization of the principle

In order to summarize what are random ferns, we can see them as independent votes based on frequency histograms of several random binary tests on the image. However, instead of taking the binary test for the comparison of the intensities, one can think about some other tests, that may be more useful, i.e. in order to improve the accuracy of the algorithm.

Therefore, instead of using one ultimate test which is supposed to detect anomalies in every kind of data set, one can try to combine several light and specified tests.

A key observation on the anomalies of the data set is the isolation of different kind of problems. In our data set, two types of anomalies can happened : the cracked ones and the contaminated

ones. These two anomalies are essentially very different. The first type refers to anomalies which are very spread, but the difference of the intensities between the anomaly and its neighbourhood is not very high. So the criterion of detection here is the size of the supposed anomaly.

In the case of contamination, the problem is very different. the anomalies are only spread over few pixels, but the difference of the intensities between the anomaly and its neighbourhood is very high. In this case, the criterion of the detection will only be this difference of intensities, whatever the size of problem. Indeed, even if one pair of pixels fulfills the propriety, this local area will be detected as anomaly.

It is easily understood that this separation of anomalies is very important and almost "per essence". Indeed, there is no overlap between the two "anomaly subspaces", because the small and obvious anomalies are considered as too small to fulfill the spread propriety of the first type of anomaly, and the spread anomaly has a too low threshold compared to the second type of anomaly to be detected.

Moreover, as we will see later, the performance of a particular test really depends on the data set.

Therefore, we will present 3 alternative features that can be turned into binary tests. We will see their behaviour and their performance.

5.2 Gradient Test

This is the feature that is supposed to detect the discrete but spread defaults, in order to point out the cracks in our case. The gradient test is in fact based on the well-known histogram of oriented gradients. This HOG is a construction of a histogram from the orientation of the gradients encountered in the image. All the development below concerning the HOG comes from [3].

First, let's define the gradient in an image. Let's consider an image as a 2-D domain, with the orthonormal basis $(1, 0)$ and $(0, 1)$. The value of the function in each pixel of the image is equal to the intensity at this pixel. Therefore, the gradient of a pixel is the gradient of the function intensity at this pixel. With openCV, the two derivatives are computed separately thanks to the pre-programmed function `sobel`. This function returns the two matrices of derivatives for the 2 directions in the images : G_x and G_y .

After that, we need to transform the two cartesian matrices G_x and G_y into polar matrices Θ and G , which are defined as :

$$\begin{aligned}\Theta(x, y) &= \arctan \frac{G_y(x, y)}{G_x(x, y)} \\ G(x, y) &= \sqrt{G_y(x, y)^2 + G_x(x, y)^2}\end{aligned}$$

The dedicated function in openCV `cartToPolar` is used to apply this transformation.

Let's define the number of bins b that we want for the HOG, which is a divisor of 180. We can compute the increment d as $d = \frac{180}{b}$, which is the quantity to go from one bin to the next one in the histogram of oriented gradients. Then, each encountered angle in the matrix Θ is decomposed into the lower and the upper bound such that :

$$r(x, y) = \frac{\Theta(x, y)}{d} - \lfloor \frac{\Theta(x, y)}{d} \rfloor$$

The number $1 - r$ is then the proportion of the angle which corresponds to the lower bin and r the proportion to the upper bin. Then, we can decompose $G(x, y)$ into 2 quantities to increment

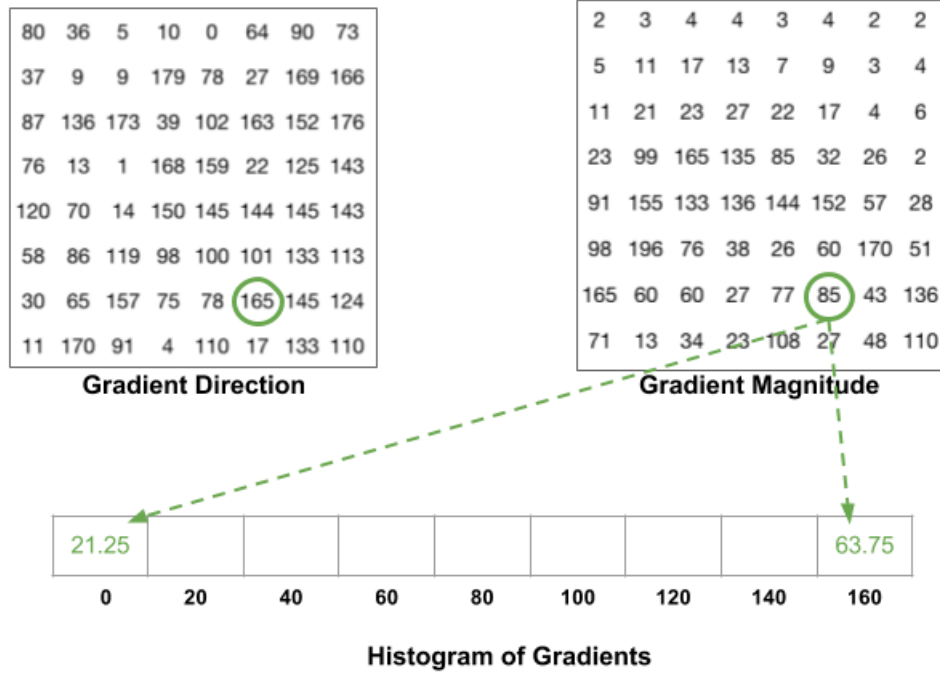


Figure 5.1: [3]

in the lower bin b_{-1} and upper b_{+1} such that :

$$\begin{aligned} b_{-1} &= (1 - r) \times G \\ b_{+1} &= r \times G \end{aligned}$$

Finally, we increment the both previous values in their respective bins of the histogram. Let's give the example of the figure. Imagine we have an angle of 165 degrees and a magnitude of 85. We made the choice of taking 9 bins, so $b = 9$ and $d = 20$. Then $r = \frac{165}{20} - \lfloor \frac{165}{20} \rfloor = 0.25$. So we put the quantity $0.25 \times 85 = 21.5$ in the bin 180 (0 in this case) and $(1 - 0.25) \times 85 = 63.75$ in 160. The example and the related figure 5.1 come from [3].

The number of bins is completely a choice. We see on the 3 following figures 5.3 to 5.5 that the bigger is the b , the higher is the performance. However, it is preferable to not go down under $d = 10$, in order to keep robust the histograms of gradients.

Now the goal is to compare these histograms, and then highlighting some trends of normality. In the case of ferns, we need to create binary tests from the HOG. A possible way to achieve it is to create binary tests that compare the bins of the HOG. Instead of choosing random pixels in the basic ferns, we can choose random pair of bins in the HOG for each fern, and make a simple binary comparison such that

$$T_g = \begin{cases} 1 & \text{if } hog[i] < hog[j] \\ 0 & \text{otherwise} \end{cases}$$

We can see a representation of this test, for the HOG in figure 5.2. The 3 tests make a fern, which can be seen as a realization for each fern. Then we can as before compute frequency histograms of the realizations of T_g . The figures 5.3 to 5.5 gives the performances of the gradient test for the three data sets, and for 3 values of b . As we see, the bigger is b , the higher is the performance for this test.

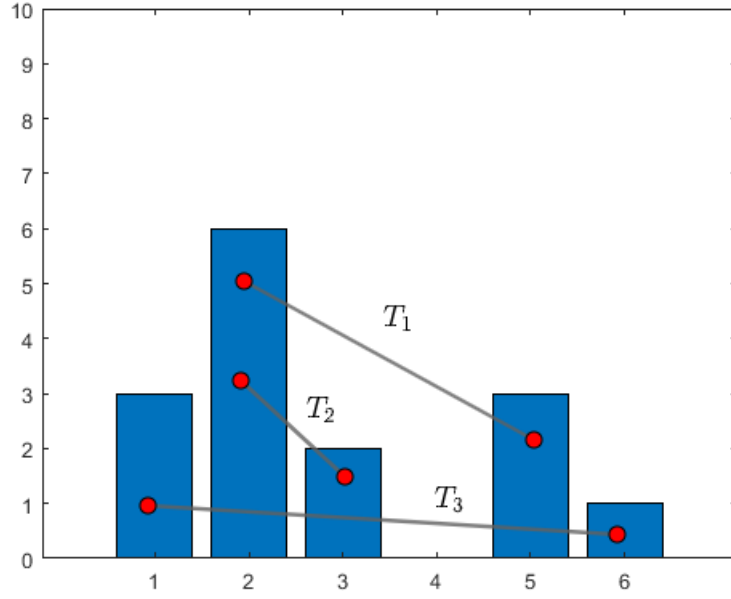


Figure 5.2

Moreover, we can observe a good improvement for the magnesium, a quite good improvement for the ginseng and no improvement for the mint. As we can see, the performance of this test is very dependent of the data set. The reason is simple: it is due to the complexity of the object. The magnesium pills are very smooth and this is exactly the test that is perfect for this type of data set. But when we use data set with granular objects, the gradients begin to become uncertain and more spread.

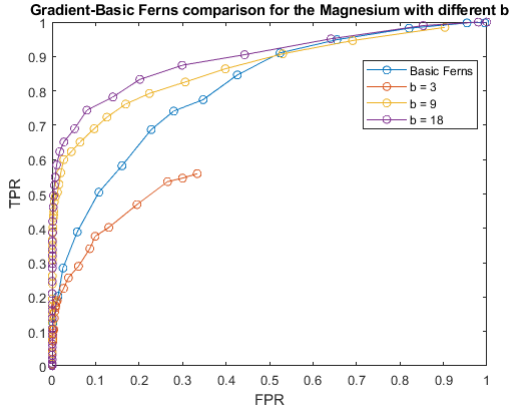


Figure 5.3

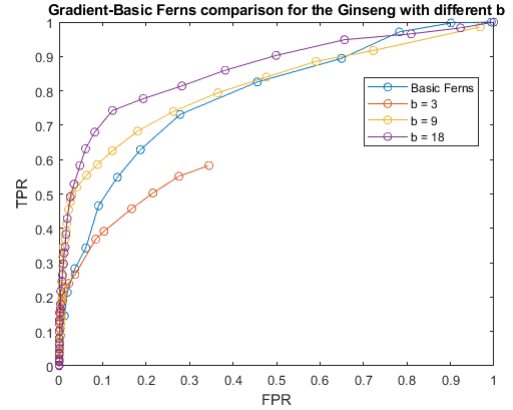


Figure 5.4

5.3 Cut Test

This is the feature that is supposed to detect the straight cutting of an image, in order to point out the contamination. As reminder, we define this type of anomalies as a big difference of the intensities of two pixels in a local area. So the basic test of comparison is just slightly modified

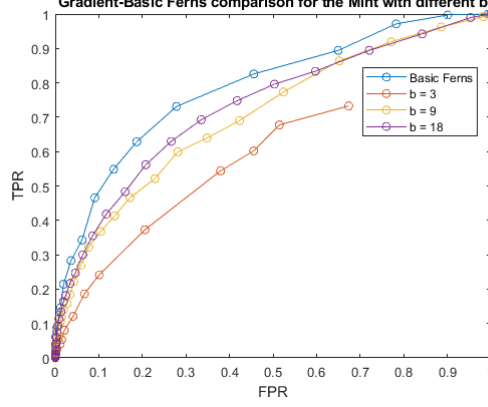


Figure 5.5

in order to get the new one as :

$$T_c = \begin{cases} 1 & \text{if } |I_1 - I_2| > C \\ 0 & \text{otherwise} \end{cases}$$

The cut-test is then 1 if the difference of the intensities of the pair of randomly selected pixels is bigger than C , where the last one is a parameter that needs to be fixed in an clever way. In the following figures 5.6 to 5.8, we see the roc curves for the 3 data sets, where each curve represents the performance for one C .

First we can observe that there is a really big difference of performance when C varies, and this optimal value is potentially different for the different data sets. There is then a huge benefit to try to get automatically the best value of C . We see how to fix this problem later in this work. The cut test seems anyway an improvement, because the best cuts gives better results than the basic ferns.

The second remark is that, though the mint data set is "harder" to deal with, the performance seems better than for the 2 other ones. It can be explain by the kind of anomalies in the mint. In fact, whatever the crack or the contamination, the anomalies are always well visible, i.e. there is always a big difference of intensities for this one, which is not the case for the two other sets. So, though the data set is more complicated, the anomalies are much simpler.

We can observe that this test is very stable compared to the data set, even if this feature is a little bit less good than the gradient test for the magnesium.

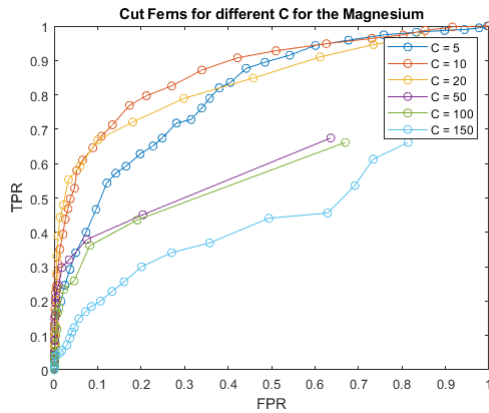


Figure 5.6

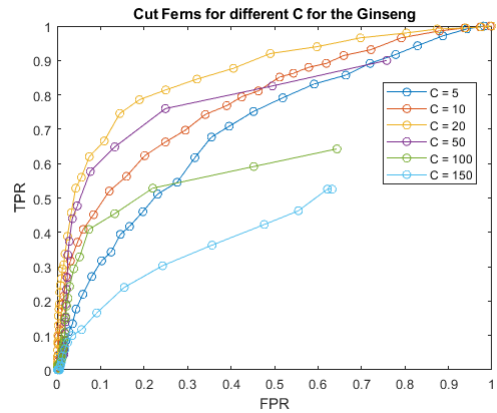


Figure 5.7

In order to try to explain why the value C that gives the best performances is dependent of the data set, the next section will focus on the distributions of the all absolutes values $|I_1 - I_2|$.

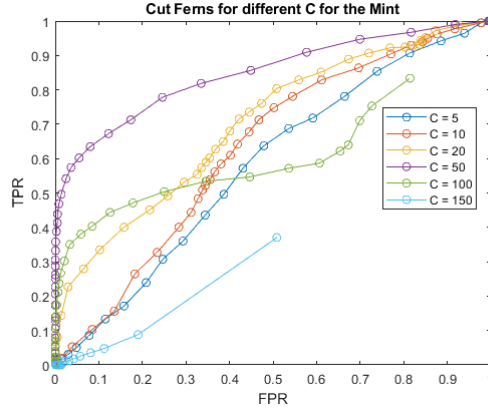


Figure 5.8

5.4 Distribution of the absolute values

The first thing to do is getting a first preview of the behaviour of the histograms as a function of the value of C . When C is very small, i.e close to zero, all the tests return 1, so the test is not discriminating anymore. If the C is too big i.e close to 255, the tests return almost always 0, and the tests are also not discriminating anymore.

For the values between these limits, the shape of the frequency histograms varies with the change of C . As we will see, these variations are due to the global distributions of the quantities $|I_1 - I_2|$. I_1 and I_2 are the intensities of pixels in the images, chosen at random. For each data set, we show these distribution in figures 5.9, 5.10 and 5.11.

In each case, we observe two peaks : one around 0 and another one around 130. The first one represents the slight variability between quite same pixels. The other peak is due to edges : the pair of tested pixels has a point inside the object and the other one outside it. However we also notice some differences between data sets, and this is the reason why we need an optimization of this parameter.

Though we have two peaks in the same position, the spread of these two peaks between the data sets are really different. We can see that the peak at 0 of the magnesium are less spread than the ones of the mint, for example. This is exactly the reason why we need some optimization. As a reminder, this test is very interesting in order to have a great robustness compared to the non-significant variability of image. The best C is then the one that split the distribution in 2 parts : the part which is associated to quite same intensities and the significant part. So the less is the spread of the peak around 0, the smaller must be the value of the best C . In summary, the goal of C is to split the peak around 0 from the rest of the distribution.

Therefore, a first attempt would be to look for this cut, when the first peak ends. Unfortunately, this seems quite complicated in practice because this intuition is not easily quantifiable, because we need some stopping rules and regularization. An attempt would be for example :

$$C^* = \operatorname{argmax}_C [N(C) - \lambda C]$$

with C^* the value of C which gives the best performances, $N(C)$ the number of quantities $|I_1 - I_2|$ such that $|I_1 - I_2| < C$ and with λ that penalizes a choice of big C . However, this model does not bring good results, and other more global approaches may be investigated.

5.5 Histogram Test

Another important aspect from an anomaly detection point of view could be the global distribution of the intensity of the pixels. This test will be called the histogram of intensities test, or HI

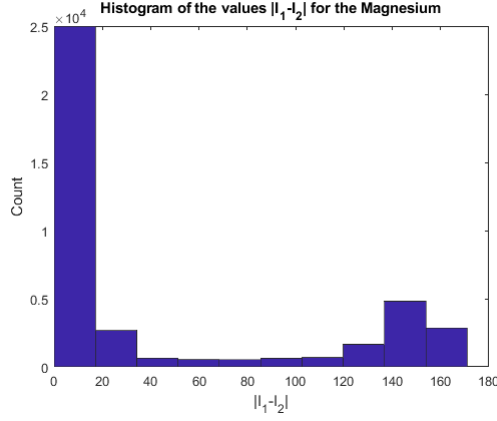


Figure 5.9

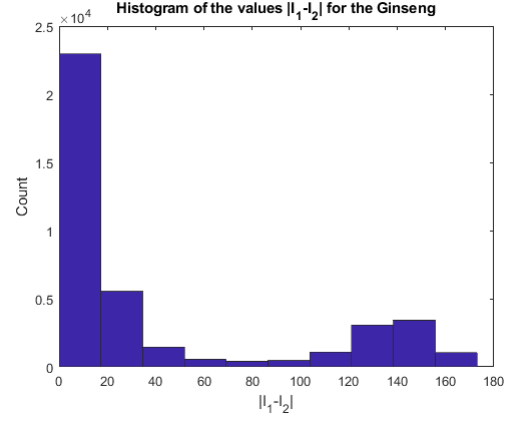


Figure 5.10

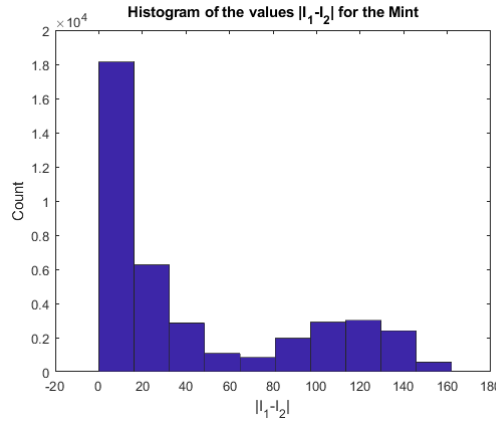


Figure 5.11

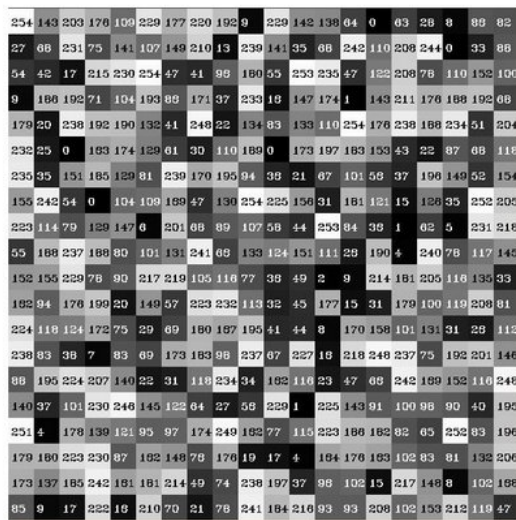
test. Though we have already considered features for straight cuts and abnormal shapes, the intensities of the pixels in a patch can be a good criterion of comparison in certain case. For the data set of mint for example, it may be interesting to investigate this feature, it may allow us to find area in the image where there are more or less small spots, for example. We can easily predict that this feature can be useful.

In a practical way, we choose as parameter a number of bins b in the histogram. For each pixel in a patch, we increment the bin of the histogram which corresponds to the intensity of the considered pixel. An example is shown at the figure 5.12. Then, each test will be composed of two bins, selected at random. The realization of the test will be then the simple comparison of these 2 bins, as for the HOG. We get :

$$T_I = \begin{cases} 1 & \text{if } hi[i] < hi[j] \\ 0 & \text{otherwise} \end{cases}$$

The openCV function `calcHist` performed the computation of the histogram of the intensities.

We can observe in the figures 5.13 to 5.15 that this feature has difficulties to capture the anomalies because the test does not bring better results than the basic test. As it was already said, it can be interesting to keep it for two reasons. Firstly, this test can suit for other type of data or in other context, as we will see in the next chapter. Secondly, it stays interesting to have a counter-example of bad test, in order to compare some proprieties of the tests as we will see in the next chapters.



$$[0, 255] = [0, 15] \cup [16, 31] \cup \dots \cup [240, 255]$$

$$\text{range} = \text{bin}_1 \cup \text{bin}_2 \cup \dots \cup \text{bin}_{n=15}$$

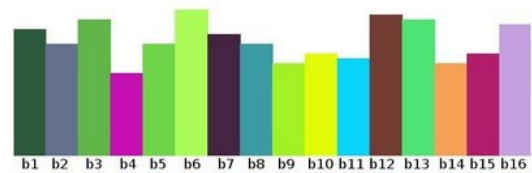


Figure 5.12

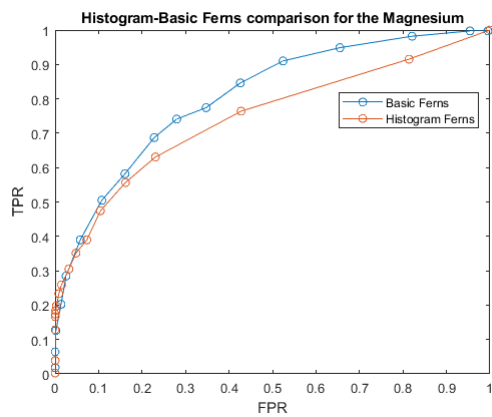


Figure 5.13: [1]

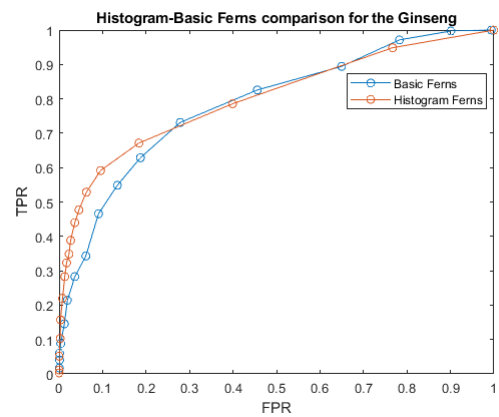


Figure 5.14

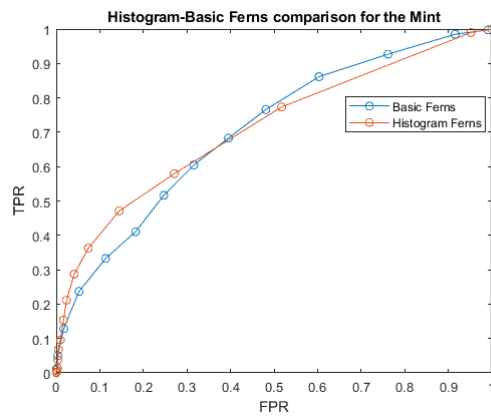


Figure 5.15

Chapter 6

Mathematical Morphology

In the previous chapter, we saw that the initial test in the random ferns is not optimal for anomaly detection. We also find new tests that can be useful to highlight anomalies in several types of data sets. However, we also notice that the database with the mint does not respond as well to tests as the other data sets for some features, as for the gradient test, for example. This obviously comes from the high variability of the small ingredients in the pill. So the question of this chapter will be how to confirm the behaviour of this data set compared to the tests, and possibly improve the response to these tests with this type of data.

A way to do it is using the mathematical morphology. The idea behind this theory is not to find new tests, but to find a way to transform this high variable kind of data in order to be able to make the previous tests relevant.

The mathematical morphology is the general theory which studies the geometrical structures inside an image. This theory is initially applied to binary images, but can be extended to gray-scale images in our case. In order to get deep knowledge in the behaviour of the tool, we can define the operation in a strict way.

6.1 Binary Morphology

This part comes from the references [11].

The binary morphology is the mathematical morphology applied to binary images, in which each pixel is equal to 0 (black) or 1 (white). In this work, the two concerned image operators are the binary dilatation and binary erosion.

For the dilation, we must define a structuring element B , which is a smaller binary image than A . B is then sliding across the image A and if a '1' of A meets a '1' of B , then the concerned pixel of $A \oplus B$ is '1', and '0' otherwise. It can be seen as taking the maximum of the multiplication of the pixels of A and B , one to one. The figure 6.1 shows the application of the dilation and the figure 6.2 shows the result of this operator from the source at the left to the output at the right.

For the erosion, B is still sliding across the image A but if all '1' of A meet all '1' of B , then the concerned pixel of $A \ominus B$ is '1', and '0' otherwise. It can be seen as taking the minimum of the multiplication of the pixels of A and B , one to one. The figure 6.3 shows the result of an erosion.

It is easily understandable that the shape of the structuring element, i.e the position of the

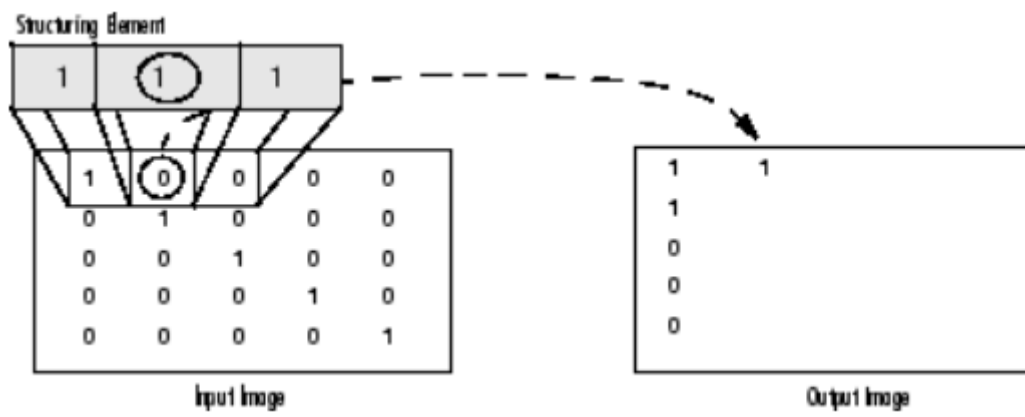


Figure 6.1: [4]



Figure 6.2: [1]



Figure 6.3: [1]

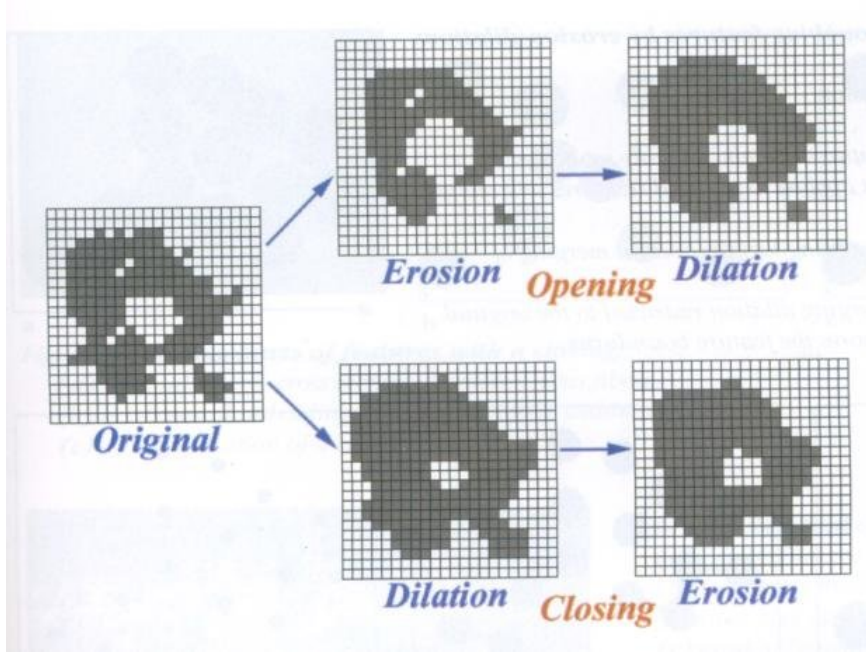


Figure 6.4: [15]

'1' in the matrix B , is determinant for the resulting matrix. These operators let us to highlight the sub-images of A which are similar to B . Therefore, some structures in the image A can disappear or stay according to its similarity with B .

Moreover, we can see the non-linearity propriety of these two operators, which comes from the fact that we take the maximum and minimum functions.

Combinations of these two operators can be created : the binary opening operator ' $\circ$ ' and the binary closing operator ' $\bullet$ '. and can be defined as :

$$A \circ B = (A \ominus B) \oplus B$$

$$A \bullet B = (A \oplus B) \ominus B$$

We can observe an example of the two operators in the figures 6.4.

6.2 Gray-scale Morphology

This part comes from the references [10] and [8].

Now, we will extend these notions to the gray-scale case. For the dilation, the structuring element (which stays binary) is still sliding along the image, but it is now the maximum of the image where the entry of the structuring element is 1. For the erosion, it is the same, but we have the min function instead of the the max one. As for the binary morphology, we can define the opening and the closure as before. We can see the application of a closure at the 6.5 to 6.8, with different sizes and different forms of structuring elements. The figures 6.5 to 6.7 show a closure for a structuring element of type ellipsoid for different sizes. The figure 6.8 shows a MM with a rectangular structure.

6.3 MM before tests

Now we will see if the mathematical morphology is suitable with all tests and the gain of performance. In the figures from 6.9 to 6.11, each figure represents one test, and on each figure

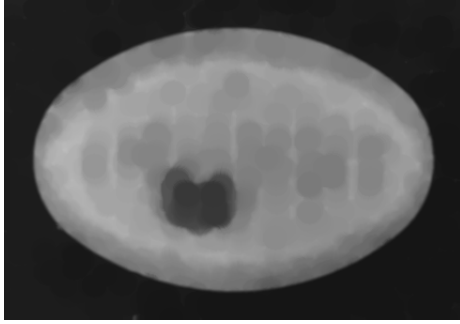


Figure 6.5

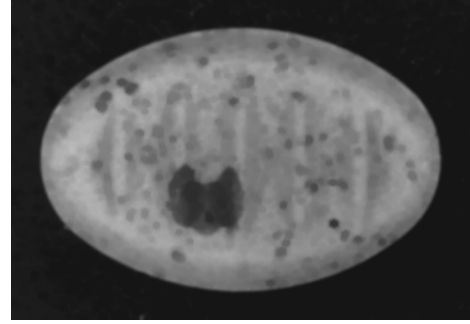


Figure 6.6

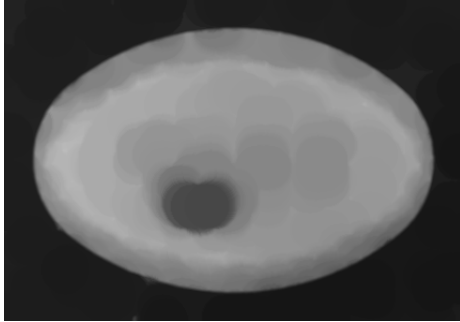


Figure 6.7

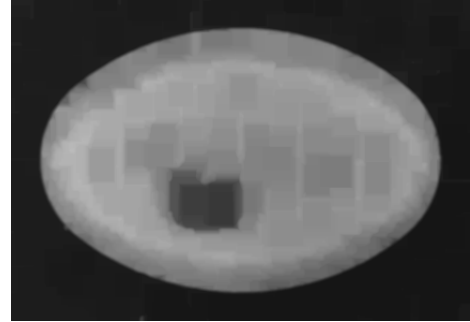


Figure 6.8

we have the basic test of random ferns, the test without mathematical morphology and the test with it.

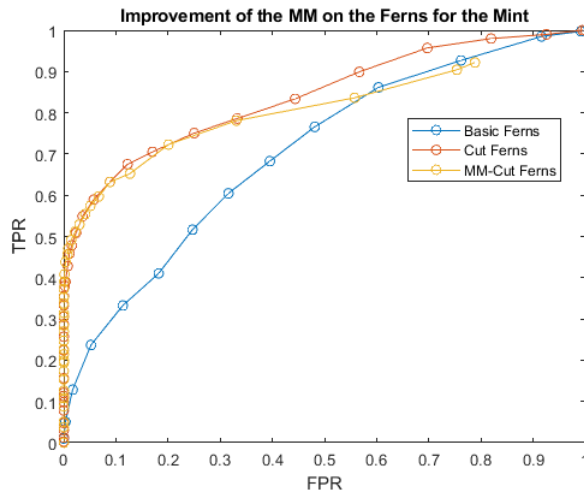


Figure 6.9

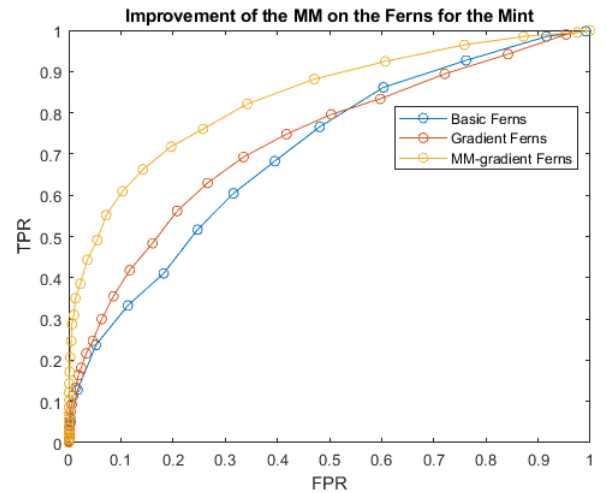


Figure 6.10

We can observe that the improvement of the performance for the mint is not systematic. There is a big improvement for the tests HOG and HI, but the MM deteriorates the performance for the cut test in the last part of the line chart.

The intuition that we have on the reason why the gradient test does not work as well for the mint as for the magnesium is confirmed with the mathematical morphology. The action of smoothing the image makes the gradient test more relevant for these types of data. However, The application of the MM means the lost of the interest of the random ferns : the benefit to not have to apply a pre-processing before the evaluation of an image. So, the mathematical mor-

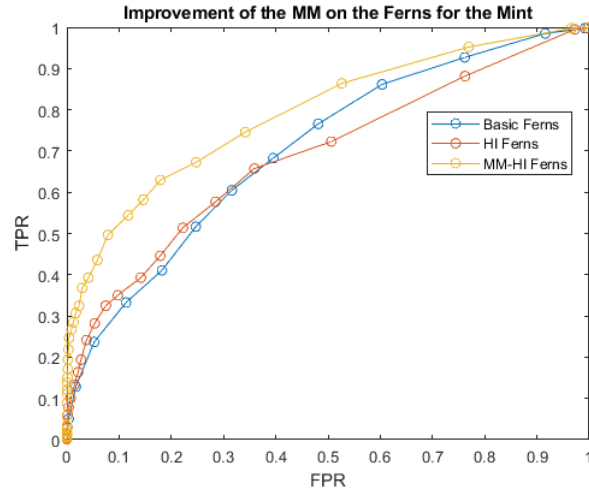


Figure 6.11

phology seems an efficient way to improve the performance of several tests, but we will prefer to use the cut test in this case for example, which gives as good results as the gradient test with MM.

So, we can see emerging a kind of specification for each test, to a certain type of data set. The next chapter will introduce a method in order to choose the best decisions to take for the algorithm in order to get the best performance.

Chapter 7

Optimization method for the adaptive random ferns

Until now, we have seen several kind of tests that can be useful in order to capture some types of problems in images and in specific data sets.

The question is now : which test do we apply and with which parameters? The purpose is to try to develop an algorithm which is able to discover which parameter and which test it must use in order to get the best performance of detection. Each choice of this adaptive algorithm can be seen as a decision to take.

Thus this chapter presents a method to do a clever decision, in order to do the best classification as possible. However, this part seems more complicated than it looks due to the unsupervised assumption. In supervised cases, we can imagine that the best parameters are the ones that can minimize some errors over the classification, according to the given labels. In our case, it is impossible to choose the parameters like that, since we do not have these labels during the training phase. Therefore, this chapter develops an alternative way to find the optimal parameters.

We can define the general decision θ , which is the considered decision that the algorithm must take, and θ^* the output of the optimization method, which is the best value of θ that the algorithm must choose. This decision variable can be a binary variable in order to decide to apply the mathematical morphology or not, or the best value of a parameter, or the best kind of test that the algorithm can apply on the data set, for example.

In particular, we will try to solve an optimization problem, which can be expressed as :

$$\theta^* = \operatorname{argmax}_{\theta} Q(\theta)$$

Then the purpose of this chapter is to find a good representation of what we want to optimize, i.e the function $Q(\theta)$.

Before that, a very important concept is used in the following lines, which deserves some particular attention. This concept is the discriminating capacity of a test. This can be defined as the characteristic of a test to be unstable compared to a bunch of images. In other words, this is the capacity of a test to give different outputs. We will see how this characteristics can be used for our purpose.

7.1 Spreading of a frequency histogram and the discriminating capacity

The discriminating capacity of a test can easily be connected to the frequency histograms of the test. A test which is discriminating is a test whose the frequency histogram is very spread,

and the other way around is also true. So how this discriminating capacity can be used for our optimization problem?

As we said in the previous chapters, a "good" fern, i.e a fern with a "good" histogram, is the one that is able to capture the underlying structure of an object, i.e deeply understands what make this object normal or defective. As a reminder, the important assumption that we made is the following : the defective objects are negligible in probability compared to the normal ones.

Hence, the following question appears : How can we connect the quality of the histograms with the information that we have ?

In order to get a deeper feeling on the frequency histograms and to try to develop a method in order to make the algorithm adaptive, we can show a fern of the accumulated frequency histograms over all the patches in the figures 7.1 to 7.3, for the three kinds of test that we presented earlier. The figures are shown for the magnesium, and each colour represent the frequency histogram of a single patch. In fact, these three images are very useful in order to deeply understand the specific working of each test, and understand how their performance comes from. There are two key observations in this section. The first thing to notice is the higher

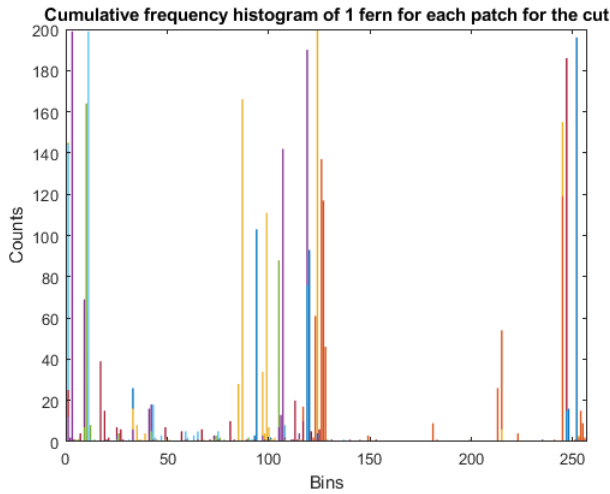


Figure 7.1

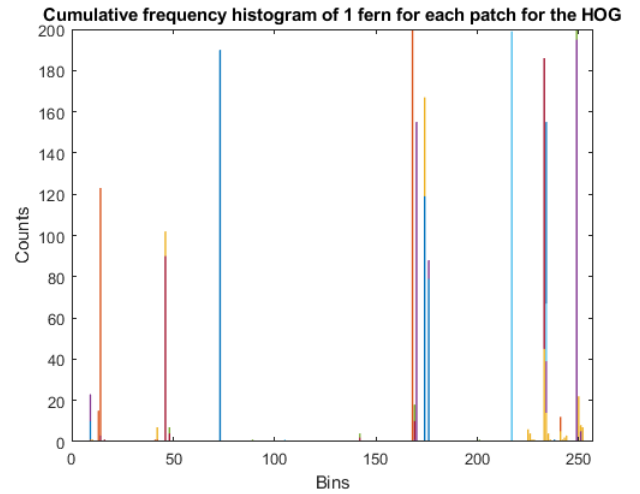


Figure 7.2

spreading of the cut test. Indeed, 109 entries of the frequency histograms over 256 total possible bins are used, while only 37 and 11 are used for the gradient test and the HI test, respectively. In order to get a good test, this test must fill the realization space as much as possible. Indeed, in order to have discriminating tests, it must be the more sensitive as possible over multiple types of images. So it is possible to explain the poor performance of the HI and test by the fact that they do not discriminate enough the images, so the resulting probabilities are not so relevant, since there is only a part of the realization space that is used. However, this is not the only variable that we need to deal with, because though the HOG test has only 37 used bins, it has as good performance as the cut-test for the magnesium. So, another variable must be kept in mind. Secondly, a second aspect must be considered : the spreading of a local area. Indeed, we can imagine that if the spreading in one colour is very high, the underlying structure of the patch is not caught. In order to be able to draw good approximations for the probabilities of the realizations, each colour in the chart must have a sub-histogram as less spread as possible. This is the reason why the HOG test is so performing while it has so low diversities : the gradient test is more efficient in each local fern.

Thanks to these figures, we have identified two important criteria for the quality of the frequency histogram.

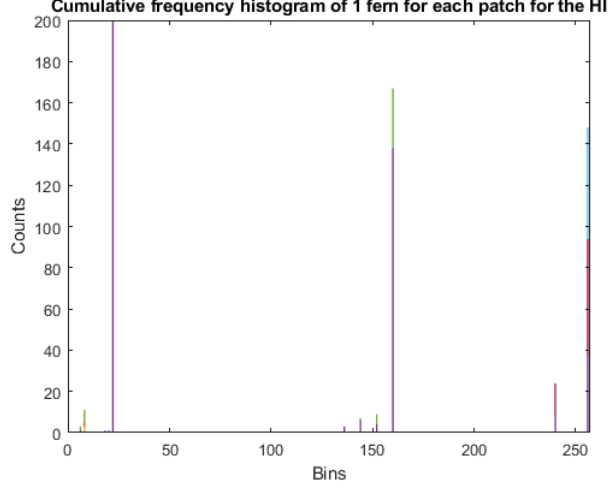


Figure 7.3

On one hand, a good test make the histograms gathering the normal objects in the same binary representation. The reason follows. In the presence of good tests, the ideal scenario would be the histogram with a high frequency for the normal images in the same binary representation (in order to give a very high probability for a new normal image) and with the most uniform distribution for the encountered binary representations of the failed objects. In other words, we want to keep the test as redundant as possible for normal images.

On the other hand, this condition is not sufficient and we also want to keep the tests discriminating enough. Some really bad tests can generate few spreading on the histogram. Imagine the cut-test with the value $C = 0$. All the images will have the same binary representation while this test is completely useless in order to do a classification. We also need to check the information contained in the tests. With regard to the previous discussion on what make a good histogram or not, it allows us to express what we want. We need a compromise between the information of the test and its redundancy for the normal images.

In fact, the information and the redundancy of a test can be seen as opposite concepts. A redundant test tends to gather the images in the same binary representation, while an informative one tends to split the images in a lot of different bins. Then, we can directly see the straight relation between the information of a test and the spreading of a frequency histogram. So, our information index of a test would be based on the spreading of the generated frequency histogram. We will define the *function of narrowness* g , which will be defined mathematically later in several way, and represents the contrary concept of the spreading. Then, the more the histogram is spread, the less will be the narrowness function of this histogram.

7.2 Optimization problem

In this section we will see the procedure that can be used in order to choose the best decision θ^* in using the function of narrowness g .

Firstly, let's remind the definition of the fern F_m^j that is the random variable of the fern m for the local area j . In the same way, we can define $h_m^j(\theta)$ as the histogram of the fern F_m^j , for the parameter θ . More simply, any frequency histogram is called h if neither the belonging to the fern nor the local area are needed.

We also need an union operator " $\cup$ " on the histograms, defined as the agglomeration of the considered histograms, i.e where each entry of each histogram are added one to one, respectively. As a reminder, we also need the function $g(h)$ of a frequency histogram h , which gives the index

of the narrowness of the histogram. For the moment, we let g abstract because it does not affect the general concept of the optimization method. Therefore, we will see later which particular function can be used.

Now, we are able to express exactly what we want with the above definitions. As a reminder, we want to highlight underlying structures in an image, while keeping relevant tests. These two objectives seems to be absolutely opposed, because having a very tight frequency histogram means that the tests contains few information, and the other way around is also true.

The idea of the optimization follows. Firstly let's reduce the problem to a single local area of the image j . As a reminder, we want the best θ such that the binary representations with regard to the tests of M ferns F_m^j gives the frequency histograms of this local area as less spread as possible.

However, we need to counteract the fact that the histograms of the local area become very tight with non-relevant tests, with C equal to 0 in the cut test, for example. So we also need to take into account the relevance of the tests in another context, in order to not interfere with the optimization of the current local area. It seems reasonable to take the other local zones as this external context, since the other zones are supposed to be independent compared to the current patch. So, the need of getting relevant tests may be requested as getting very spread histograms in the other patches.

Finally, we are able to write down the complete objective function as the compromise between a small spreading for the M histograms of the current patch, and a big spreading for the M combined histograms of the other patches. The combination of the histograms refers to the global discrimination of the tests among all the other patches. In practice, this is achieved by feeding the same frequency histogram with the realizations of all the other patches, without clearing it. It is important to take several ferns, in order to optimize the value of the cut independently from the position of the test in the patches. In taking M ferns, we reduce the probability to get the trivial case of having bad or good tests for all values of all the decision θ . In practice, taking M ferns into account can be done in averaging their narrowness, i.e. their evaluation of $g(h)$.

An important remark can be done for the choice of the mean on the different ferns. It is important to keep in mind that the final performance of several ferns are not exactly correlated with the mean of the performance of each fern. Indeed, we have seen that in presence of a good test (i.e informative enough) a fern can never bring bad information on the object, it brings no information in the baddest case, where the frequency histogram tends to the uniform distribution. Therefore, even if the mean of several ferns is not very high, they can contain few ones which are excellent, and these ones are strong enough to be sufficient, whatever the performance of the other bad ferns. So, this computation of the mean can be token, but we must pay attention to the fact that it is a very specific kind of global performance and other ones could be considered, as few of the best performances, for example. However, we will see that the consideration of the mean is sufficient to get very good results, and it will be kept for the rest of this work.

We can see an example of the method for a single patch j in blue and for a single fern, in the figure 7.4. The patches in red are the complementary patches of j . We can observe the construction of the histogram $h_m^j(\theta)$ in blue, and the agglomerate histogram $\hat{h}_m^j(\theta)$ in red.

Now, we can define the quantity $Q_j^1(\theta)$, which represents the global narrowness of the test for the area j and is computed as :

$$Q_j^1(\theta) = \frac{\sum_{m=1}^M g(h_m^j(\theta))}{M}$$

$Q_j^1(\theta)$ is then the average of the narrowness on the M frequency histograms of the area j .

Then, we compute a second set of histograms, which are the agglomerative histograms of the application of these same ferns, to all the other local zones. So we can define :

$$\hat{h}_m^j(\theta) = \cup_{k \neq j} h_m^k(\theta)$$

After that, We can describe the quantity $Q_j^2(\theta)$ which is the global narrowness on the M agglomerative histograms of all the patches, without the patch j . This quantity is computed as :

$$Q_j^2(\theta) = \frac{\sum_{m=1}^M g(\hat{h}_m^j(\theta))}{M}$$

Now, what we want is catching the best value of the cut θ^* that making the h_m^j as less spread as possible (because it can highlights a good split between the good and defective images), while making the $\hat{h}_m^k$ the more spread as possible (because we want informative tests), for each patch. Therefore, for each patch j , we can define $Q_j(\theta, \lambda)$ the compromise that we want to have as :

$$Q_j(\theta, \lambda) = Q_j^1(\theta) - \lambda \times Q_j^2(\theta)$$

Where λ is a weight in order to give more or less importance of the second term compared to the first one. The goal is thus to maximize the quantity $Q_j(\theta, \lambda)$, in order to have a good test in the local area j . Of course, the value of λ has a determinant role in the optimization problem. Typically, we will see that the choice of θ^* can be highly dependent of the value λ , so it will be denoted as $\theta^*(\lambda)$. Now we have the P quantities $Q_j(\theta)$ and we must find the best decision with these quantities. Two options are possible. The first one is to define a global decision for the whole space of the image. The other one is the optimization of the decision for each local area. We will define the both approaches in the following sections.

7.2.1 Global optimization

A possible way to do it is the search of only one best decision value for the whole space. The best $\theta^*(\lambda)$ will be choose as the θ that maximizes the mean over all the local area j of the quantity $Q_j(\theta, \lambda)$. The maximization problem becomes :

$$\theta^*(\lambda) = \operatorname{argmax}_{\theta} \frac{\sum_{j=1}^P Q_j(\theta, \lambda)}{P}$$

We recall that P is the number of patch that we use per image.

7.2.2 Local optimization

A second way to solve the problem is to define P decisions θ_j^* , which is the best decision in the local area j . Therefore, there will be P independent optimization problems such that :

$$\theta_j^*(\lambda) = \operatorname{argmax}_{\theta} Q_j(\theta, \lambda)$$

Though not necessary in our case, the main benefit of this approach is the processing of multi-components object, with different parts and with different natures for example. In this case, we can imagine better results with this approach. The main drawback of this method is the lack of the capacity of robustness compared to one main decision. Indeed if one local area is very poorly organized, the local optimization would also be very inefficient. So in this case, we can loose the benefit of the mean.

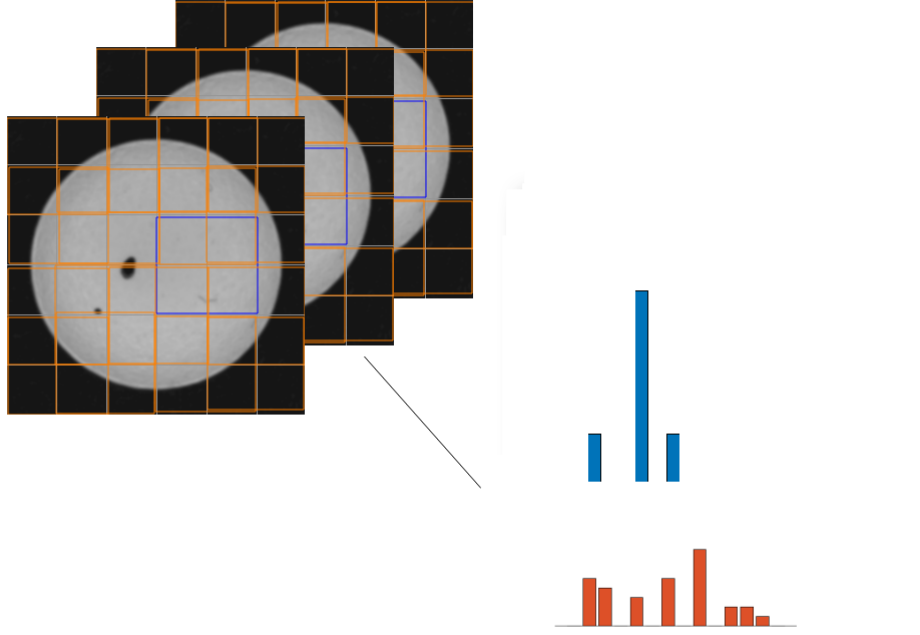


Figure 7.4

7.3 Function of narrowness

In this section we will see how to define the function g , which was left besides until now. We introduce mathematically three possible functions. For each function g , we explain the way to transform the intuition behind the concept of narrowness into a mathematical concept, which seems to be useful for our purpose. Three functions catch our attention : the opposite of the entropy, the maximum bin and the first difference of bins. In this part, we use the notation h for the frequency histograms as the belonging to the fern or the patch is not needed, in order to make the notation lighter.

7.3.1 The entropy of the histogram

The first function that we will introduce is the notion of entropy. We can define it as a function applied on a histogram h . As reminder, the entropy of a distribution is a measure of its uncertainty. More rigorously, let's remind the random variable F_m^j and its possible realization $f_m^j \in [0 : K - 1]$. The total number of possible realizations is K , but not all the realizations are non-zero. Then we must only take into account the realizations whose the entry in the frequency histogram are non-zero. Here, we do not need to take a bias in the computation of the probabilities, because the non-zero entries play a very important role in the computation of the entropy. So, we can compute the estimations of the probability in extending the definition of $\hat{P}_{\text{naive}}$ in the equation (4.2).

So, we get :

$$\hat{P}_{j,m,f}^{\text{naive}} = \frac{h_m^j[f]}{N} \quad \text{with} \quad \sum_{f=0}^{K-1} \hat{P}_{j,m,f}^{\text{naive}} = 1 \quad \forall m, j$$

Therefore, the entropy of a random variable $H(F_m^j)$ can be approximated thanks to its frequency histogram h_m^j such that :

$$H(h_m^j) = - \sum_{f=0, \hat{P}_{j,m,f}^{\text{naive}} \neq 0}^{K-1} \hat{P}_{j,m,f}^{\text{naive}} \times \log_2(\hat{P}_{j,m,f}^{\text{naive}})$$

It can be proved that this entropy function achieved its maximum value when all the $\hat{P}_{j,m,f}^{\text{naive}}$'s are the same and are equal to $\frac{1}{K}$, for all m and for all j . Then it is easily to see that the maximum value is $\log_2(K)$ or in our case S , the number of tests considered in a fern. Thus it allows us to draw a the function g as :

$$g(h) = -H(h)$$

The sign minus is here in order to keep the maximization problem. Therefore, the more is the value of $g(h)$, the less is the value of the entropy, and this coherent with the definition of the spreading of a frequency histogram.

7.3.2 The value of the maximum bin

A possible way to express the narrowness of a frequency histogram is observing the maximum bin of the histogram, which is called $b_{\max}$. We can define it rigorously as :

$$b_{\max}(h) = \max_{f \in [0:K-1]} h[f]$$

Indeed, the more is the value of $b_{\max}(h)$, the more the histogram tends to have a high peak, which can be expected as the binary representation of the normal objects. On the contrary, the less is the value of $b_{\max}(h)$, the more spread is the histogram. We can then define g as :

$$g(h) = b_{\max}(h)$$

7.3.3 The first difference of bins

Another type of narrowness function can be to target more precisely the need of narrowness in order to take into account the maximum bin $b_{\max}$, but also the second one with the second highest value b^* . The idea would be to express the notion of spreading in computing the value of $b_{\max}(h) - b^*(h)$. The higher is this difference, the less must be the spreading. The function g becomes :

$$g(h) = (b_{\max}(h) - b^*(h))$$

Though the entropy is well known to express the capacity of a frequency histogram to be spread, it is slightly deviated from the idea of spreading in the context of anomaly detection.

Chapter 8

Validation of the method

Until now, we have seen different tests that can be useful according to the type of data set, and we have proposed an unsupervised method in order to let the algorithm find the best feature and parameters to apply. In this chapter, we will make a validation of the method, i.e we will test our algorithm on the different data sets and compare the decision of the algorithm with the parameters and the tests that we manually identify as optimal.

In the first part, we will see the optimization of the parameter C of the cut-test with the 3 spreading functions that we proposed in the previous chapter. Then the decision of doing mathematical morphology or not will be analyzed. After that, the different tests will be compared with our method.

Another experiment that is interesting is to compare the performance between the unsupervised method, with the supervised version of our algorithm. This will let us to give a degree of quality for our method.

8.1 Parameters

According to the previous chapter, we will apply our optimization method to the parameter C , for the 3 spreading functions. We will also see the application of the multi-cut optimization.

8.1.1 Single cut optimization

In this section, we are going to develop the method above in order to choose only one C^* for all the patches. The best C^* will be selected as the C that maximizes the mean over all the patches j of the quantity $Q_j(C)$. The maximization problem becomes :

$$C^*(\lambda) = \operatorname{argmax}_C \frac{\sum_{j=1}^P Q_j(C, \lambda)}{P}$$

We will see that this approach is highly sufficient, and only one C^* gives good results for the cut test, and lets us to find the best parameter.

8.1.2 Choice of g

For each function g , we will show two charts for each data set. The first one is the variation of the quantity Q^1 and Q^2 with regard to λ , i.e the evolution of the local spreading and the global spreading when we make changing the weight of the compromise. The second one is the selected C^* compared to the given value of λ . The cost functions for the max bin and first difference are normalized.

The entropy of the histogram

We can observe the results at the figures 8.1 to 8.6. For each data set, we can observe a high value of $Q^1(C)$ at the value of $C = 0$ and $C = 130$. As expected, the entropy of the complementary image (in orange) is much more entropic than the blue one, which is due to the non-local agglomeration of the frequency histograms. We can observe the choice of the C^* compared to the choice of λ in the figures 8.2, 8.4 and 8.6. We can observe that for the two first data sets, the optimal parameters can be obtained for a specific value of λ , and can be approximated in the case of the mint. However, we can observe that the choice of the parameter is very unstable compared to the value of λ . Moreover, the value of λ that reached the best C is quite dependent of the data set. For the magnesium, the λ that reaches the optimum are the value from 1 to 1.5. For the second data set, the value are from 0.75 to 1 and for the third set, the approximation of C^* is obtained for the λ close to 1. The second problem is that the robustness of the selected C^* is quite poor : a small range of λ reaches the optimal parameter. Then it might be interesting to find a more stable way in order to express the spreading of the frequency histograms.

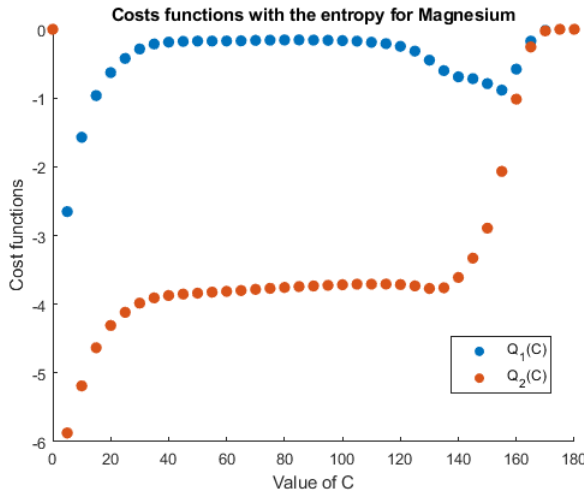


Figure 8.1

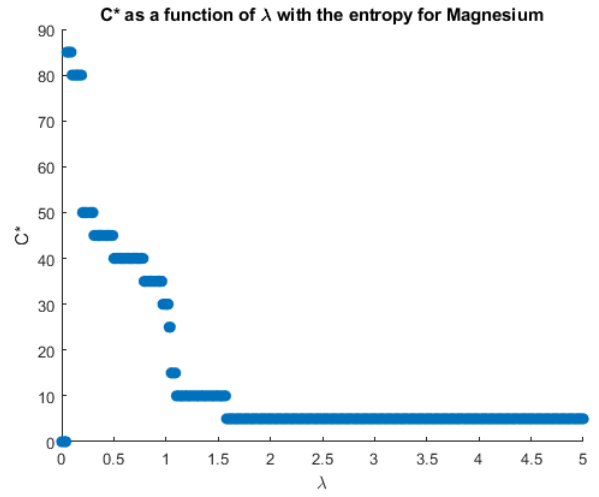


Figure 8.2

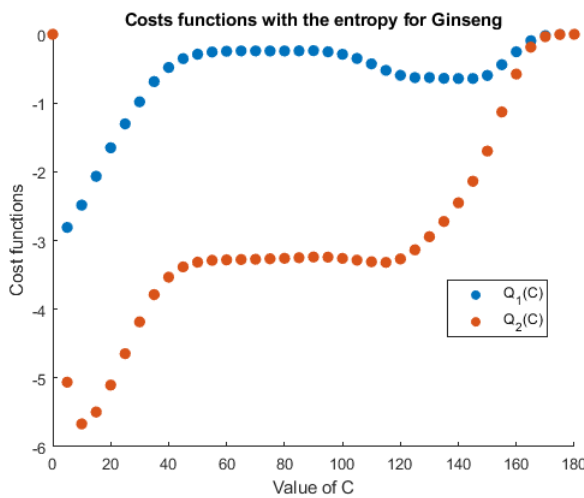


Figure 8.3

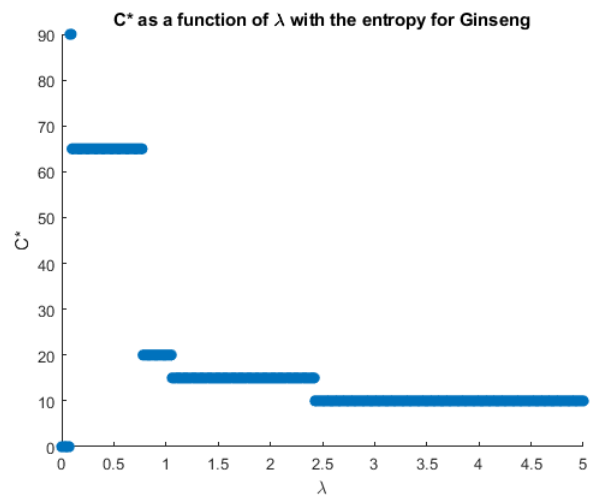


Figure 8.4

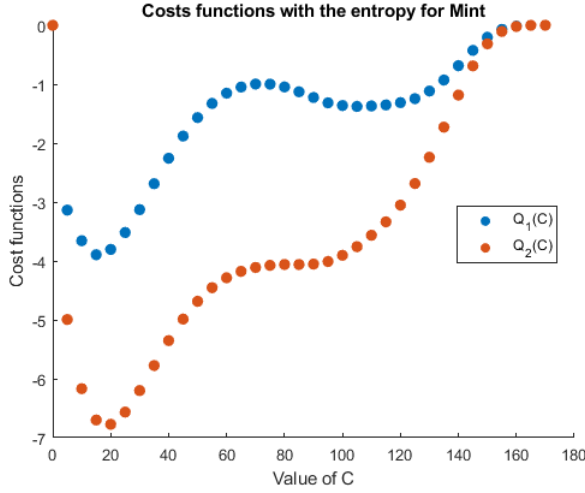


Figure 8.5

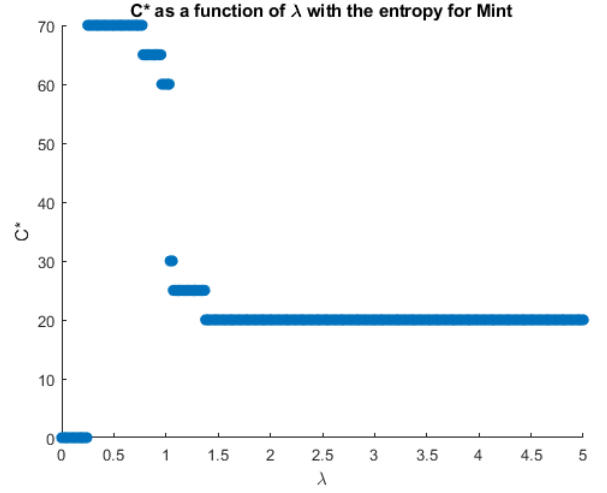


Figure 8.6

The value of the maximum bin

We can see the results for each data set in the figures 8.7 to 8.12. We can observe that the quantities $Q^1(C)$ is low at the values for C of 0 and 130, which correspond to the frequent absolutes values encountered. We see that the right value of λ is found for the data set magnesium and ginseng for big λ : after 2. However, the right value of C^* for the mint is not find for big λ , but it is closed to the optimal value for $\lambda = 2$. We can observe a better behaviour in the search of the best parameter, because the choice compared to λ is more stable, and some values of λ work for all the data set at the same time.

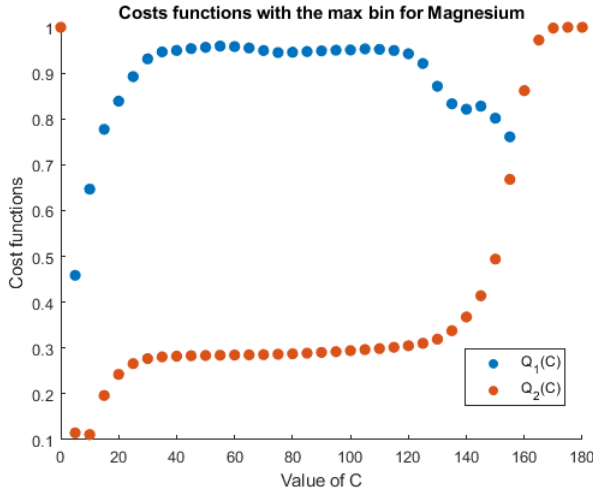


Figure 8.7

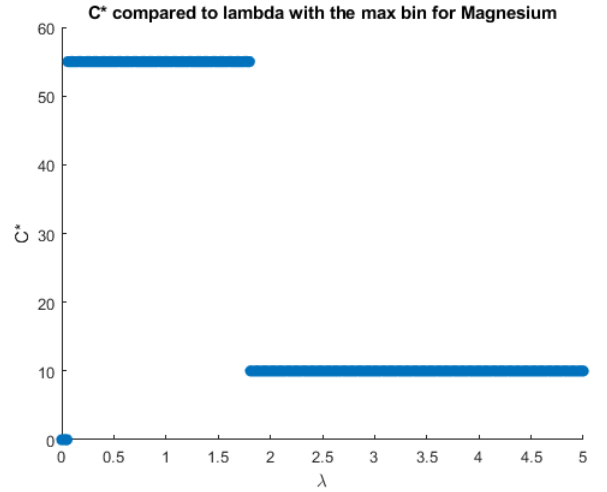


Figure 8.8

The good question to ask is the reason why the entropy, which is a common criterion for the spreading of a frequency histogram, does not work as well as the current criterion. In fact, the entropy is not always related to the maximum value of the bins. If we compare the values of $Q^2(10)$ for the magnesium, between the entropy and the max bin, we see that there is a big decreasing for the entropy while the max bin stays stable. So, this phenomenon can highlight some possible difference between these two spreading functions. Indeed, the max bin can stay the same, but the other bins are less spread.

If we take back the figure 3.1, we recall that the perfect test collects all the normal images in

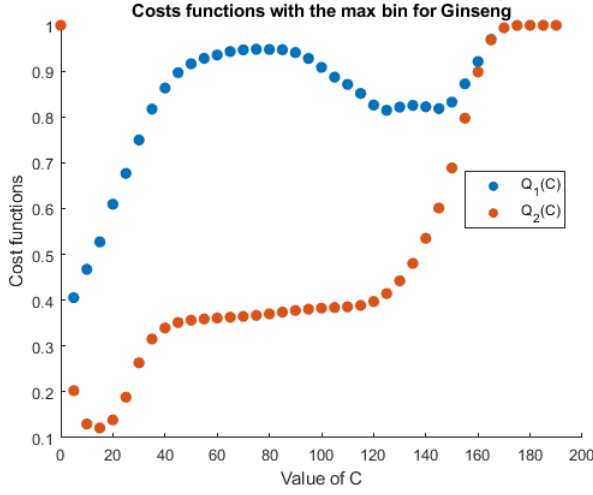


Figure 8.9

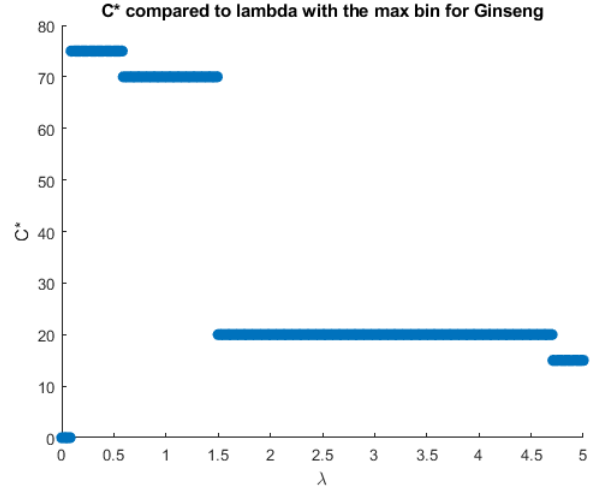


Figure 8.10

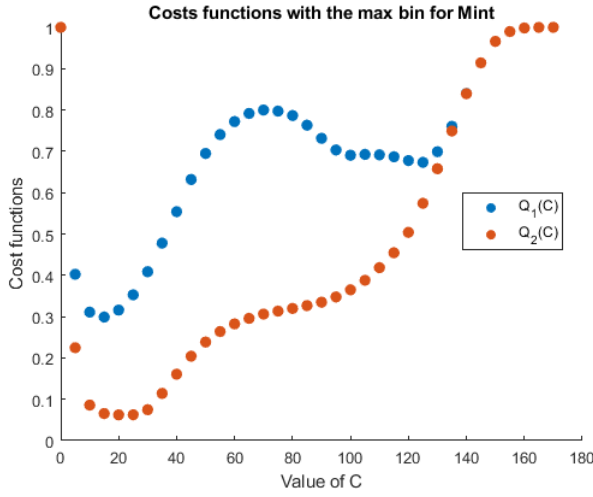


Figure 8.11

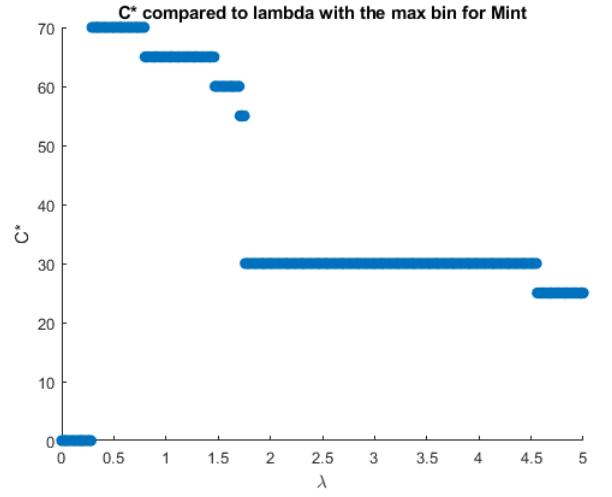


Figure 8.12

one realization and the defective ones are split in the other realizations. The problem with the entropy function is that it does not capture what we really want to point out, i.e. the gathering of the normal images.

This remark goes further for the definition of a perfect test : a perfect test gathers as possible the normal images and splits as possible the abnormal ones. Although this is really difficult to verify, this can be the explanation of the better behaviour between the stability of the two spreading functions.

The first difference of bins

As we see, the max bin highlights the characteristic of the spreading. However, we can try a more advanced function, which can be more precise in order to underline the narrowness of the histogram. Therefore, we try the third function that we introduce in the previous chapter. We can see the results at the figures 8.13 to 8.18. We can observe that the charts are very similar to the preceding spreading function. This can be explained by the high dependency between the first bin and the first difference. Since there is no improvement of the method, we will keep the max value of bin for the optimization of the other decision and see if it works as well for these

ones.

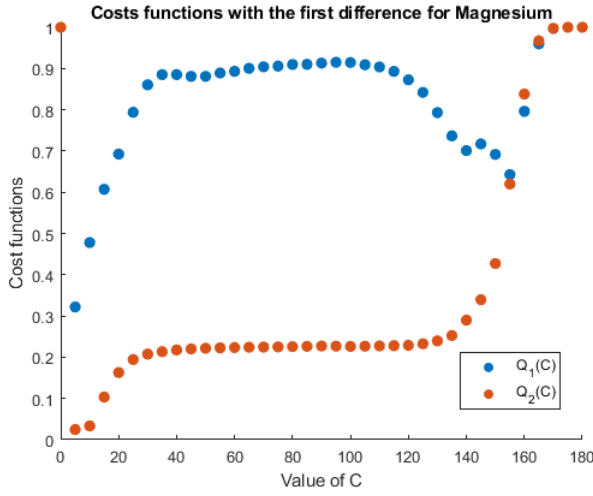


Figure 8.13

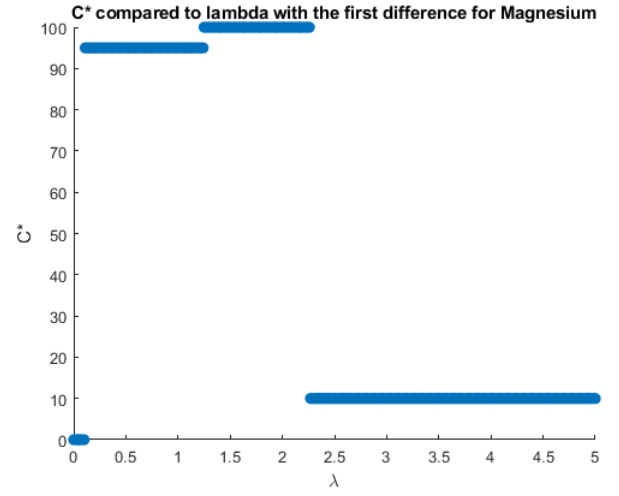


Figure 8.14

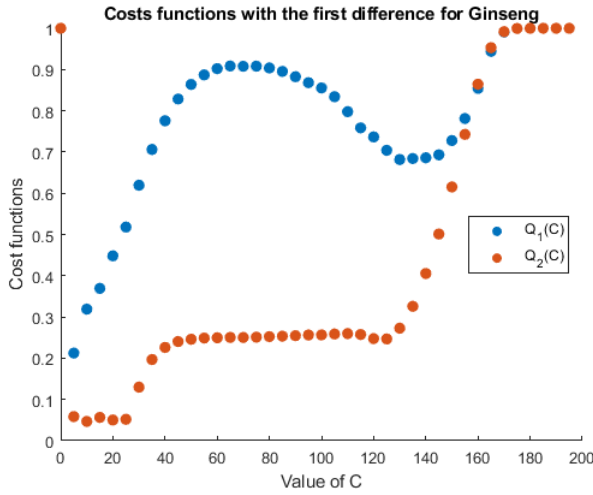


Figure 8.15

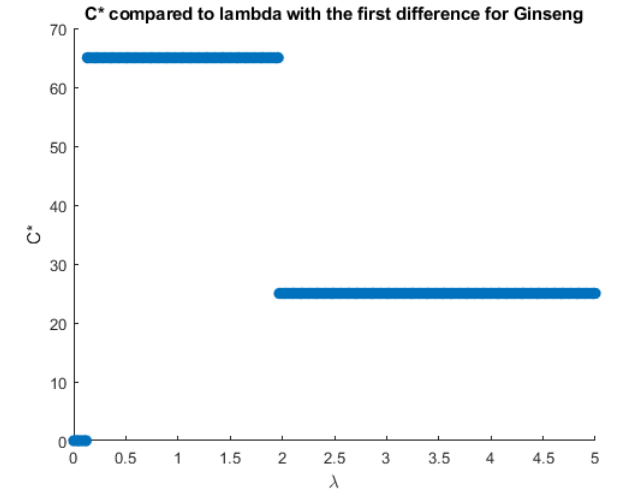


Figure 8.16

As a conclusion of this chapter, we have seen a method that can find the best parameter C^* , thanks to a compromise between the spreading of the frequency histograms and the information they contains. Three definitions of spreading were seen, and we notice that the entropy function does not well represents what the spreading is in our case. Thus we will prefer the definition of the $b_{\max}$.

8.1.3 Multi-cuts optimization

Now, instead of choosing only one C^* , we can choose one optimal value of C for each local zone, that we denote C_j^* . Therefore, there will be P independent optimization problems such that :

$$C_j^*(\lambda) = \operatorname{argmax}_C Q_j(C, \lambda)$$

The results are observable in the figures 8.19 to 8.24 for the magnesium. Only the patch number 0, 12 and 16 over 25 are shown. We can see the different behaviour of the 3 functions $Q^1(C)$: the patches 0 and 16 contain edges of the object, then there is a drop in the chart for a value of C close to 130, which is not the case for function in the patch number 12.

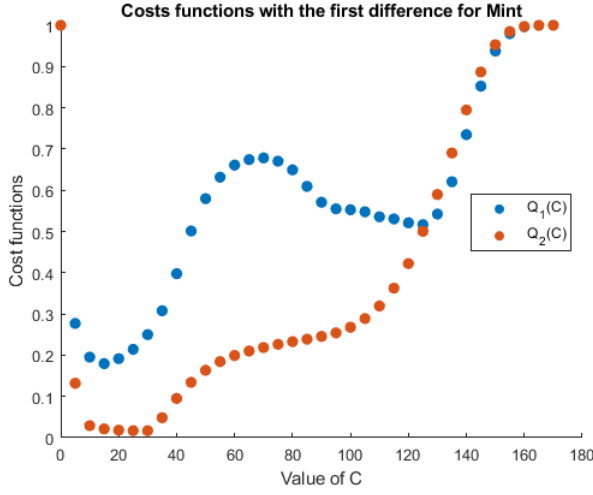


Figure 8.17

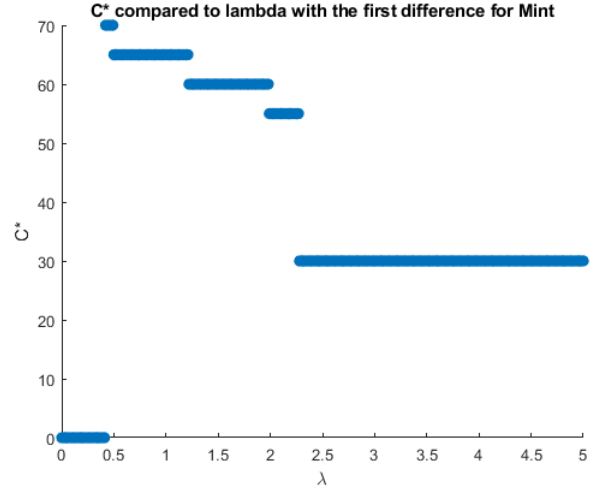


Figure 8.18

Though not necessary in our case, This approach can be very interesting in the case of multi-components object, with different parts and with different natures for example. We can observe that we obtain quite the same values for each patch, which is quite the same as the single value that we obtained for the global optimization.

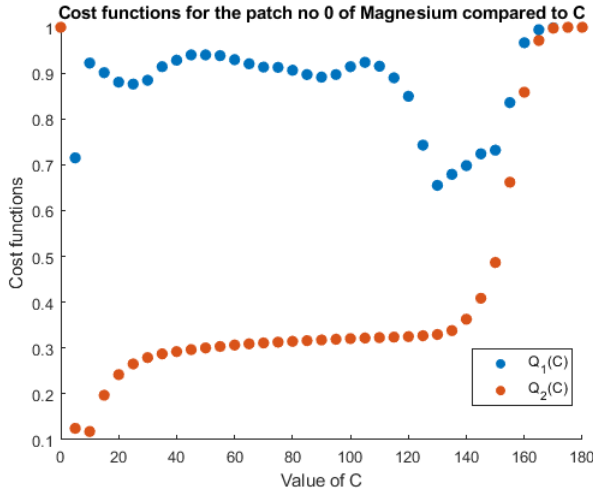


Figure 8.19

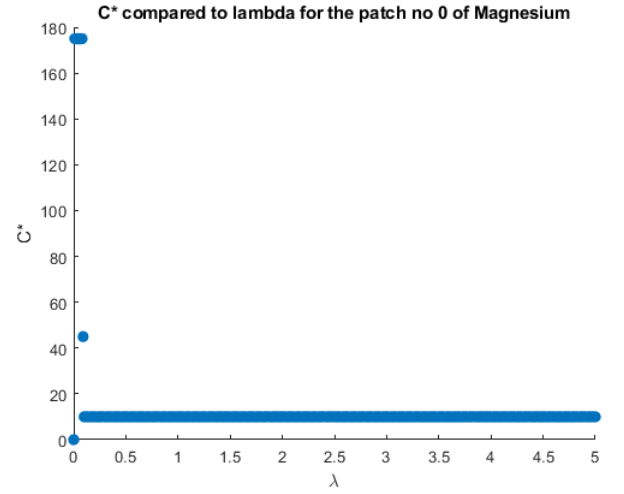


Figure 8.20

8.2 Decision of mathematical morphology

As we saw in the chapter dedicated to the presentation of the mathematical morphology, the application of these tools is not systematic, and even forbidden for several data sets, since it can drastically reduces the performance of the test. So the question that comes up is when do we must apply the tools of morphology or not. Then, the variable θ is a binary variable, it can be equal to 1 if we apply the morphology and 0 otherwise. We see that the most powerful function of narrowness that we saw for the optimization of C is the max bin, so this is the one that we will try for the other following decisions.

We can see the results in the 3 figures from 8.25 to 8.27 for the HOG test in the 3 data sets. The blue line is the value of the objective functions $Q(0, \lambda)$ and the red one is $Q(1, \lambda)$, as a function of the weight λ . We can observe that the proposed method in the previous chapter works for the

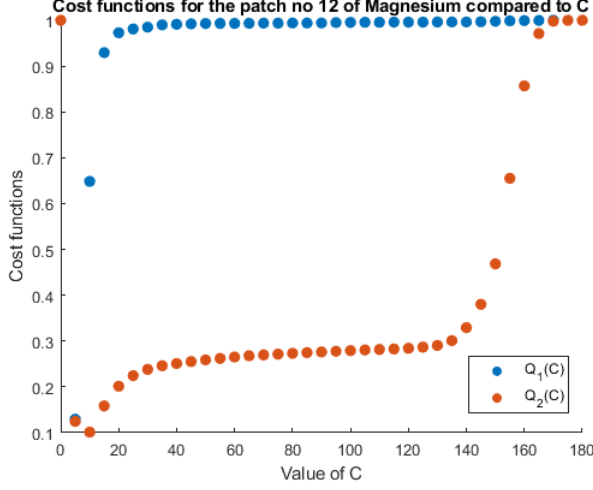


Figure 8.21

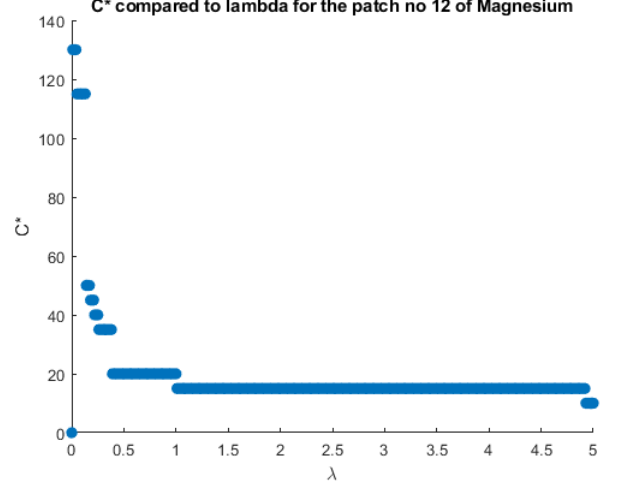


Figure 8.22

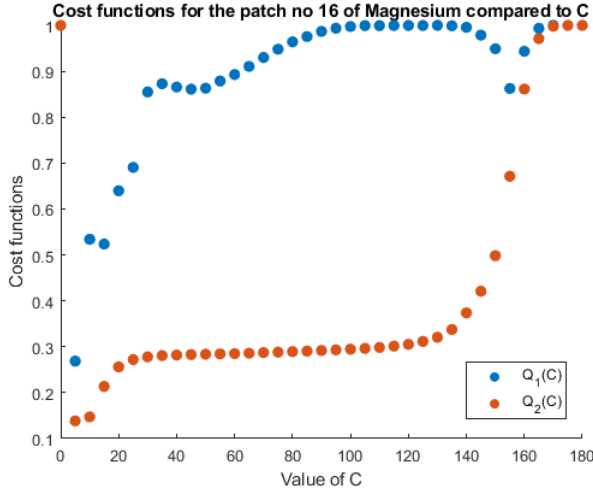


Figure 8.23

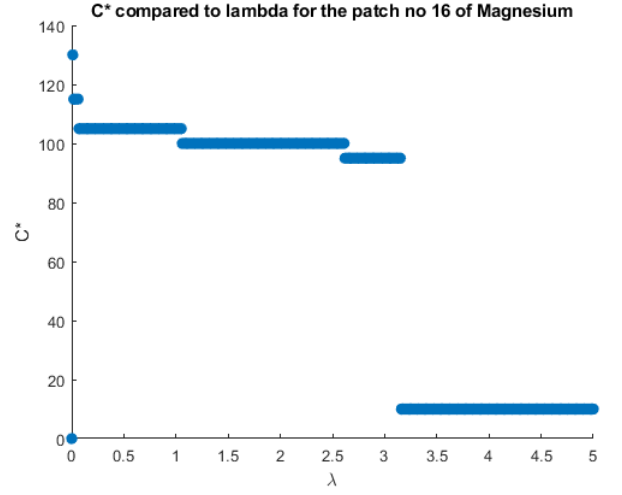


Figure 8.24

3 data sets, and the decision is very few dependent of the value of λ . Indeed, the magnesium and the ginseng does not need MM, and their lines are very close one to each other. We can observe a normal phenomenon for the two first data sets : for the small values of λ , we would better do not apply the MM and the other way around for big value of λ . This behaviour is normal, since the MM tends to smooth the images, so all the frequency histograms tend to become more narrow. Thus for small values of λ , we give more weight to the narrowness of the local histograms, so we would prefer the application of the MM in order to make the histograms less spread. However, if we look at the chart for the mint, we can observe that the red line is always above the blue one, and the difference is significant.

8.3 Decision of the test

Now we will see if we can predict the performance of a test on a specific data base with the method described above with the function b_{max} . We can see the application of the method with the function g as b_{max} in the 4 figures 8.28 to 8.31.

Firstly, these charts confirm the difference between the behaviour of the cut test against the 2 other tests. The cut test has globally more spreading in its frequency histograms. These charts

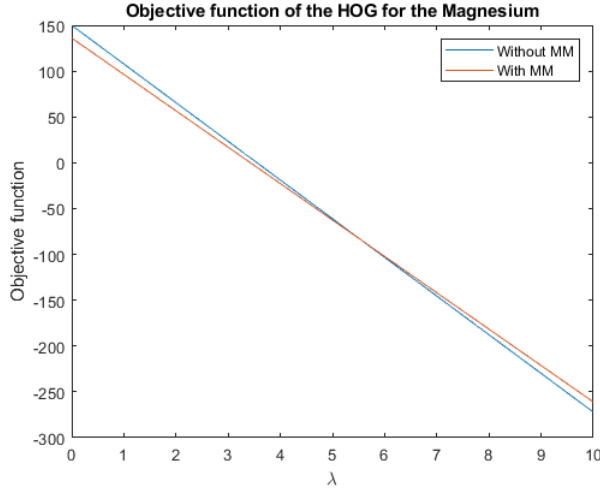


Figure 8.25

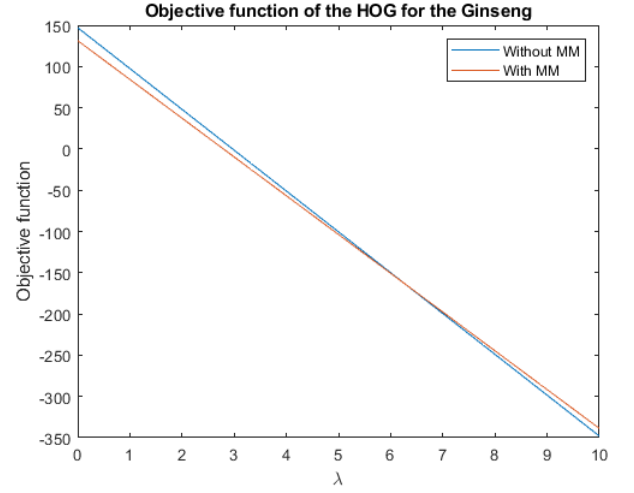


Figure 8.26

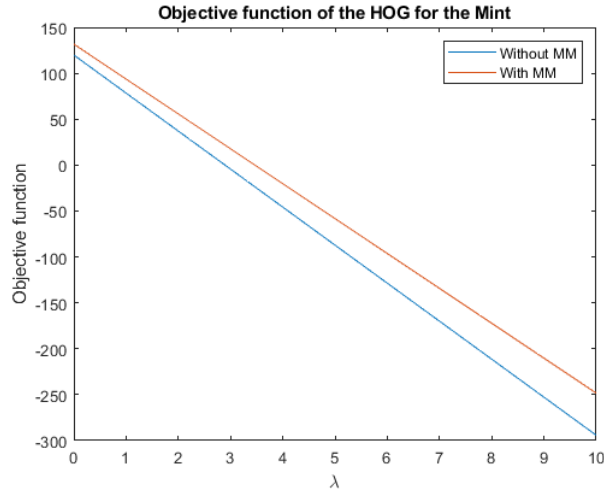


Figure 8.27

can give a piece of information for the behaviour of the 3 tests. We can see that, we cannot globally recover the general order of performances for the three tests with only one range of values for λ . The HI test is always the worst test to apply for these data sets. However, the relation between the HOG test and the cut test, which have quite the same performances, is more indeterminate. The objective function tends to overestimate the difference between the two tests.

We can also observe that when the application of the MM is performed on the data set of the mint, the HOG test has performances close to the ones of the cut test, this is exactly what we observed in the figure 6.10.

However, we can notice that the choice of the test is quite stable with regard to the value of λ , as long as this one is big enough. Moreover, we can highlight the bigger importance that we need to give to the global spreading of the test.

We can try to take back the function entropy and see if there is an improvement in the selection of the best test. We can see the results in the figures 8.32 to 8.34. We can observe that the entropy function is much more realistic in order to compare the different tests. The HI test has always the worst performance, but the performances of the HOG and the cut tests are really close for the magnesium and the ginseng, and the cut test is the best test as depicted

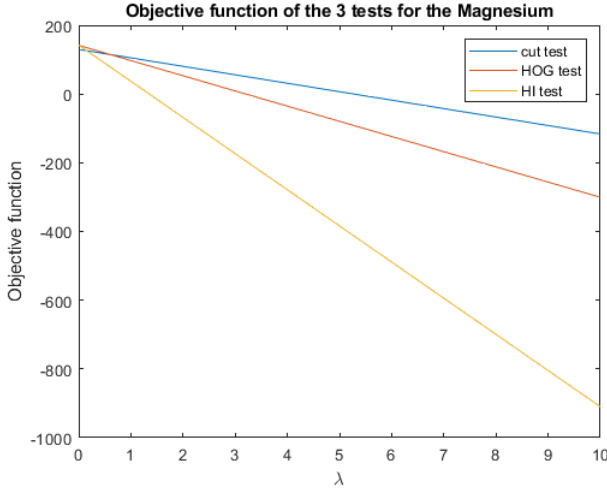


Figure 8.28

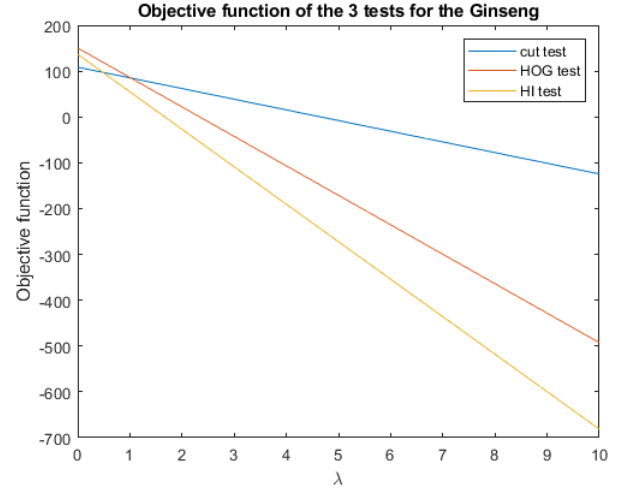


Figure 8.29

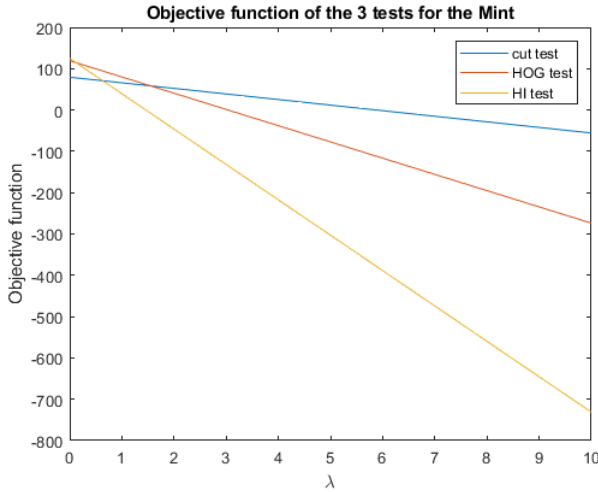


Figure 8.30

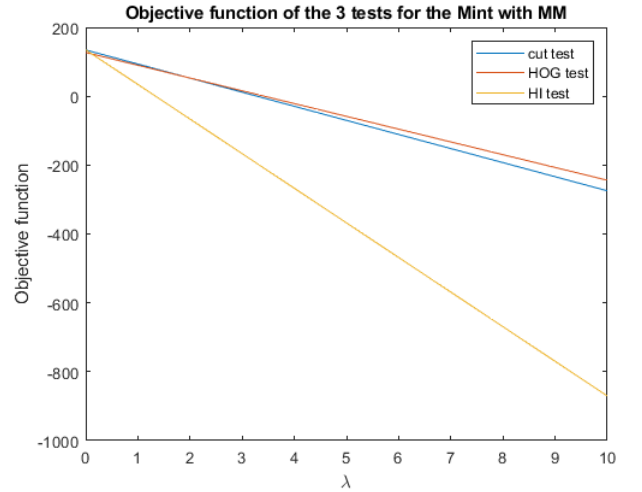


Figure 8.31

in the figure 8.34. We can even find a range of values for λ which gives exactly the good order of performance : $\lambda \in [0.3 : 0.4]$. For the 2 data sets where the hog test and the cut test have quite similar performances, there is an intersection between the two curves. We can observe that the intersection appears earlier in the case of the magnesium compared to the ginseng. It is the characteristics that the HOG test is better in the case of the first data set than for the second one, since the cut test is always advantaged by big λ .

8.4 Opening on the supervised case

Our initial problem was to find an unsupervised method to do an anomaly detection. A good question can be the following one : what does it happen if we apply the same adaptive algorithm to the supervised case? In other words, we can try to compare the performance of the unsupervised optimization method that we developed above, with the case where we have the labels anomaly or clean for the training, in order to find the best decision for each test. We will try to do it for the example of the search of the best value C^* for the new cut test. For sure, this can be applied to the whole adaptive algorithm, i.e we could also do the same work for the comparison between the different tests.

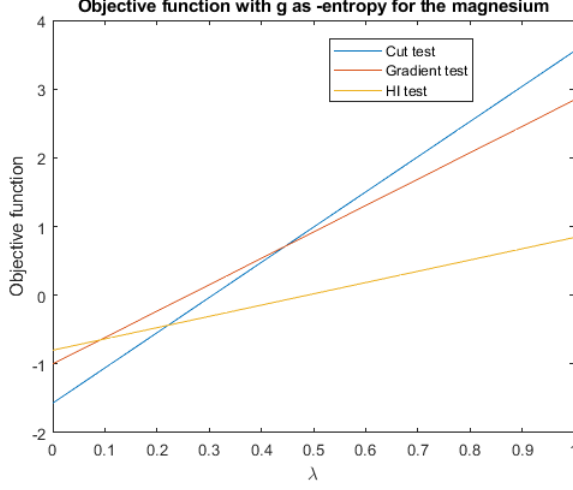


Figure 8.32

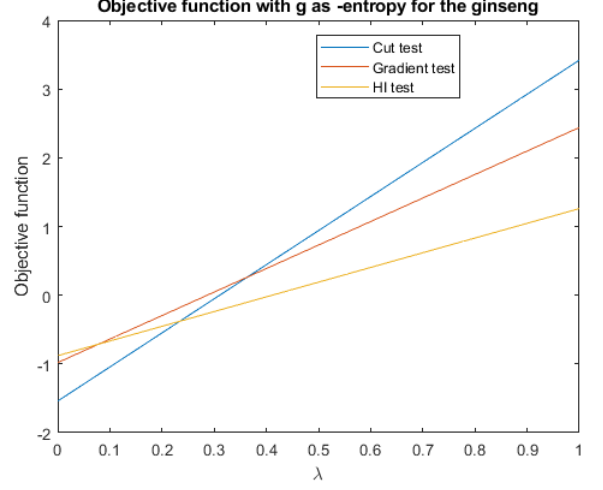


Figure 8.33

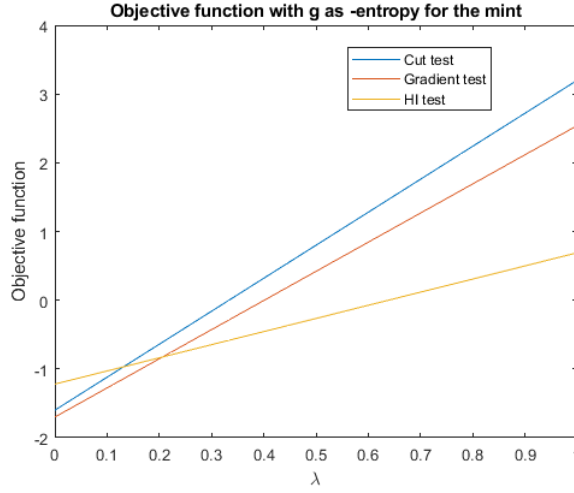


Figure 8.34

The idea is, for each value of the value α , finding the best parameters of the test and the best threshold τ which give the best performance, i.e. the best compromise between the true positive rate and the false positive rate for the training image. The idea is then to express this compromise between the two variables as :

$$(\theta^*(\alpha), \tau^*(\alpha)) = \operatorname{argmax}_{\theta, \tau} TPR(\theta, \tau) - \alpha FPR(\theta, \tau) \quad (8.1)$$

Where θ^* is the best decision θ that can be taken, based on the training with the labels. Then, we can recover these trained values in order to apply them in new examples, and evaluating its performance. So we define two new sets, a set of training where for each value of θ and the threshold τ are iterated, and a second set in order to test the performance on new data. This performance will be compared to the one in the unsupervised case.

8.4.1 Defective and non-defective data available

In this section, we will apply the optimization problem to the data set with defective and non-defective objects. We can see the results in the figures 8.35 to 8.39. In the figure 8.35 and 8.36, we can see the cut and the threshold respectively, that are chosen in the supervised case.

We can observe that the magnitude of the chosen C^* is kept : the cut for the mint is bigger than the cut of the ginseng, which is itself bigger than the cut of the magnesium. We can also notice that the optimal value of C for each data set is closed to the value obtained with our unsupervised method. We can observe that the more the data set is difficult to handle with, the more the selected parameters are dependent of the value of α . We can also observe that the global evolution of the optimal cut is an increase and the global evolution of the optimal threshold is a decrease, with the increase of α . This is coherent with the intuition behind the parameter α : when we increase it, we give more importance to the FPR . In order to decrease the number of false positives, we must increase the value of the cut (because it decreases the discrimination of the test) and decrease the threshold (in order to make the detection more lax).

In the figures from 8.37 to 8.39, we plot the roc-curve with parameters α . So each point of the roc-curve is the optimal point find by the supervised search for a fixed α . These roc-curves are then considered as the best performance that our algorithm can hope reaching. The 3 figures represent the 3 data sets, the blue line is the performance on the learning set, and the red one the performance on the test set, so with new images. We can observe that the red curve is globally under the blue curve. It is normal because the optimal parameters are computed on the learning set. We can also noticed a bad behaviour of the testing roc for the mint. This can be explained with the poor improvement of the training, because of the high variability of the images.

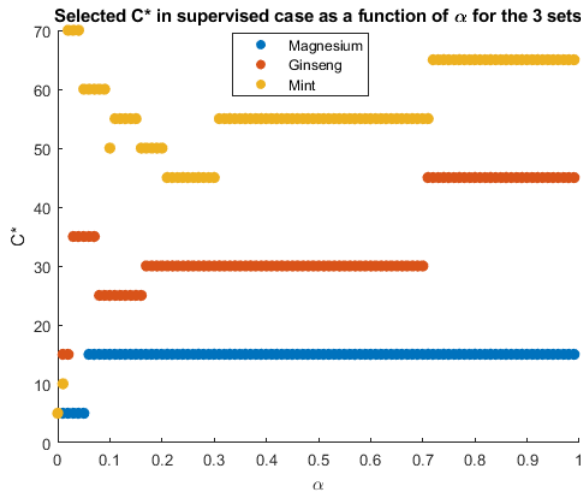


Figure 8.35

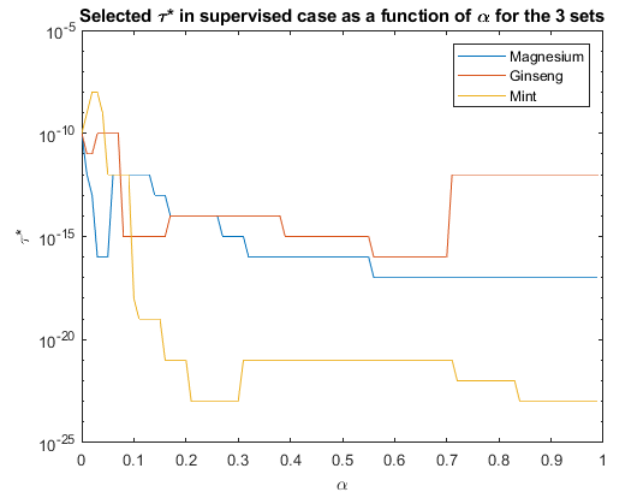


Figure 8.36

8.4.2 Artificial Defective and non-defective data available

Another experiment that we can performed is the same as the previous one, but where the real defective objects are replaced by synthetic defective objects. The synthetic defaults are added to the normal images. The added defaults are simply white rectangles of the same dimensions as the real defaults.

We can make the same figures as for the previous experiment. These figures are shown at the figures from 8.40 to 8.44. For the selected best parameters, we can see that the choice is much less organized. Indeed, the choice of the optimal parameters is very unstable compared to the value of α . Though the good order of magnitude is kept for the choice of C^* according to the data sets, the value of the cuts is globally higher for this experiment.

If we take a look at the figures 8.42 to 8.44, we can observe very different behaviours than for the first experiment. The first remark is that the learning curve is much better in this case than for

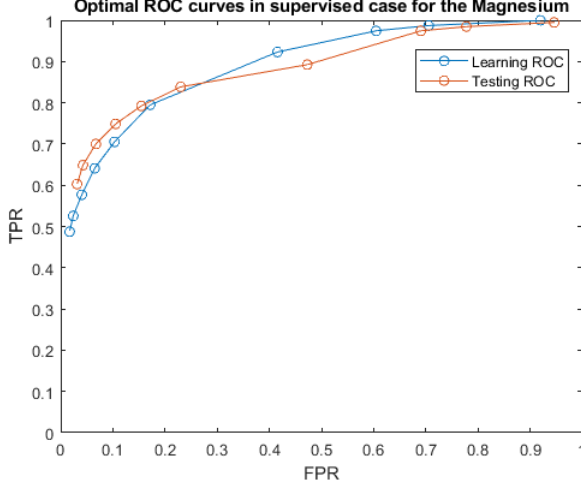


Figure 8.37

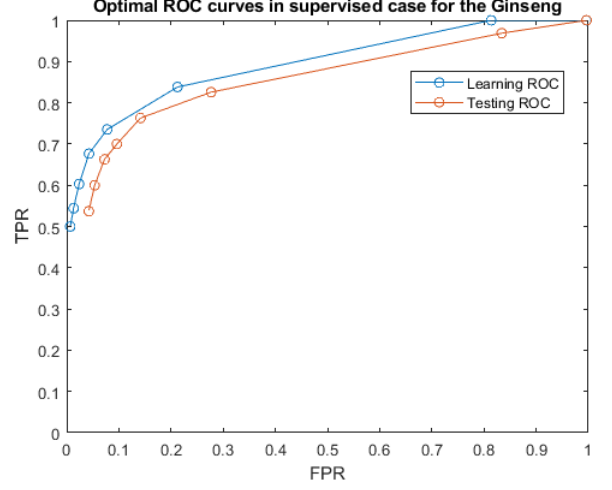


Figure 8.38

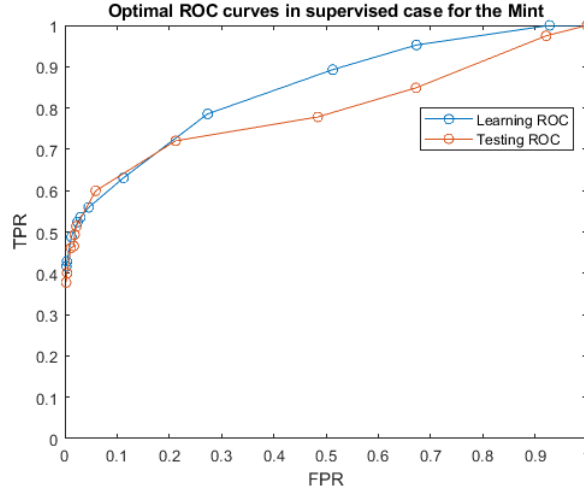


Figure 8.39

the real supervised case. We can explained it by the fact that the synthetic defaults are much more simple to handle with than the real ones. This is especially true for the cut test, since the abnormal borders are easily found because of the high difference of the intensities between the object and the white colour. Obviously, this simplicity is a lure, because this training loses its interest in using simpler defaults. Therefore the roc for the testing set of images is much worse than for the first experiment.

We can see here the need of exploring as much different types of anomalies as possible during the training phase. This is true even if we get the supervised training.

8.4.3 Comparison with unsupervised performance

Now, we can compare the unsupervised performance with the supervised one computed in the experiment 1. We can observe the results in the figures 8.45 to 8.47, for the three data sets. We can observe, for the three data sets, that the method, for quite same value of $\lambda \approx 2$, the roc curves are almost the same as the curve in the supervised case.

A remark can be done on these results. The testing roc curves is in fact a lower limit of

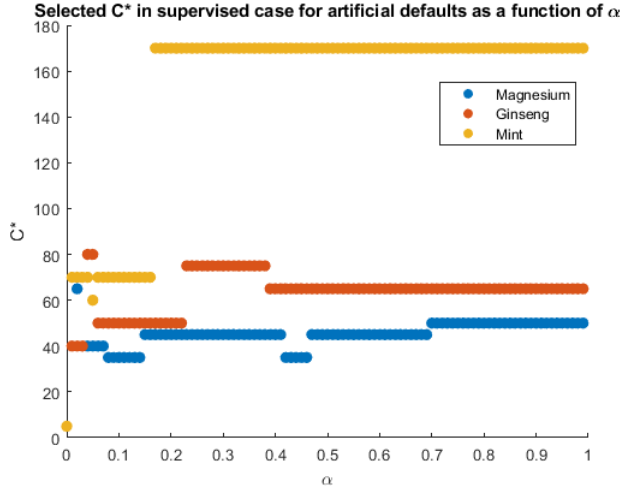


Figure 8.40

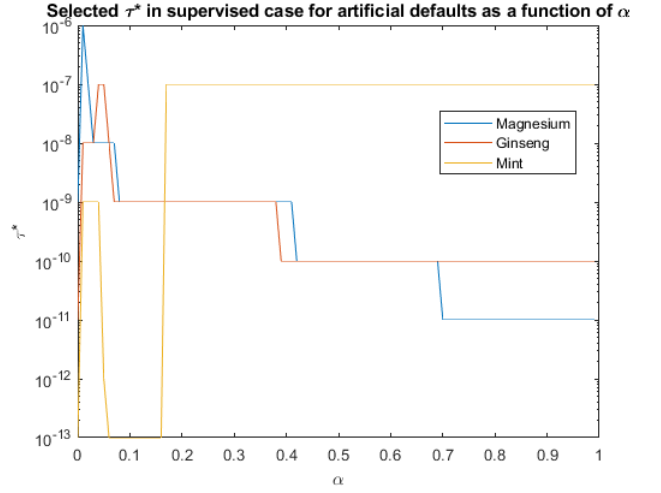


Figure 8.41

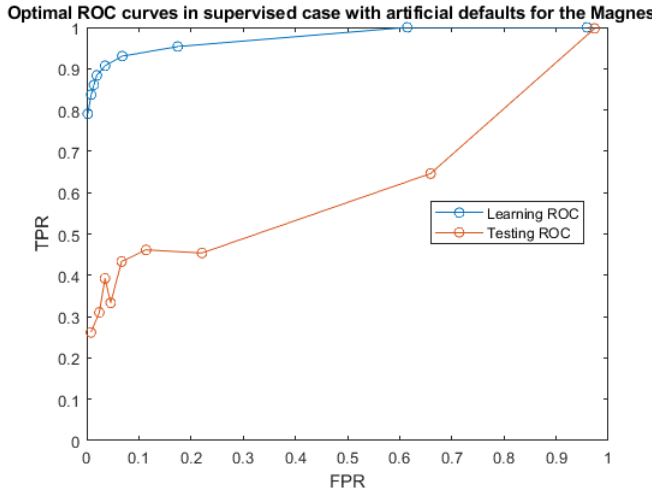


Figure 8.42

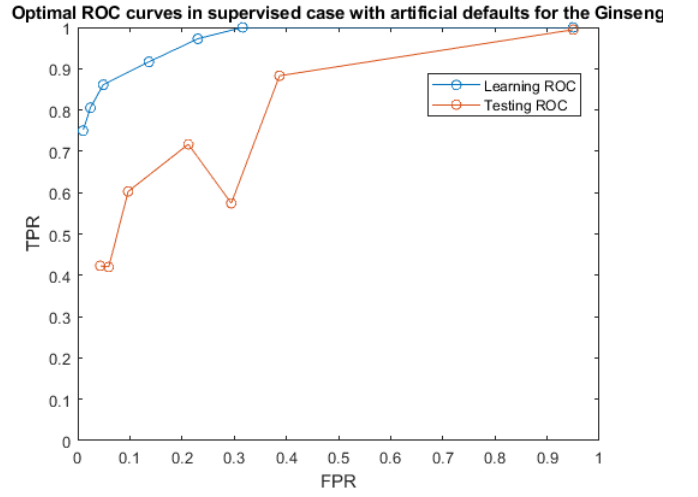


Figure 8.43

the real one. Indeed, our assumption from the beginning is that only few defaults are observable. So for the training, only few defaults cases can be used. So, from a machine learning point of view, the class of anomalies is less represented in the training than in the testing phase. However, even if we used the learning roc curve, the unsupervised method gets close to it, and this is a really good clue of optimality.

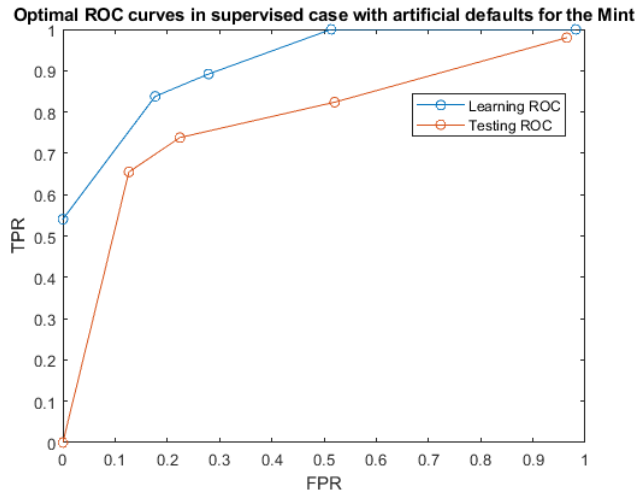


Figure 8.44

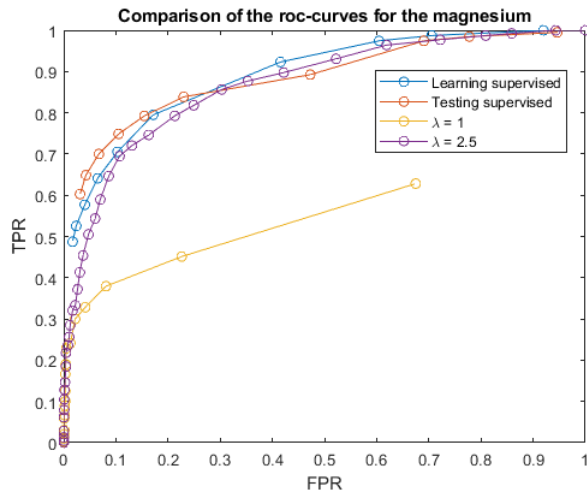


Figure 8.45

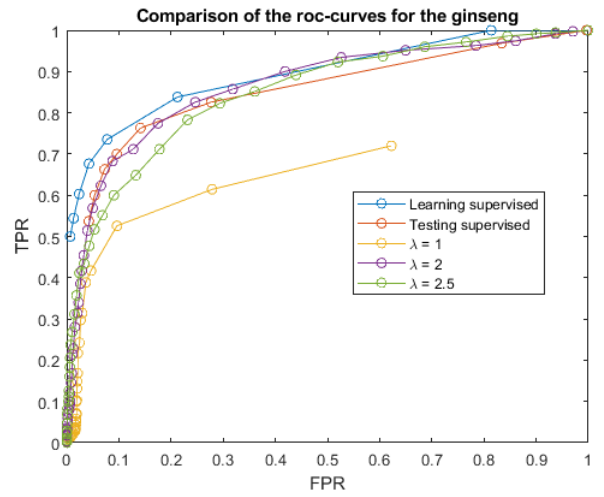


Figure 8.46

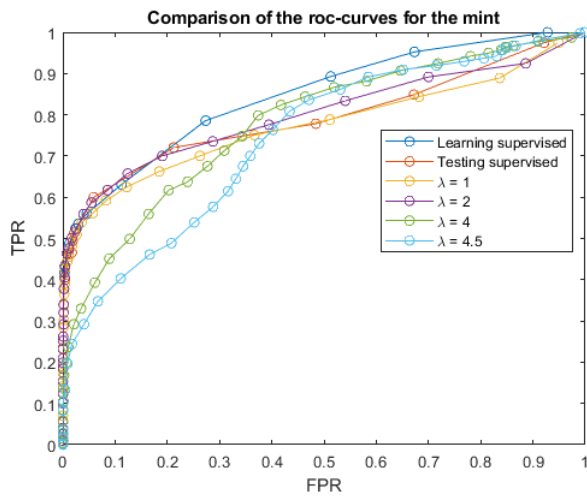


Figure 8.47

Chapter 9

Groups optimization

In this chapter, we will introduce a theoretical suggestion for further improvements. Until now, we have created the tests and grouped them in a random way. The randomness of the tests is very interesting, since it allows the tests to be spread quickly over the whole space of the image. However, we have seen in the first chapters that the principle of the ferns is based on the following assumption : the ferns are independent, one from each other. As it has been already said, this is absolutely not the case, but this works quite well in practice.

The purpose of this chapter is then to introduce a way to make this assumption as true as possible, i.e to make the ferns as independent one from each other as possible. The author of [13] claimed that the mutual information could be interesting, but this search is not pursued in practice. Therefore, this side of the problem was not studied in the article.

Nevertheless, this improvement could be interesting to investigate. In presence of a good test, quite good or bad ferns can be created from the generated tests. Some tests are not as useful as other ones, but the way to group them can also be a point to precise. Firstly, we remind the concept of mutual information and why this concept is useful in our case. Then we see how to express the problem and a possible way to solve it in practice.

9.1 Mutual information

We introduce the proprieties of the mutual information thanks to [6].

The mutual information $MI(T_i; T_j)$ measures the global (linear and non-linear) dependence between the random variables T_i and T_j , which are the variables associated to 2 tests. The mutual information can also be written in this form : $MI(T_i; T_j) = H(T_i) - H(T_i|T_j)$. Intuitively, the mutual information is the remaining quantity of information given by T_i when we know T_j . Some important proprieties can be drawn from the formula :

- MI is symmetrical : $MI(T_i; T_j) = MI(T_j; T_i)$
- T_j does not bring more information on T_i than T_i contains : $MI(T_i; T_j) \leq H(T_i)$
- The information that T_i brings on T_i is the information in T_i : $MI(T_i; T_i) = H(T_i)$
- If T_i and T_j are independent, $MI(T_i; T_j) = 0$

Since we have a good quality criterion for the dependence of two random variables, we can now describe our optimization problem.

The last thing to pointed out is the evaluation of the mutual information. Since we do not have the exact distribution of the realizations of the test, we need to approximate it with the realizations that we have. In order to stay consistent with the principle of the random ferns, the frequency histograms will be used in order to approximate the distribution of the probabilities.

So, when we evaluate the MI between the tests T_i and T_j , we must construct the frequency histograms of the single test T_i , the test T_j and of the joint frequency histograms of T_i and T_j .

We can take back the definition of the h_i which is the frequency histogram of the test i . We can define the probability of the test i of being equal to 0 or 1 :

$$\hat{P}_i^0 = \frac{h_i[0]}{N} \quad \text{and} \quad \hat{P}_i^1 = \frac{h_i[1]}{N} \quad \forall i$$

We must also define the joint histogram $h_{i,j}$ as the histogram with 4 entries, where each entry corresponds to the probability for T_i to be 0 or 1, and for T_j to be 0 or 1. So we can define the associated probability :

$$\hat{P}_{i,j}^{f_i, f_j} = \frac{h_{i,j}[2f_i + f_j]}{N} \quad \text{with} \quad \sum_{f_i=0}^1 \sum_{f_j=0}^1 \hat{P}_{i,j}^{f_i, f_j} = 1 \quad \forall (i, j)$$

The approximation of the mutual information can be written as :

$$\hat{MI}(T_i; T_j) = \sum_{f_i=0}^1 \sum_{f_j=0}^1 \hat{P}_{i,j}^{f_i, f_j} \log \frac{\hat{P}_{i,j}^{f_i, f_j}}{\hat{P}_i^{f_i} \hat{P}_j^{f_j}}$$

In the last equation, we must forget the probabilities which are equal to 0. Indeed, we consider that the realizations that are not encountered during the construction of the frequency histograms are impossible.

9.2 Combinatorial optimization problem

In this section we will see how the optimization problem could be formulated. As a reminder, the goal is to make the best M combinations of S tests, such that $M \times S$ tests have been randomly generated. Let's define the test T_i for $i \in [1 : M \times S]$ and the fern F_j for $j \in [1 : M]$. We can define a combinatorial problem in defining the variable x_{ij} such that :

$$x_{ij} = \begin{cases} 1 & \text{if } T_i \in F_j \\ 0 & \text{otherwise} \end{cases}$$

Some constraints on x_{ij} must be set. First, we need taking one and only one time each test :

$$\sum_j x_{ij} = 1, \forall i$$

Secondly, each fern must contain exactly S tests, so we get the following constraint :

$$\sum_i x_{ij} = S, \forall j$$

Now we must define the objective function. A possible approach would be to minimize the mutual information between each pair of tests that do not belong to the same fern. We can express the complete problem like :

$$\begin{aligned} & \min \sum_i \sum_{j \notin N_i} MI(T_i, T_j) \\ & \text{s.t.} \quad \begin{cases} \sum_i x_{ij} = S, \forall j \\ \sum_j x_{ij} = 1, \forall i \\ x_{ij} \in \{0, 1\}, \forall (i, j) \end{cases} \end{aligned}$$

Where N_i is the fern containing the test i .

We can observe that the problem above combines the difficulties of the non-linear objective function due to the logarithm and the binary variables x_{ij} . Therefore it seems very complicated to find an analytic way to solve this optimization problem. However, trying all the possibilities is completely intractable, due to exponential complexity of a binary optimization problem. Indeed, if we tried all the possible permutations of the S tests for each of the M ferns, the number of evaluations of the objective function will be $\frac{(MS)!}{M! \times (S!)^M}$. Since the objective function requires $(M-1) \times S \times (M \times S)$ evaluations of the MI , the total number of MI evaluations is :

$$L(M, S) = M(M-1)S^2 \times \frac{(MS)!}{M! \times (S!)^M}$$

The numerator comes from the permutation of all the tests in all the ferns. The second term in the denominator comes from the permutation of the S tests inside the M ferns. The first term in the denominator comes from the possible permutation of all the entire ferns. For example, if we use the half of the number of ferns and test per fern that we usually use ($S = 4$ and $M = 8$), the number of evaluated objective function is 5.9×10^{19} , which is totally unrealistic.

9.3 Greedy local search

Therefore, we would better to search an approximation algorithm, that simplifies the initial problem. A possible way to do it is the greedy local search. The principle of this approximation comes from [14]. The idea of this method is to begin with an empty set, and constructing gradually the ferns with the $M \times S$ pre-computed tests. In our case, the greedy algorithm can be expressed like this.

We begin with M empty sets. At each step, we pick up at random one test. Then, we compute the M means of the mutual information between the new test and the tests already picked up in each fern. A local optimal solution can be found in this way, we add the current test to the fern that have the maximum mean of mutual information between this test and all the ones which are inside this fern. Since in this way, we minimize the mean of mutual information between this test and the tests in the other ferns. Obviously, while the iteration is under the number of available ferns, we can add the current test to the empty set. We change the sum into the mean for the computation of the MI , in order to not penalize the fern which are more filled than the other. Therefore we can express the number of MI that is computed. For each test, we need to compute the mutual information between this test and each test that is already chosen. So the total number of evaluation of the MI is

$$L'(M, S) = \sum_{k=M}^{M+S-1} k = (M + \frac{S-1}{2}) \times S$$

We can observe that the complexity is not exponential anymore though the price to pay is a search of a sub-optimal solution.

We can summarize the idea in the algorithm 9.1. The following algorithm shows the procedure of the greedy local search. A particular remark can be noticed. The approximation of the MI has obviously a big impact on the performance of the local search. In practice, as explained in [12], the approximation of the MI thanks to the frequency histograms are not the best way to evaluate it. A better way to approximate it can be the estimation of the mutual information developed by Kraskov [5].

Algorithm 9.1 Greedy local search

Require: S, M

Create M empty sets F_j

Compute $h_i, \forall i$

for $i \in [1 : M \times S]$ **do**

 Compute $h_{i,j}, \forall j$

 Compute $j^* = \operatorname{argmax}_j \frac{\sum_{k \in F_j} \hat{M}I(T_i, T_k)}{|F_j|}$

$S_{j^*} \leftarrow S_{j^*} \cup T_i$

end for

Chapter 10

Conclusion

As a reminder, the initial problem was to find a way to detect anomalies in a bunch of images. The required specifications of the algorithm are the unsupervised case and the basic algorithm that comes to mind firstly is the random ferns. Although this method works well in a supervised classification case, this method suffers from a lack of performance in the unsupervised context of the anomaly detection.

The first question to be asked is the relevance of the initial test. We saw that a more specific description of test to the data set and the anomalies that we are facing, gives best performance. Indeed, some tests gives better results on some data sets and for some type of anomalies. We have so described three binary tests developed from well-known features as HOG, but also from the new cut test.

The following question appeared : how do we choose the best test for our data set? We have proposed an unsupervised method in order to identify it, that we validated for some decisions as the optimal parameter C of the cut-test, for the application or not of the mathematical morphology and for the optimal test. We see that the optimization gives very close results from the manual tuning of the parameters and tests. However, we see that the choice of the best decision is really dependent of which definition we give to the spreading of the histograms. Though the choice of the best test was introduced, the general decision stays complicated in practice. The choice of the representation of the spreading and the good gap to give to the weight λ stay complicated to catch.

However, the choice of the parameter of the cut test in the unsupervised way are very close compared to the value that we obtain in using a supervised training.

Finally, the last chapter was dedicated to the optimization of the assemblage of the ferns. A possible way to do it is using the mutual information in order to make the ferns as independent as possible, which makes the naive Bayesian assumption more accurate.

Some further improvement can be brought. The first one is the adding of tests. The three presented tests are sufficient in order to analyze correctly the 3 data sets that we used. However, some other features can be useful for other typical data, as textures. Some features are dedicated to these types of data as the co-occurrence matrix. Then this feature must be turned into a binary test in order to be inserted in the algorithm.

However we must still pay attention to the cost of adding a new test. Indeed, the time of the learning and optimization phase(i.e the time needed to compute the frequency histograms and choose the best one) is proportional to the number of features. So a good idea could be a compromise between the number of tests and their specificity.

During this work, some general remarks can be done. The first one is the special difficulty of the handling of the anomaly detection. This part of machine learning has very difficult assumptions to counteract. For example, the classification cannot be handled as usually with the same methods, since there is a too small proportion for the class of anomalies, and because the similarities between the defective and the non-defective objects are too high. Moreover, the unsupervised assumption makes the problem even more difficult, since we cannot "wait and see" if the decisions that we took are the best ones.

Bibliography

- [1] <https://opencv.org>.
- [2] <https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5>.
- [3] <https://www.learnopencv.com/histogram-of-oriented-gradients>.
- [4] <https://nl.mathworks.com/help/images/morphological-dilation-and-erosion.html>.
- [5] Peter Grassberger Alexander Kraskov, Harald Stoeckbauer. *Estimating Mutual Information*. 2003.
- [6] Olivier Peirera Benoit Macq, Jérôme Louveaux. *INGI2348 -Information theory and coding*.
- [7] Ming-ting Sun Fan Chen, Zicheng Liu. *Anomaly detection by using random projection forest*.
- [8] Zhuang Haralick, Sternber. *Grayscale Morphology*.
- [9] Cécile Hauteceur. *Impact of coherence and occurrence in the training of CNN for anomaly detection*. 2018.
- [10] Václav Hlaváč(Czech Technical University in Prague). *GrayScale Mathematical Morphology*.
- [11] Václav Hlaváč(Czech Technical University in Prague). *Mathematical Morphology*.
- [12] Michel Verleysen John Lee. *LELEC2870 - Machine Learning : regression, dimensionality reduction and data visualization*.
- [13] Lepetit Fua (EPFL) Ozuysal, Calonder. *Fast Keypoint Recognition using Random Ferns*.
- [14] Laurence A. Wolsey. *Integer programming*. 1998.
- [15] Yasuharu Yamada. *Flood extent detection in paddy area and future plan of disaster information sharing platform in rural areas*. 2018.

