University of Minho
Engineering School

Vanessa Catarina Costa Pessoa

Exploration of Machine Learning Techniques for Automatic Optical Inspection

Master's dissertation in Engineering and Management of Information Systems

Work done under the guidance of/from

Professor Henrique Manuel Dinis dos Santos

And on the premises of

Amkor Technology Portugal

October 2018

DECLARAÇÃO

Nome: Vanessa Catarina Costa Pessoa

Endereço eletrónico: a69178@alunos.uminho.pt

Telefone: 913611788

Bilhete de Identidade: 14591353

Título da dissertação: Exploration of Machine Learning Techniques for Automatic Optical Inspection

Orientadores:

Professor Doutor: Henrique Manuel Dinis dos Santos

Ano de conclusão: 2018

Mestrado Integrado em Engenharia e Gestão de Sistemas de Informação

É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA DISSERTAÇÃO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE.

Universidade do Minho, __18__/__10__/_2018_

Assinatura:

Acknowledgments

First, I want to thank my parents, João and Isabel, for all the support, patience, effort and courage that they always have given to me. Without them I would never be here today.

To my grandparents, António and Maria, that never let me give up and always supported me no matter what.

To my advisor, professor Henrique Santos, for all the support, encouragement and disposition to help me.

And of course, I also want to thank the entire Amkor Technology Portugal Staff specially to Engineer Jorge Ferreira and to Engineer Teresa Silva for providing me this opportunity and for all the patience, kindness and support in this journey.

Thank you all for making this possible!

RESUMO

Atualmente, a indústria dos semicondutores enfrenta problemas sérios em relação à deteção das anomalias nas wafers, devido à diminuição do tamanho dos dispositivos, o que se traduz numa diminuição do espaço disponível para dispor os componentes de que necessita, o que torna o processo de inspeção das wafers algo complexo, especialmente se esta tarefa for executada manualmente.

A execução deste projeto de pesquisa consiste no estudo da viabilidade da implementação de uma solução para a deteção dessas anomalias de forma automática numa linha de produção na indústria dos semicondutores, assim como idealizar como será a conceção e implementação de uma ferramenta deste tipo, bem como perceber de que forma é que os problemas que a indústria se depara podem ser reduzidos, capacitando inclusive a solução para o processamento de grandes quantidades de dados.

Assim sendo, o objetivo principal deste projeto é o de desenvolver uma solução que reconheça padrões nas imagens anómalas resultantes do processo de inspeção ótico das wafers e que sugira uma classificação dessa anomalia ao utilizador, usando técnicas de processamento de imagem para extrair informações relevantes nas imagens processadas e algoritmos de Machine Learning para a classificação automática dessas informações. Com isto, podemos melhorar as taxas de erro ocorridas nas classificações dos operadores e ajudar os profissionais de Sistemas de Informação nas suas análises, facilitando a tomada de decisão, o que se pode traduzir em poupanças de tempo e dinheiro e na realocação de recursos necessários a outras áreas.

Para a realização deste projeto, será seguida a abordagem metodológica Design Science Research, que ajudará na criação dos artefactos do projeto e na realização de uma pesquisa com o máximo rigor.

Palavras-chave: *Deteção de anomalias, Semicondutores, Processamento de imagens, Inspeção Ótica Automática*

ABSTRACT

Nowadays, the semiconductors industry is facing a huge problem concerning anomaly detection, due to the decreasing of device sizes, which means that they have less space to place their components, making product inspections even more difficult, especially when this kind of tasks are made manually.

This research project consists of studying the viability of implementing a solution for detecting anomalies automatically in a factory’s production line in the semiconductors industry, as well as thinking about how to develop this kind of tool, how to implement it, perceive how to reduce this kind of problems and enable the solution to handle the processing of large amounts of data.

The main goal of this research is to develop a solution that could recognize patterns in anomalous images resulting from the wafer’s optical inspection process and that suggests a classification to the user, using image processing techniques to extract relevant information in the images processed and Machine Learning algorithms to classify this information automatically. With this, the error rates given by the manual inspection can be improved and this can also help the Information Systems team on their data analysis, improving the decision-making process, saving time and money and help in the reallocation of the necessary resources in other areas.

As a methodological approach, Design Science Research will be used which will help in the project's artifacts creation and in doing a research work with a maximum rigor.

Keywords: *Anomaly Detection, Semiconductors, Image Processing, Automated Optical Inspection*

Contents

List of Acronyms xv

1. **Introduction** 1
   - 1.1. **Context** 1
   - 1.2. **Project's proponent company – Amkor Technology** 2
   - 1.3. **Main objectives of this dissertation** 2
   - 1.4. **Organization of the dissertation** 3

2. **Image Processing Techniques** 6
   - 2.1. **Colorspaces** 6
   - 2.2. **Edge Detection** 7
   - 2.3. **Histograms** 9
   - 2.4. **Image thresholding** 11
   - 2.5. **Find contours** 12
   - 2.6. **Segmentation** 13
   - 2.7. **Smooth Images** 14
   - 2.8. **Pattern Matching** 15
   - 2.9. **Image pyramids** 16

3. **Anomaly detection** 18
   - 3.1. **Wafer's failure detection** 18
   - 3.2. **Related projects (Anomaly Detection)** 19
   - 3.3. **Machine Learning** 26
   - 3.4. **Methods for Automatic Learning** 27
     - 3.4.1. **Learning methods** 27
     - 3.4.2. ***Decision Trees*** 28
     - 3.4.3. **Classification rules** 29
     - 3.4.4. **Inductive Logic Programming** 29
     - 3.4.5. **Support Vector Machines** 30
     - 3.4.6. **Bayesian methods** 31
     - 3.4.7. **Artificial Neural Networks** 33
     - 3.4.8. **Cascading Classifiers** 35
   - 3.5. **Performance metrics and model evaluation** 36

3.5.1. **Confusion matrix** ... 36
3.5.2. **Accuracy (ACC)** ... 37
3.5.3. **Precision (PPV)** ... 38
3.5.4. **Specificity (TNR)** ... 38
3.5.5. **False Positive Rate (FPR)** ... 38
3.5.6. **False negative rate (FNR)** ... 38
3.5.7. **Recall (Sensitivity, True Positive Rate)** ... 39
3.5.8. **Cohen's Kappa coefficient** ... 39
3.5.9. **F-measure** ... 39
3.5.10. **Receiver Operating Characteristic curve (ROC curve)** ... 40
3.5.11. **Precision-Recall curves** ... 40
3.6. **Conclusion** ... 42

4. **System implementation** ... 44
4.1. **Wafer failure detection system** ... 44
4.2. **Solution design** ... 46
4.3. **Architecture** ... 48
4.4. **Software specification** ... 49
4.5. **Implementation** ... 50
4.5.1. **Used algorithms for detection and classification** ... 50

5. **Results** ... 52
5.1. **Experiences carried out** ... 52
5.2. **User's feedback** ... 61
5.3. **Results conclusions** ... 62

6. **Final considerations** ... 64
6.1. **Final conclusions** ... 64
6.2. **Future work** ... 65

Appendix ... 74
a) **AOI Classifier portal user's manual** ... 74
b) **AOI Classifier samples gallery page** ... 77
c) **AOI Classifier recalibration page** ... 77
d) **Advantages and disadvantages** ... 78

List of tables

Table 1 - List of Acronyms xv
Table 2 - Camteks vs iFocus: images and alarms 45
Table 3 - Software specification 49
Table 4 - Reclassification's general information 53
Table 5 - Metrics targets 54
Table 6 - Pie chart legends 55
Table 7 - Performance evaluation for product AOI1DL2-PM8937-X-P06 56
Table 8 - Performance evaluation for product PM8953-X-P02 58
Table 9 - Performance evaluation for product AADI-X-P05 60
Table 10 -Project’s implementation approach advantages and disadvantages 78

List of figures

Figure 1 - Changing image colorspace from BGR to grayscale ... 6
Figure 2 - Object Tracking by color ... 7
Figure 3 - Detecting Edges in an image ... 8
Figure 4 - Detecting circles in an image ... 9
Figure 5 - Detecting existing lines in an image ... 9
Figure 6 - Creating histograms from images ... 10
Figure 7 - Equalizing image histograms ... 10
Figure 8 - Applying thresold ... 11
Figure 9 - Appplying Adaptive Threshold ... 12
Figure 10 - Select a Bounding Rectangle ... 13
Figure 11 - Smoothing images ... 15
Figure 12 - Matching a template ... 15
Figure 13 - Gaussian pyramid representation ... 16
Figure 14 - Representation of a decision tree ... 28
Figure 15 - Graphical representation of a Support Vector Machine for two classes ... 31
Figure 16 - Non-Causal Bayesian Network Example ... 32
Figure 17 - DBN model ... 33
Figure 18 - An example of a ANN with a single hidden layer ... 35
Figure 19 - Confusion matrix definition ... 37
Figure 20 - ROC curve representation example ... 40
Figure 21 - Precision-Recall curve representation example ... 41
Figure 22 - Process "As is" ... 46
Figure 23 - Process "To be" ... 47
Figure 24 – Architecture ... 48
Figure 25 - Portal's login window ... 74
Figure 26 - Portal's information selection window ... 74
Figure 27 - Portal's assistant buttons ... 75
Figure 28 - Portal's main menu window ... 75
Figure 29 - Portal's detractor selection menu ... 76
Figure 30 - Portal's create positive samples window ... 76
Figure 31 - Portal's image gallery page ... 77
Figure 32 - Portal's automatic classifier's recalibration page ... 77

List of Acronyms

The table below briefly describes the definitions of the main acronyms contained in this document.

Table 1 - List of Acronyms

Acronym	Definition
AOI	The A*utomated Optical Inspection* (AOI) consists of performing the visual inspection in an automated way to the manufacturing printed circuit boards. This process takes place through a camera that, autonomously, runs the entire wafer to be tested over a scan throughout its area, capturing images and looking for catastrophic failures and quality defects. This is a non-contact test and is therefore widely used in a number of industries, including the semiconductor industry [2].
WLP	*Wafer-Level Packaging* (WLP) is a key technology in the semiconductor industry, especially when produced on 300mm carriers because of the performance and cost advantages it offers for smartphones, tablets and other applications that require high functionality and low energy consumption in a short allocation space [3]. Basically, the WLP corresponds to the packaging of an integrated circuit while it is still part of the wafer, instead of being partitioned into individual circuits and then being packaged [4].
WLPFO	*Wafer-Level Fan-Out* (WLPFO), is a technology that consists of a standard *Wafer-Level-Packaging* (WLP), designed to provide a solution for semiconductor devices that require a higher level of integration and better thermal and electrical performance [5].
eWLB	*Embedded Wafer-Level Ball Grid Array* (eWLB), is an integrated circuit packaging technology. Their interconnections are applied on an artificial wafer made of silicon chips and a casting compound [6].
HDFS	Distributed file system that stores data on machines within the cluster. Corresponds to the *Hadoop* file system.
SVM	*Support Vector Machines* (SVM) are supervised learning models with associated learning algorithms that analyze the data used for classification and regression analysis.
k-NN	*K-nearest neighbor (k-NN)*, is a non-parametric method used for classification and regression, where an object is assigned to the most common class among its nearest k neighbors.

1. Introduction

This chapter aims to frame the context in which this project is inserted, as well as the reasons that led to the present dissertation's development, including their main objectives, importance and the methodological approach adopted.

1.1. Context

Nowadays the trend towards the size of electronic devices is to decrease more and more. Thus, companies dealing with the manufacturing of semiconductors have encountered some challenges arising from this trend since, by decreasing the size of the devices, chips will need to have a greater number of layers and consequently go through more processing steps.

As the target area to be inspected is decreasing, defects will also become smaller, which leads to difficulties regarding the optical resolution limit [1]. Besides that, the manual classification that is performed, being a rather slow method, does not stop the occurrence of false results, since these depend on the attention, concentration, experience and knowledge of the operator. So, it is essential to pay attention to all the details in the images resulting from the Automatic Optical Inspection (AOI) process, because small failures can induce huge problems in the future.

Therefore, for the companies dealing with nano-scale devices, it is necessary to reduce to the maximum, in its inspection process, both the error obtained by the machines performance, and the errors associated to manual inspection, since fewer failures presuppose more profits, and all the processes in the production line are based on the optimization factor, with the purpose to get the best at all levels. Moreover, as it might be expected, a company of this size generates large amounts of data daily, so it is necessary to acquire existing storage and processing solutions, taking advantage of them by using it as best as possible.

Then, the solution to these problems is to create an automatic way of detecting, analyzing and classifying the existing anomalies, so that the computer, upon wafer's manual inspection, can suggest the type of anomalies presented to the operator, and whether, according to the acceptance or rejection criteria, the results of the automatic reclassification would be approved or not.

In terms of studies and techniques to be used in this project stands out the research, the analysis of the used procedures and the criticism of the obtained results in scientific papers

that deal with topics related to this dissertation, as well as the study and analysis of the bibliography appropriate to the topic, such as the follow-up of the methodological approach *Design Science Research* applied to Information Systems projects implementations, in the sense that this approach will help in formulating the project requirements, as well as measuring the performance of each of them correctly under the scientific point of view.

1.2. Project's proponent company – Amkor Technology

At the beginning of this journey, the project's proponent company was called Nanium S.A., but after some time the organization was bought by Amkor Technology Inc, so this work will always be referred to as having been proposed by Amkor Technology Portugal.

Amkor Technology Portugal is a company located in Vila do Conde, Portugal, which operates in the manufacturing of electronic components, particularly in the semiconductor industry, having inherited from Nanium S.A. an extensive experience in the manufacturing and processing of wafers and carriers of 300mm. It has established itself as a global leader in high-volume Wafer-Level Fan-Out (WLFO) technology, in addition to Amkor's vast experience worldwide in other semiconductor technologies.

Since the company operates daily with the manufacture of circuit boards and with very small measurement units (at the nanometer scale), one of the company's concerns is to minimize the existence of errors when inspecting wafers, due to the margin of error obtained as a result of the machines normal operation that make the automated optical inspection (AOI), as well as reduce the margin of error in the anomalies manual classification performed by the factory's production line operators.

1.3. Main objectives of this dissertation

The main objective of this dissertation is the development of a decision support tool to be used in the wafer-level visual inspection. With this development it will be possible to validate the inspections performed by the operators, reduce the error rates associated to classification mistakes, as well as establish a set of metrics that can validate the quality of the automatic classifiers. Consequently, algorithms and techniques for segmentation, automatic detection and classification of wafer anomalies in images were studied. The tool must be

prepared to handle the increasing amount of data it processes, thus, the use of Hadoop was investigated and tested throughout the execution of this project.

Therefore, this dissertation has the following main goals:

I. To investigate existent frameworks and identify the ones to use;

II. To study and analyze image processing techniques to extract the anomalies from images;

III. To study and analyze different object detection algorithms;

IV. To develop a tool to detect anomalies in wafers automatically;

V. To discuss the results of the project, by comparing it with other research projects.

1.4. Organization of the dissertation

This dissertation is structured in 6 chapters.

The first chapter presents the main needs that led to the proposed implementation, in which context these occurred as well as the proposed objectives for the project to be developed within the scope of this dissertation.

The second chapter describes the most common image processing techniques used and referred by the literature.

The third chapter presents the state of the art, which explains some fundamental aspects for the project's understanding, such as how wafer's anomaly detection is made, the study of related projects already implemented and how those projects will help to choose the best techniques for the implementation to be developed. This chapter also describes the methods for automatic learning and the performance metrics to evaluate models and algorithms.

The fourth chapter presents the considerations about the new proposed architecture to be followed, how will be integrated in the actual process as well as the solution design and the software specifications.

The fifth chapter shows the implementation validation results measured by some of the previously studied metrics and describes the analysis about the obtained results.

The sixth and last chapter presents the conclusions taken from this implementation and the suggestions for future work.

2. Image Processing Techniques

In this section will be presented different image processing techniques, allowing filtering and object segmenting for later detection and classification, as well as changing some properties of these images (such as noise, brightness and background). Therefore, the combination of several of these techniques will have to be considered and adapted later, according to the desired solution [2]. Many of the examples are based on OpenCV documentation (library of programming functions aimed at computer vision), due to this being an open-source library which allows the possibility of making some free tests to give an overview of the existing functions and that may be useful for future developments.

2.1. Colorspaces

For many image processing applications, in some cases color information does not help to identify borders or other features. In this case, the literature suggests changing the images colorspace for grayscale, because if we work with color images, we are adding extra complexity to the problem and we can still reduce the processing speed of these images. Figure 1 shows the difference between a color image and a grayscale image.

Figure 1 - Changing image colorspace from BGR to grayscale (transcript from [3])

The techniques used in Colorspaces may also be important for Object Tracking, since, if we can represent a certain color, in OpenCV (library of programming functions aimed at computer vision) for example, we just need to convert the BGR image to HSV [4], apply a threshold to the image in a range for the selected color, (that, as we can see in Figure 2, the selected colors were blue, red and green), and then we can extract the object of that same color.

Figure 2 - Object Tracking by color
(transcript from [5]*)*

2.2. Edge Detection

Canny Edge Detection was developed by *John F. Canny* in 1986 and is a popular Edge detection algorithm. This algorithm is characterized by having multi-stages, which are, noise reduction, find the image intensity gradient, *Non-maximum Suppression* and *Hysteresis Thresholding.*

In *Edge detection*, image noise reduction is a rather important stage, as this factor can compromise the detection accuracy. So, for this, *the user can* apply a *Gaussian Filter* and specify an intensity. This filter takes the neighborhood around the pixel and finds its Gaussian weighted average [4]. Therefore, it's not recommended to apply too strong values in this filter since it tends to blur the edges and the main purpose is just to reduce the noise.

After smoothing the image, the *Canny Edge Detection* method filters it with a Sobel kernel to obtain the first derivative in the horizontal (G_x) direction and in the vertical (G_y) direction. From these two images, the gradient and the direction of the edge for each pixel is found according to the following formulas [4]:

$$Edge_{gradient}(G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

After obtaining the gradient's magnitude and direction in the previous step, the image is completely swept away to remove the unwanted pixels that may not be part of the edge. Thus, for each pixel, it is verified if it corresponds to the local maximum of its neighborhood in the direction of the gradient (this is always perpendicular to the edges). If the pixel is considered the maximum local of its neighborhood it passes to the next phase, otherwise this one is suppressed, that is, its value will be zero.

In the *Hysteresis Thresholding* stage, it is decided which pixels will be considered edges and which will not be considered as such. To do this, the user, when invoking this method in *OpenCV*, refers to what maxVal and minVal he wants to implement, so any gradient edges of intensity greater than maxVal are classified as "sure-edge" and those that are below the minVal will be classified as non-borders and therefore will be discarded. All borders between the two thresholds will be classified according to their connectivity, that is, if they are linked to “sure-edges”, they will also be considered borders, otherwise they will also be discarded. At the end of this whole process, what we get is the strongest edges present in the processed image.

Figure 3 shows the implementing result of the *Canny Edge Detection* technique, applied to a grayscale image.

Figure 3 - Detecting Edges in an image
(transcript from [6]*)*

However, this is not the only technique that can be used for this purpose. The *Hough Transform* consists in a technique used in image analysis, computer vision and digital image processing [7] to find imperfect object instances within a certain class of forms through a voting procedure. Traditionally, classical Hough Transform was concerned with the identification of lines in the image, however, later, this approach was extended to the identification of positions of arbitrary forms, like circles or ellipses. Although this technique was originally created by *Paul Hough* in 1962 [8], currently people use the patent created by *Richard Duda* and *Peter Hart* in 1972 [9], now known as "generalized *Hough Transform*" and that was popularized in the *Computer Vision* community through the article "Generalizing the *Hough transform* to detect arbitrary shapes "by *Dana H. Ballard* in 1981 [10]. Thus, *Hough Transform* is a popular technique to detect any shape since you can represent it in a mathematical form.

OpenCV uses the method *Hough Gradient* which uses the gradient information of edges to detect circles in images [4]. In addition, for lines detection in images, this software also has the basis of the existence of bright spots in some image places, insofar as these indicate that

there are the parameters of the possible present lines, iterating on the values of þ and Ɵ provided by the user. þ and Ɵ are used to represent any line (as þ, Ɵ), so the method uses them to create a 2D matrix, keeping the values of both parameters and defining their size according to the required precision.

Figure 4 and Figure 5 show an example of detecting lines and circles using the above-mentioned methods.

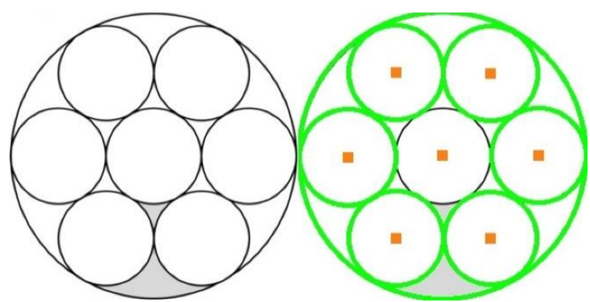


Figure 4 - Detecting circles in an image (transcript from [11]*)*

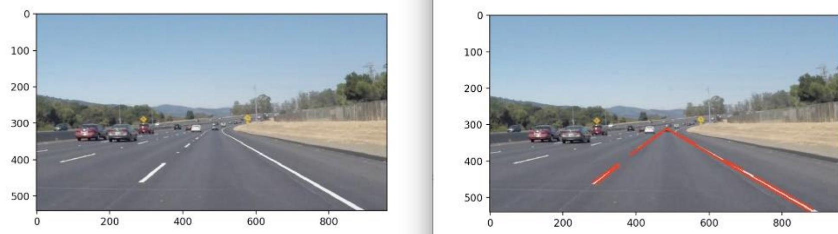


Figure 5 - Detecting existing lines in an image (transcript from [12]*)*

2.3. Histograms

A histogram consists of a graph that gives a general idea of the color intensity distribution of an image, where the X axis shows the pixel values and the corresponding number of pixels in the image are shown on the Y axis. Thus, when analyzing a histogram, it is possible to get an idea about various image information, such as contrast, brightness and the distribution of its intensity, for example. Figure 6 shows the example of a histogram created from an image:

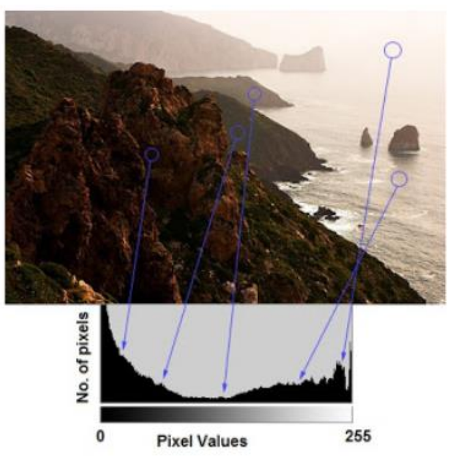


Figure 6 - Creating histograms from images (transcript from [4]*)*

In case of images with different luminosities, it may be useful to apply the technique of colors normalization, since different luminosities can show or hide interesting details. For this, it is necessary to equalize the histogram values of the images, which usually means an improvement in the image contrast. This technique is very used for cases where it is interesting to make all the images with the same lighting conditions, as for example, on facial recognition issues [4], where it is important that the brightness does not pose a problem for the recognition system.

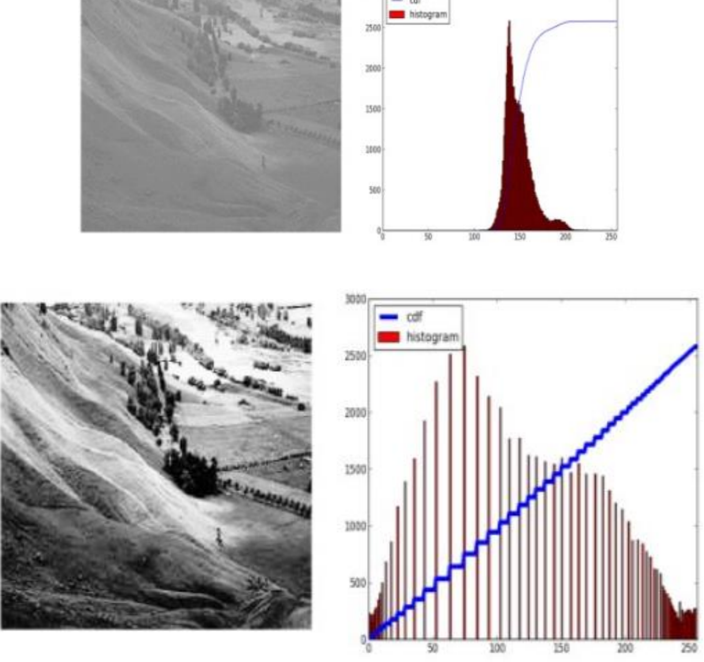


Figure 7 - Equalizing image histograms (transcript from [4]*)*

As we can verify in Figure 7, despite the significant contrast variation between both images, after the equalization, they remain similar.

2.4. Image thresholding

Thresholding is considered the simplest method used in image segmentation. Through a grayscale image, thresholding can be used to create binary images [7]. Basically, if the value of a given pixel is greater than the threshold specified in the function, it is assigned a value (which may be white), otherwise another value will be assigned to that pixel (which may be black).

OpenCV provides several thresholding styles, which are defined in the function available for this purpose that has four different threshold types: the *binary threshold*, the *inverted binary*, the *truncated*, the *tozero* and the *inverted tozero* threshold. Figure 8 shows an example of its implementation.

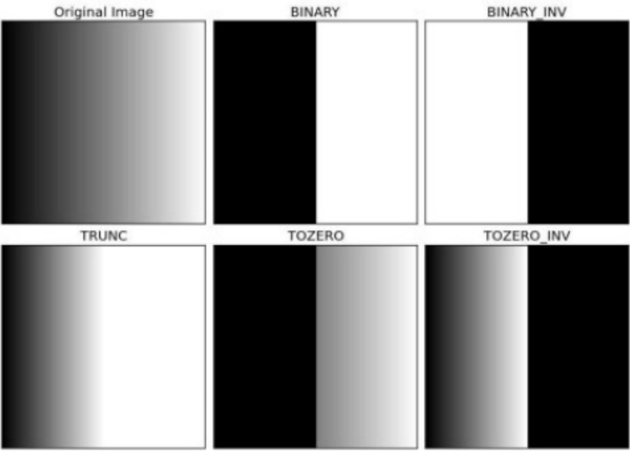


Figure 8 - Applying thresold (transcript from [4]*)*

In addition, *OpenCV* also includes the *Adaptive Threshold* method. This method is most suitable for cases where different lighting conditions exist, so this function calculates the threshold for image's small regions, ensuring better results where the illumination varies throughout the image. Figure 9 shows an example of its implementation.

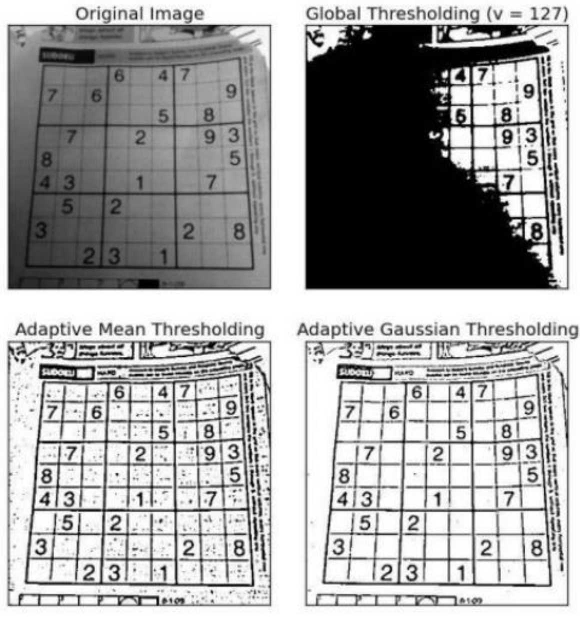


Figure 9 - Appplying Adaptive Threshold
(transcript from [4]*)*

2.5. Find contours

Contours are a very useful technique for form analysis and for object recognition and detection. These consist of a line joining all continuous points along the boundary, having the same color or intensity. According to the *OpenCV* documentation [4], binary images should be used to obtain a better precision. For this, it is necessary to use techniques such as Image thresholding or as Edge Detection, which have already been discussed in earlier sections in this chapter.

Sometimes it is also useful to select an image's area of interest. In this way, OpenCV provides two methods that help filter an image in the area where a contour has been detected, for example, and in these cases bounding rectangles are used. There are two types: the *Straight Bounding Rectangle* and the *Rotated Rectangle*. As the name implies, the first consists of a straight rectangle, which does not consider object rotation, and the second rectangle, is characterized by being drawn according to the minimum area, since it considers object rotation.

Figure 10 shows an example for these techniques' implementation, where the green lines shows the normal bounding rectangle and the blue ones shows the rotated rectangle.

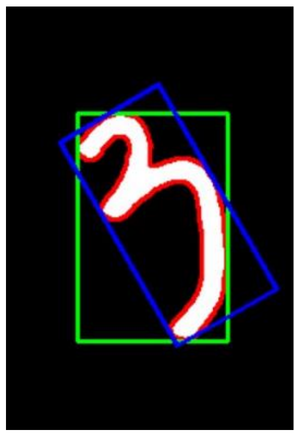


Figure 10 - Select a Bounding Rectangle (edited from [13]*)*

2.6. Segmentation

In image processing, segmentation is a widely used technique, in that we can simplify or change the image's representation to something of greater significance and easier analysis [7].

Najman et al. [14], studied the *Watershed algorithm* defined in a continuous domain. This algorithm is a transformation defined in a grayscale image. Thus, any grayscale image can be viewed as a topographic surface where high intensity denotes peaks and hills while low intensity denotes valleys. It can be started by filling every isolated valley (local minima) with different colored water (labels). As the water rises, depending on the peaks (gradients) nearby, water from different valleys, obviously with different colors will start to merge. To avoid that, is necessary to build barriers in the locations where water merges and the work of filling water and building barriers will be made until all the peaks are under water. Then, the barriers created will give the segmentation result. This is the "philosophy" behind the watershed.

OpenCV has implemented a marker-based watershed algorithm where it can be specified all the valley points to be merged and which don't. Then, we must give different labels to the object we know, such as label the region which we are sure of being the foreground or object with one color (or intensity), label the region which we are sure of being background or non-object with another color and finally the region which we are not sure of anything, label it with 0. That will be our marker. Then, we must apply watershed algorithm and finally our marker will be updated with the labels we gave, and the boundaries of objects will have a value of -1.

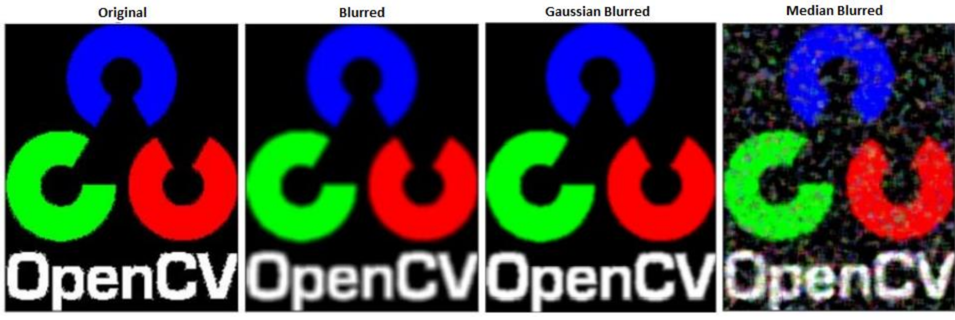
Although this is a good example of an algorithm to perform image segmentation, there are many other algorithms that can be used for the same effect. In addition, this type of algorithms are very useful for image background removal tasks, for example [4].

2.7. Smooth Images

In *OpenCV* there are four functions (most used) responsible for Smoothing Images: *Blur*, *Gaussian Blur*, *Median Blur* and *Bilateral Filter* [4]. Despite its multiple functions, one of the main objectives in this type of technique is the noise reduction on the processed images. Thus, the Blur function is done by convolving the image with a normalized box filter and constitutes the simplest filter of all, where the average of all the pixels in the kernel area is calculated and where the central element is replaced by this same average. The *Gaussian Blur* function, which is probably the most used (but not the fastest) filter [3], is performed by wrapping each image point with a *Gaussian* kernel, and then all points are summed to produce the matrix about to leave.

In turn, the *Median Filter* function (like the previous one) passes through each image element and replaces each pixel with the median over a given window of neighboring pixels. Finally, the *Bilateral Filter* function considers the neighboring pixels with weights assigned to each of them, in addition, these weights have two components, the first one is the same weighting used by the Gaussian filter and the second component considers the difference of intensity between neighboring and evaluated pixels.

Figure 11 shows an example for the result obtained with this technique according to the different chosen functions (respectively).



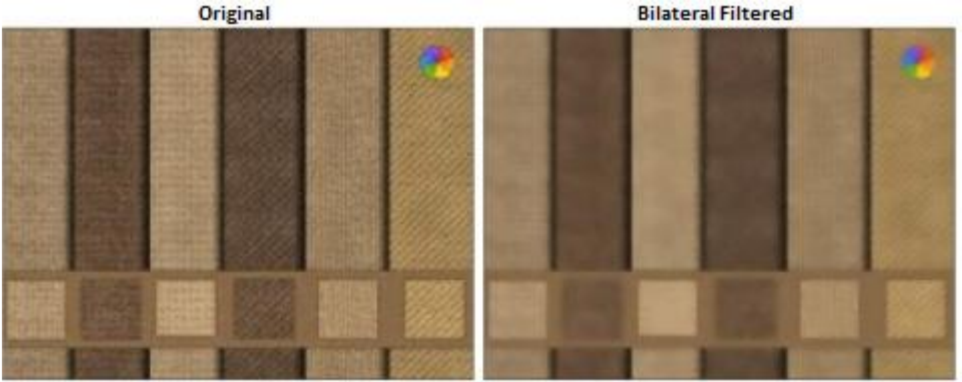


Figure 11 - Smoothing images
(transcript from [4]*)*

2.8. Pattern Matching

Template Matching is a method for searching and finding the location of a template image in a larger one. It simply slides the template image over the input image (as in 2D convolution) and compares the template and patch of input image [4]. Several comparison methods are implemented in *OpenCV*. Despite that, it returns a grayscale image, where each pixel denotes how much the neighborhood of that pixel match with template. If input image is of size $(W\ x\ H)$ and template image is of size $(w\ x\ h)$, the output image will have a size of $(W - w + 1,\ H - h + 1)$. Once you got the result, you can use *cv2.minMaxLoc()* function to find where is the maximum or the minimum value. Take it as the rectangle's top-left corner and take $(w\ x\ h)$ it's width and height. That rectangle will be the region of template. In case of want to detect multiple objects, we must use, for example, *Threshold* instead of *MinMaxLoc* function so that we can match and find multiple occurrences in an image [4]. Figure 12 shows an example of the Template Matching technique working.

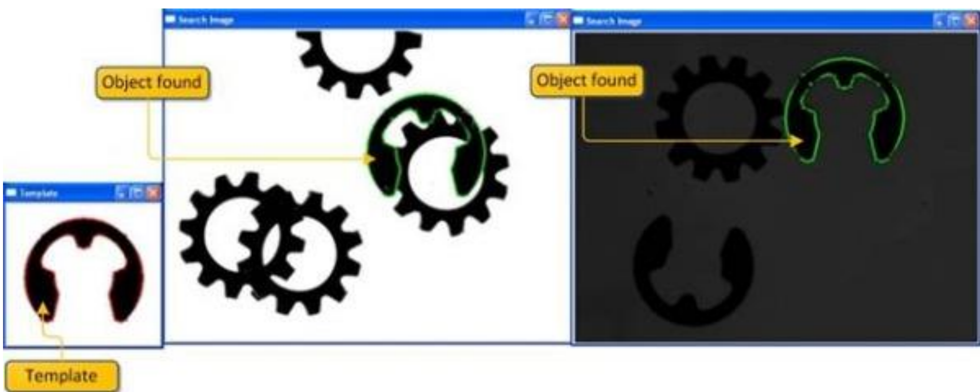


Figure 12 - Matching a template
(transcript from [15]*)*

2.9. Image pyramids

An image pyramid is a collection of images, where all arise from a single original one and that are successively multi-scaled until some desired stopping point is reached, which is a very useful technique that allows us to find objects in images at different scales or when combined with a sliding window we can find objects in images in various locations [16].

This happens because sometimes we are not sure how big the object will be in the image [17], and to look for objects in all of them. To these images' sets, we call Image Pyramids, since they are maintained in the form of a pile, where the largest image is at the bottom and the smallest image lies at the top of the pile, resembling a pyramid. There are two types of pyramids [4]: the *Gaussian* ones and the *Laplacian* ones.

Higher level (low resolution) in a Gaussian Pyramid is formed by removing consecutive rows and columns in lower level (higher resolution) image [4]. Then each pixel in higher level is formed by the contribution from 5 pixels in underlying level with gaussian weights. By doing so, a $(M\ x\ N)$ image becomes $(M/2\ x\ N/2)$ image. So, area reduces to one-fourth of original area. It is called an Octave. The same pattern continues as we go upper in pyramid, so, resolution decreases. Similarly, while expanding, area becomes 4 times bigger in each level [4].

In its turn, *Laplacian Pyramids* are formed from the *Gaussian Pyramids*. There is not any exclusive function for that. *Laplacian* pyramid images are like edge images only. Most of its elements are zeros. They are used in image compression. A level in *Laplacian Pyramid* is formed by the difference between that level in *Gaussian Pyramid* and expanded version of its upper level in *Gaussian Pyramid* [4]. Figure 13 shows an example of a Gaussian pyramid image.

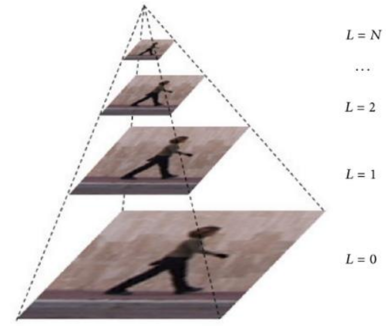


Figure 13 - Gaussian pyramid representation (transcript from [18]*)*

3. Anomaly detection

This chapter aims the state of the art, which explains some fundamental aspects for the project's understanding, such as how wafer's anomaly detection is made, reports some relevant contributions made by the scientific community related to projects already implemented, and provides an overview on the existing methods for automatic learning and the performance metrics to evaluate models and algorithms.

3.1. Wafer's failure detection

Wafers consist of a thin slice of semiconductor material like crystalline silicon, used in the integrated circuits manufacturing [19]. These serve as a substrate for the microelectronic devices which are constructed in and on the wafer. It undergoes many processing steps during its manufacture, such as deposition of various material layers and photolithographic standardization, among others. At the end of all processes, the wafers give rise to individual microcircuits, which are cut into small units and properly packed.

Regarding the limitations that have been found in the context of anomalies detection, the literature states that, as the individual devices are becoming smaller and smaller, the chips have a greater number of layers and consequently go through more processing steps. In addition, as the target area to be inspected is decreasing, defects will also become smaller and smaller, which translates into difficulties to detect defects when processing images in automatic systems due to the optical resolution limit [20] and because the image must be captured using a high degree of magnification, which may have an impact on the images quality that are captured. This is crucial for the wafer's inspection phase since the production line operators perform this process through images observation.

Thus, it is extremely important to perform a good wafer's inspection, since a contamination, for example, can lead to damaging the circuits and severely affect the product's quality. In addition, products with defects (or anomalies) should not be sent to customers because, if this happens, it will lead to high costs and further problems.

Bennett et al. [21], authors of the study on the state and trends of industries, argue that the clean room (the room where the production line is set up), manufacturing practices, and what is supposed to be a contamination free environment are also increasingly becoming a

critical aspect for successful device's manufacturing, whereby one of the high priority factors to consider is the elimination of defect sources.

On the other hand, authors like *Tobin et al.* [22], defend the need for wafer inspections to be performed automatically and, preferably, that the implemented solution has the capacity to be trained over time. However, identifying these anomalies as effectively being a defect, obtaining its location accurately and diagnosing the anomaly cause is something that still requires manual classification. This way the classification may be compromised, since the results depend on production line operator's available time, knowledge, experience, skills and disposition. In addition, manual classification is a rather slow method and does not prevent the occurrence of false results [23].

Despite all its versatility and complexity, the automation of this process is an object of study that is very often required now, as can be seen through the large number of patents and studies in the technological market for the object detection and recognition in images. The following section of this dissertation presents some projects and studies made in this subject.

3.2. Anomaly Detection - State of art

3.2.1. Medicine

There are several studies in the field of pattern recognition and image processing that cover the most diverse areas. In medicine, for example, studies are performed on the identification of oncocyte cells in tumors, since they have hidden patterns and tiny details that may not be so easily evidenced by the manual analysis previously made by doctors [24]. Thus, the studied dissertation's project consisted on a tool, called OncoFinder, that could open, process and classify thyroid tumor images with the least possible probability of error [24] and thus, help the pathologists decision-making regarding the diagnosis, increasing the probability of indicate the most appropriate treatment to the patient.

This study was focused on the cell nucleus analysis, to predict if a cell is cancerous or healthy due to its characteristics. The author of this study reveals that its main difficulties in this project's implementation were in the cellular nuclei image segmentation. In addition, the image's size also proved to be an obstacle [24] since they were too large and, due to the used machine's limitations, made the images difficult to process.

This tool's implementation led the experts to use it to create data regarding the image pixels, which allowed generating data sets readable by different Machine Learning algorithms, so that these images can be used to perform automatic classification, and the tool can automatically detect whether the tumor is cancerous or not.

The conclusions drawn from this research are that, although the cell nucleation segmentation process is not perfect, using different data classifiers, they have shown that some of these can construct reliable models, with an accuracy above 90%, showing that is possible to have a good prediction regarding the detection of tumor cases in the thyroid. However, the author reveals that, although the image segmentation process is somewhat limited, the studies have shown success in finding the kind of characteristics that need to be detected and in constructing a learning model that can perform the automatic classification [24].

For future work, some points are highlighted in this study, like the necessity to improve the developed tool, so that it could carry out a segmentation of high quality images, increasing its reliability giving the tool the ability to handle the processing of high quality microscopic images, segment cell nuclei with high precision, generate training data, test new tumor cases through the generation of data sets for new tests with the trained models with the available data, and the efficient results' display, showing the type of tumor and facilitating access to the process statistics [24].

Thus, they could reduce the need for the experts' interventions, making the work more automated, given the ease of handling the tool. Another factor pointed out is that could be included some additional feedback from the experts because their experience is important to select training data from the classifier's learning, in the initial stages of this tool's utilization [24] by emphasizing the importance of searching for more characteristic image attributes to add to the already identified cell nucleus contour, allowing the classification accuracy improvement. This would generate a greater amount of data, so a database connection would be required to safely store the data being recorded for the classifiers, for example, so that it can be easily queried to extract treated data sets and balanced for automatic classification.

3.2.2. Information Systems Security

In the context of pattern detection, data cannot be obtained only through images. In Information Systems Security, for example, there are studies in the field of pattern recognition, to detect network intrusions, as automatic analysis performed on network traffic using

techniques of pattern detection in the collected data and the techniques of *Machine Learning* so that these can be categorized.

García-Teodoro et al. [25], authors of the study "Anomaly-based network intrusion detection: Techniques, systems and challenges", use several techniques of knowledge acquisition to use data for training, making the system learn with the data. Bayesian networks (establishing probabilities among the variables of interest), Markov models, Neural Networks, Fuzzy logic techniques, genetic algorithms and clusters were tested.

The challenges founded by these authors were that development of high-quality knowledge is a difficult and slow process, and that anomaly detection systems must be prepared for the appearance of new anomalies, to have the capacity to identify new attacks. Moreover, the Internet traffic surveillance is used to prevent computer attacks, helping to find suspects for such events based on the web information, which are often related to many of the threats to people or infrastructures in the real world.

In this type of events Bouma et al. [26] consider that the pattern's automatic detection is a crucial mechanism since the amount of data on the Internet increases rapidly and the flow of traffic from several sources is monitored at the same time. The main objective of this study was to create a system that had the ability to monitor trends, create profiles based on correlation analysis of a few variables (sentiment analysis, reposts, frequency of posts, among others) and to recognize abnormal changes in users activity, behavior or emotion, in this case through Twitter data, analyzing the content of new or reused images, extracting keywords that are associated with it and then performing the sentiment analysis of the made posts, which will allow the system to alert the analyst of some abnormal behavior in an early stage of the problem, so this study, is concentrated on two basic aspects: the detection of abnormal behaviors on the Internet and the combination of image content and overlapping text results to generate additional knowledge.

One of the difficulties pointed out in this study was the concept of anomaly, since the data that are being analyzed come from complex events, with many concepts and tendencies of interaction, besides that some anomalies become normal after some time, and the borderline between normal and abnormal is sometimes very narrow, since the anomaly is only considered, depending on the context in which it occurs [26].

In conclusion, they characterize an anomaly as an abrupt change, consisting mainly of an explosion of positive sentiment. In addition, they followed a multivariate analysis approach, creating a system that calculates correlation matrices for many variables over time,

implementing calculations such as the mean value, standard deviations between correlations, and the z-score, which is later aggregated and for a given time interval, is described by a graph in which the peaks that it may possess mean the presence of anomalies, displaying also a resource polygon that connects all variables chosen as points with an inner polygon representing their mean values and an outer polygon that represents any deviations of certain variables, which helps analysts to analyze data, making them realize which variables are responsible for a given peak in the anomaly curve [26].

3.2.3. Biometrics

Automatic pattern detection along with image processing also comes within the scope of biometrics and user identity verification systems. Arun Ross and Anil Jain [27], authors of "*Information fusion in biometrics*", address the problem of merging information into biometric verification systems by combining information at the level of three biometric modes: face, fingerprint and hand geometry.

According to them, user's biometric data are acquired using a biometric reader, which stores the records made in a database [27]. These data are properly labeled in a model according to the user's identity (name, identification number, among other information). When the user tries to access he must interact with the sensor, so that the sensor can extract the same features and process them. This way, the extracted values are compared to those of the model previously stored and generate a corresponding score, so that the decision regarding accepting or rejecting the user's access is given according to the score obtained.

The obtained data from each sensor is used to calculate a Feature Vector [27]. A Feature Vector consists of an n-dimensional vector of numerical features that represent some object, so in the case of images, these vectors represent their pixels. The score obtained results in the calculation between the calculated vector's proximity at the time the supposed user is in contact with the sensor and the vector that was generated when the model was created with the user's identity [27]. Techniques, such as logistic regression are used to combine the results, if they come from different sensors, since several biometric indicators are being evaluated at the same time and, as such, this combination of results is made so that the truth can be affirmed of the identity that is being presented.

The techniques of scoring and logistic regression used in the study are intended to maximize the genuine acceptance rate for a given rate of false acceptance [28]. The automatic

decision-making is performed thanks to the classification that is made to the analyzed Feature Vectors, which can be classified into two classes: accept or reject [27]. The authors of this study chose classifiers such as K-nearest neighbor and Neural Networks to perform the fingerprint classification, although several strategies for combining multiple classifiers were suggested [29] with the aim of reducing a certain set of classes, so the techniques suggested by J. Hull et al. [29] are relevant for problems that address a large number of classes, which, according to Arun Ross and Anil Jain [27], did not apply to the study in question.

Authors such as Lin Hong and Anil Jain [30] argue that the fusion of multiple biometric indicators, also called multibiometric fusion, aims to improve the speed and accuracy of a biometric system that integrates the different obtained scores from the different biometric sources used. In order to compare these scores, the literature used in this study, based on Ben-Yacoub et al. [31], U. Dieckmann et al. [32] and Kittler et al. [33] studies, suggests the implementation of classifiers such as Decision Trees, Support Vector Machines (SVM), K-Nearest Neighbor (KNN), and Bayesian methods.

Summarizing, the study performed indicates that the rule of sum the scores shows better results than the methods of the Decision Trees and the Linear Classifier. In addition, one of the advantages evidenced by the authors was that the benefits of multi-biometric merge become more evident with the increase in the user's number in the database. Another point mentioned by the authors was the fact that it is convenient to assign different weights to the different individual modalities for the biometric indicators [27].

3.2.4. Processing huge amounts of data

As previously mentioned, image processing is an important task in many areas of research, such as medical imaging, remote sensing, Internet traffic analysis, among others. However, due to the high amounts of images that are currently processed and the size they occupy, it is not possible to process them on a single computer, which transposes the need to implement distributed computing systems to process large amounts of images without the compromising the system's efficiency. According to *Epanchintsev et al.* [34] distributed computing is a very complicated subject, since it requires deep technical knowledge and sometimes cannot be used by researches that develop image processing algorithms. Hence the need to look for tools that can somehow hide the complicated details of distributed computing, so solution developers can focus more on image processing tasks as this is their primary purpose.

Epanchintsev et al. [34], describe the extension to the *MapReduce Image Processing* (*MIPr*) framework that provides the ability to use *OpenCV Hadoop Cluster* for distributed image processing. *OpenCV* is an open-source library, used in the scope of computer vision, that is, the tasks automation that the human visual system can perform [35].

In this study, an approach based on the *MIPr* structure was explored, which was previously developed for *Hadoop* image processing, and an extension was made to the *MIPr framework*, so that it could include image representation based on *OpenCV* in the *Hadoop Distributed File System* (*HDFS*), as well as tools for reading these *HDFS* images, saving the processed images back to the *HDFS* format, and even the drivers to support the employment of the *MapReduce Jobs* with *OpenCV*. This way, the implemented extension allows the development of *Java* programs for image processing, using *OpenCV* [34].

The authors concluded that, for large clusters and image data sets, the scalability of the modified *MIPr* application is almost linear. However, when the test was performed on a single-node cluster and for a small data set, the authors found that there was poor performance in the application due to the overhead of *Hadoop*. In addition, these include, as a future work theme, the use of *MIPr* for *Apache Spark* [36], so that image processing can be done in memory, and that the ability to use *Python* and *C++* in program development for image processing using *MapReduce*.

There are also several works carried out in the anomalies detection, namely with the use of *Hadoop* and the technique of *MapReduce*, to detect anomalous situations in data coming from sensors. The analyzed article presents a *Cloud Computing* observation model for detecting any abnormal event that may be a threat. For this, the authors defend the use of *Hadoop* and *MapReduce*, since *Hadoop* is an open-source framework used to compute large amounts of data in the *Cloud* and uses the *Machine Learning* technique based on *MapReduce* to detect anomalies occurring during observation of the sensors data, by classifying them, which ensures the stabilization of virtual sensor networks [37].

Regarding the *Machine Learning* algorithms tested, the authors used *Random Forest*, *OneR* (*One Rule*), which creates a level one *Decision Tree*, *Bagging* and the *J48*. In addition, the implementation proposed was used, using only 75% of the total data to test and train the data and thus obtain the model. This model’s outputs were divided into subparts to simplify their analysis and evaluation [37]. So, the first subpart has examples of instances that were correctly classified and instances that were not correctly classified, which were divided by their

percentage value, the absolute and relative error, and by the root values of mean absolute error and the root of mean square error.

Summarizing, *Epanchintsev et al.* [34], prevailed with the importance of collecting the sensor's observations for analysis in a continuous and properly integrated way, so that a more precise forecast model can be constructed. They also mentioned that better results were obtained through the using of the Random Forest algorithm, compared to the other techniques that were used.

3.2.5. Semiconductors industry

As can be seen throughout the analysis of the various studies reported above, the area of pattern recognition and image processing is a sector that has gained increasing strength in recent decades. However, an industry where this type of technology is quite supported is the semiconductor industry. There are systems available in the market, although they are very expensive, that are responsible for the location and detection of potential wafer's anomalies. In addition, these systems may not meet the specifications of all companies in the same way, due to the high specificity of their products and their customers, as well as due to the technology used in the manufacture of their products, among other factors.

Despite this, several authors like Tobin et al. [22], defend the need for wafer's inspections to be performed automatically and, preferably, that the implemented solution has the capacity to be trained over time by the user.

The study carried out by *Mark Schulze et al.* [38], consisted on the presentation of case studies of a new wafer inspection based on digital holography. Digital holography records the wave front amplitude and phase of a given target object directly into a single image acquired through a *Charge-Coupled Device* (CCD), which was specially developed for ultraviolet imaging [39]. These authors also argue that the wafer's defects detection using this technique is performed by directly comparing two complex wavelengths of the corresponding fields of the adjacent field of view on the wafer. So, to isolate in which field of view a defect has been detected, two comparisons are made for each field of view and the defect is assigned to those fields only if it appears in both difference images, which are the images that can be computed as phase or amplitude differences, or as an image of the composite difference [38].

The tests performed in this study comprised image surface variations visualization through the phase information, visualization and detection of sub-recorded high-ratio contacts,

and visualization and defects detection derived from a partial extension respecting their height. These tests were performed either using the *nLine Fathom* tool (which is an optical tool used to extract images from wafers) or using the *Scanning Electron Microscope*.

Concluding, the authors note that the *nLine Fathom* tool has some addition capabilities regarding defect detection that are not provided by existing inspection tool technologies. To compare two images of different dies, the authors draw attention to requirements such as the pixels alignment between both images (spatial register), as well as to match their total intensity and phase shift, an operation called normalization [38].

3.3. Machine Learning

According to *William L. Hosch* [40], *Machine Learning* deals with the study of pattern recognition and the theory of computational learning in the field of artificial intelligence. Thus, the main goal of *Machine Learning* is to study algorithms, so that computers learn certain tasks, such as making accurate predictions or being able to behave intelligently without human intervention or assistance, learning to do better in the future based on past samples and experiments. So, *Machine Learning* is based on some kind of observations or data, such as examples, direct experience, or instruction [41], and explores the study and construction of algorithms that can learn from their own mistakes and elaborate predictions about the data they deal with [42].

Rob Schapire [41] argues that, to implement an intelligent system, we must make use of learning and for this, the system must be able to perform the functions that are associated with intelligence such as language or vision. Thus, the author classifies these tasks as "difficult to solve" because even though automatic learning is a subarea of artificial intelligence, it is related to many other areas, namely, statistics, computer science, mathematics, physics, among many others, making its implementation quite complex [41].

In addition, the author considers that despite the complexity of its implementation, this is a topic of great interest to the scientific community due to the large number of application domains it covers, such as optical character recognition, object detection, spam filtering, topic spotting, spoken language understanding, medical diagnosis, customer segmentation, fraud detection or weather prediction, among others [41].

According to *Stuart Russell* and *Peter Norvig* [43], in *Machine Learning*, tasks are typically classified into three categories depending on the performed learning type. The learning

can be classified as supervised if the system's learning objective is to learn a general rule that maps the inputs to the outputs. Thus, if no label type is given to the learning algorithm it can autonomously find the structure of the inputs provided, the learning can be classified as unsupervised. In turn, if feedback is provided to the program regarding awards and punishments insofar as the problem space is known, then this is the type of reinforcement learning. The authors also note that, in addition to these three categories, we may also consider semi-supervised learning, where a set of training data is provided with some or even several of the desired objectives missing [43].

According to *Michael I. Jordam* [44], ideas about methodological principles and the theoretical tools of *Machine Learning* have a long prehistory in statistics, so the author suggests that the term *Data Science* is like a substitute for calling the field as a whole. In turn, *Leo Breiman* [45] distinguished two paradigms of statistical modelling: the data model and the algorithmic model, where the last refers to *Machine Learning* algorithms. In addition, some statisticians have adopted methods of *Machine Learning*, prompting them to call this field *Statistical Learning* [46].

Therefore, according to some of the authors mentioned, we can consider that *Machine Learning* and Statistics are closely related fields.

3.4. Methods for Automatic Learning

This section will present the Automatic Learning and training methods resulting from a survey in the available literature about this subject.

3.4.1. Learning methods

There are different methods of using Machine Learning. In the supervised learning method, the system is programmed or trained from a set of predefined or labeled data. In the unsupervised learning, the program can automatically find patterns from a set of data, and in the semi-supervised learning, it is made up of a small amount of labeled data, which usually has a higher cost and a larger volume of unlabeled data.

3.4.2. Decision Trees

Decision Trees focus on a complex problem, dividing it into simpler problems, recursively applying this strategy to successive sub-problems, and in the end, the solutions of the sub-problems can be combined to generate the solution of the initial problem [47].

Instances are classified starting at the root to a given leaf node, each node of the tree specifies an instance attribute, and each descendant branch being one of the possible values for that attribute. Each instance begins to be classified by the tree's root, testing the attribute defined by the node and descending later along the branch corresponding to the value concerning that attribute, repeating this whole process for the subtree, whose root is a new node.

Therefore, decision trees consist of a division of combinations, that is, each path from the root to a given leaf reflects the combination of attributes, whereas the tree itself consists of dividing the same combinations.

Figure 14 shows an example of a decision tree's representation. In it, the concept of playing tennis is represented. It should be noted that, in this case, the classification associated with each of the leaves is yes or no, and the paths that run through the tree nodes are arranged along this from the root to the appropriate leaf node.

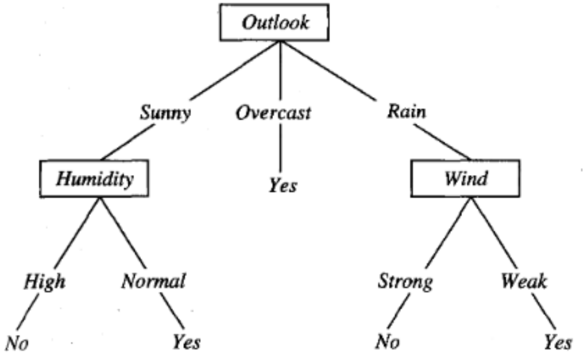


Figure 14 - Representation of a decision tree (transcript from [48]*)*

In this case, the above decision tree expresses that, if the weather is cloudy or if the weather is sunny with normal humidity values or it rains but with weak wind, then you can play tennis. Otherwise, if the weather is sunny and the humidity is high or if the weather is rainy with a strong wind, then you can't play tennis [48].

3.4.3. Classification rules

The learning made by classification rules or association is an approach that consists of the discovery of interesting relationships among existing variables in databases of considerable size.

Tree-learning, for example, is one of the approaches that follows this method [49]. Its representation uses if-then rules, which will cause the algorithms to "learn" certain sets of rules. These algorithms, for the most part, are apt to learn first order rules that contain variables, which is something significant, since this type of rules is much more expressive than the propositional ones [50].

3.4.4. Inductive Logic Programming

As well as the learning methods mentioned above, with inductive logic programming is possible to also produce classification rules. This method uses logical programming, giving rise to models, which correspond to a set of rules. Models are often generated from background knowledge, that is, from data generated from prior knowledge. Inductive logic programming systems can also create models from input data, obtained from a training process for a given set of examples [50].

Several systems use this method and usually follow an approach regarding the discovery of patterns, and the space of searching for these same patterns can be very wide or even infinite. Thus, *Induced Logic Programming* systems often use search strategies such as greedy, randomized search or branch-and-bound search. Thus, each generated pattern is evaluated to determine its quality, and all patterns that are not suitable to the model will be discarded. Thus, once a pattern is found that meets all requirements, the search is terminated [50].

A certain pattern is evaluated together with the information about the previous knowledge acquired, so that we can test the training examples perception in relation to the pattern found.

However, for a limited number of training examples, this activity can be an extremely time-consuming process, demonstrating this as one of the disadvantages inherent to this method, as well as the fact that these are computationally expensive systems, since individual

rule evaluation may take several hours to complete until a model is returned, which means that low efficiency levels are one of the biggest obstacles that systems of this nature face [50].

Therefore, expressiveness, readability and the use of prior knowledge constitute the main advantages of Inductive Logic Programming. The first point is characterized by the fact that first order logic allows the representation of a series of more complex concepts compared to traditional languages. The author in [49] states that a first order representation is probably easier to read than one of zero order. In addition, in the third point described above, the author states that this factor could be a good discovery system and that, in some cases, the prior knowledge could be extended throughout the time of discovery itself [49].

3.4.5. Support Vector Machines

Support Vector Machines (SVM) are a set of supervised methods that are used for classification and regression. This approach is often used for a set of training examples that fall into two categories, since the objective of the SVM is to find the best classification function that allows the distinction of members of both classes. Thus, for a linearly separated set of data, there exists a linear classification function corresponding to a hyperplane $\boldsymbol{f}(\boldsymbol{x})$ that crosses the two classes, dividing them. When this function is determined, the new instance $\boldsymbol{x_n}$ is classified according to the function signal $\boldsymbol{f}(\boldsymbol{x_n})$ and $\boldsymbol{x_n}$ belongs to the positive class if $\boldsymbol{f}(\boldsymbol{x_n}) > \boldsymbol{0}$ [51].

Given the existence of many hyperplanes, an SVM ensures that the best function is found after maximizing the margin between both classes, corresponding to the shortest distance between a set of points closer to each other and a certain point on the hyperplane (in geometric terms). Although there is an infinite number of hyperplanes, only one is constituted as the solution for the SVM in question. In addition, despite this approach reveals high accuracy rates, SVMs are also very slow when processing large datasets [50].

Figure 15 shows the representation of an SVM, denoting the presence of some existing hyperplanes, a margin, as well as a linear separation of two distinct classes. The example in question portrays the concept of buying a computer, in which the prediction of an electronic store customer when acquiring a computer is calculated [52].

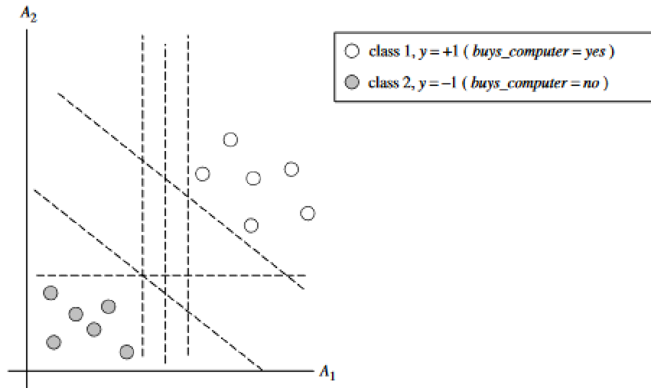


Figure 15 - Graphical representation of a Support Vector Machine for two classes (transcript from [52])

3.4.6. Bayesian methods

Bayesian methods are characterized by associating a probability to each forecast, thus representing the confidence level of the classifier in the final classification [53], so another of the difficulties presented by this type of method is the need for a significant investment in computational terms to determine the optimal *Bayes* hypothesis for the general case [48].

The following equation is implicit in all current artificial intelligence systems for probabilistic inference and is known as the *Bayes* rule.

$$P(Z|Y) = \frac{P(Y|Z)P(Z)}{P(Y)}$$

Learning methods based on *Bayesian* networks are relevant for the study of automatic learning, due to these algorithms, which calculate probabilities for certain hypotheses, are one of the most used approaches for the resolution of several problems' types. In addition, *Bayesian* methods provide a useful perspective in understanding several learning algorithms that do not explicitly manipulate probabilities. Even so because the *Bayesian* classifiers are statistical classifiers [52].

Michie et al. [54], authors of a study comparing the *naive Bayes* classifier to a series of other learning algorithms, namely algorithms related to decision trees and neural networks, reveal that the Bayes naive classifier is extremely competitive with several of these algorithms in many situations. In some cases, the Bayes naive classifier even surpasses these other algorithms [48]. Therefore, this classifier is characterized as being one of the most effective known classifiers for certain learning tasks.

Pompe et al. [55] states that the simplicity and robustness of the naive Bayes classifier make it a good candidate for the combination of rules learned. In addition, according to *Wu et al.* [51], although this classifier is not the best for certain situations, most of it proves to be extremely robust, revealing even high levels of performance.

Despite this, one of the difficulties inherent to the application of Bayesian methods is that it usually requires the knowledge of a series of probabilities. However, if these probabilities are not known, estimates based on prior knowledge are often made, that is, they are calculated based on previously available data or assumptions in the form of underlying distributions.

Therefore, a *Bayes* classifier consists of a rule that predicts the most likely class for a given example, based on the considered data set distribution, if this is known [56]. In addition, the topology of a *Bayesian* network is composed of a *Directed Acyclic Graph* (*DAG*), where each node represents a random variable, so a set of random variables gives rise to the network nodes, and these variables can be discrete or continuous.

Bayesian Networks provide a complete description of the domain they represent, since any entry into the distribution of joint probabilities can be computed through the information present in the network. Figure 16 represents an example of a *Bayesian Network* consisting of only two nodes and a link.

Figure 16 - Non-Causal Bayesian Network Example (transcript from [57]*)*

Figure 16 was taken from the study carried out by students at the University of Delaware and later reported by *Snee* in 1974 [58], which consisted of studying the eyes and hair's color of statistical students using a sample of 592 students. In this case, it is important to note that this Bayesian network does not contain any causal hypotheses, that is, there is no knowledge of the causal order among the variables. Therefore, the interpretation of this network should be merely informative or statistical.

In turn, if we consider a problem that includes a *Dynamic Bayesian Network* (*DBN*), for example, there is a need to monitor variables whose values change over time, we can capture the process representing multiple copies of stage variables, one for each stage of time.

According to *Weber and Joufe* [59], *Dynamic Bayesian Networks* are a generalization of the *Hidden Markov* (*HMM*) and *Kalman Filters* (*KF*) models where, each of them can be represented with a *DBN*. In the case of *HMM*, if we represent it in the form of a *DBN*, we make it much more compact and, in turn, better understandable. The nodes in the *HMM* represent the system states, while the nodes in the DBN represent the system dimensions.

Figure 17 shows the *DBN* representation in an example of a valve system in 9 nodes and 11 arcs against 26-node and 36 arcs if the example was developed in the *HMM* representation [59].

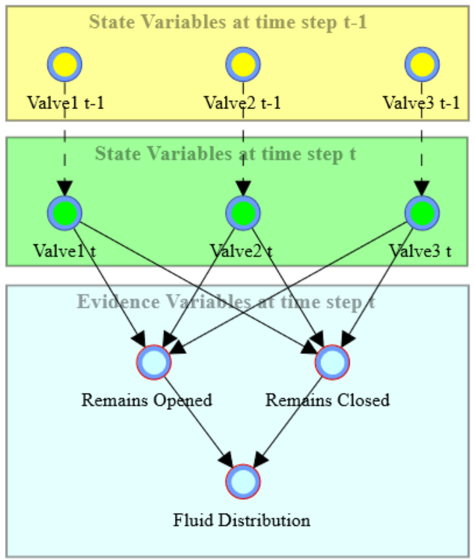


Figure 17 - DBN model
(adapted from [59]*)*

3.4.7. Artificial Neural Networks

Artificial Neural Networks (*ANNs*) provide a general and practical method for learning discrete-value functions of real-valued vector values from examples. This learning method is robust to errors in training data and has been successfully applied to problems such as interpreting visual scenes, speech recognition, and robot learning control strategies [48], as several authors have reiterated that the Backpropagation algorithm (calculate the gradient of the loss function in relation to the weights in an *ANN*) proved to be surprisingly successful in many practical problems, such as learning to recognize handwritten characters [60], as well as learning to recognize spoken words [61] and learning to recognize faces [62].

The study of neural networks is essentially inspired by the observation that biological learning systems are constructed of very complex webs of interconnected neurons. Thus, artificial neural networks are constructed making an analogy to the biological learning system,

which are generated from a densely interconnected set of simple units, where each unit has a series of real-valued inputs (possibly the outputs of other units) and produces a single output of real value (which can become the input to many other units).

Tom Mitchell [48] begins by comparing some facts of neurobiology with those of *Artificial Neural Networks* (*ANN*). One of these examples resides in the processing time of a recognition response, in that it is estimated that the faster switching times of neurons are known in the order of 10^{-3} seconds, which is shown to be much slower in relation to the speed of some computers, which lies in the order of 10^{-10} seconds.

However, humans can make quite complex decisions in a surprisingly fast way, for example, it is estimated that a human need approximately 10^{-1} seconds to visually recognize his mother. Thus, this observation has led many researchers to speculate that the information processing skills of biological neuronal systems must be followed by highly parallel processes that operate in representations that are distributed across many neurons.

Thus, one of the motivations in the construction of *Artificial Neural Networks* (*ANN*) systems is to capture this type of highly parallel computing based on distributed representations, so that most of the software that uses *Artificial Neural Networks* is executed in sequential machines, emulating distributed versions, although some faster versions of the algorithms have also been implemented in highly parallel machines and specialized hardware designed specifically for this type of applications [48].

Therefore, *Artificial Neural Networks* consist of a method for solving problems by simulating the human brain, simulating even its learning behavior, making mistakes, and making discoveries, acquiring knowledge through experience. *ANNs* have nodes or processing units, each unit having connections to other units, where they receive and send signals, thus simulating the neurons activity that receive and relay information.

Artificial Neural Networks may have one or more layers, for example, in an artificial neural network used for an automatic classifier, we can implement a three-layer system, the first being the input layer, where the units receive the patterns. Then we would have the intermediate layer, where the patterns processing and its characteristics extraction would be made, and finally, the output layer, where the algorithm concludes and presents the result. Figure 18 shows a diagram representing an *Artificial Neural Network* of this type.

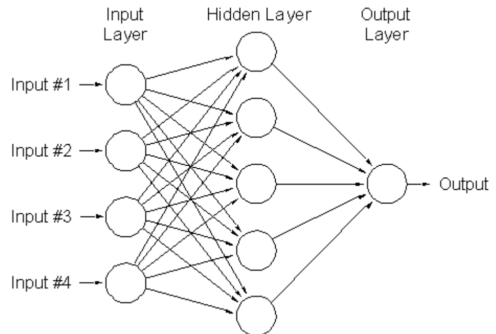


Figure 18 - An example of a ANN with a single hidden layer (transcript from [63]*)*

According to *Daniel Graupe* [64], the higher the layers number, the better is ANN's learning ability, because many layers can use many filters simultaneously.

In the real world, this type of method is widely used for tasks: function mapping, time series prediction, classifications, and pattern recognition. Their learning consists of modifying the connections weights between the neurons, where the initial weights, resembling the synapses of the human brain, are modified iteratively, by an algorithm that follows some type of learning paradigm.

According to *Hertz et al.* [65], learning paradigms can be referred to: supervised learning, where a training set containing the corresponding inputs and the desired outputs for that sample is presented to the algorithm; the reinforcement learning paradigm, where, for each input presented, an indication is given of the corresponding outputs produced by the network or the unsupervised learning paradigm, where the network updates the initial weights, without any indication of the outputs that the network should produce, since only the inputs were provided.

3.4.8. Cascading Classifiers

Cascading Classifiers are based on the concatenation of various classifiers, using all information collected from the output of a given classifier as additional information for the next classifier in the cascade [66]. This learning method is characterized by having **N** stages, where its main objective is to optimize the object recognition, causing many regions not containing the desired object to be discarded in their initial stages to ensure greater precision in later stages to avoid a false positive in the analyzed region. Thus, if an area in the image passes through the last stage of the cascade, then it contains the desired object. Stages within a cascade are created

by combining classification functions previously assembled using the AdaBoost learning algorithm. Since this is a restricted optimization problem, each stage of the classifier cannot have a detection rate (sensitivity) below the desired rate.

AdaBoost is a machine learning meta-algorithm formulated by Yoav Freund and Robert Schapire [67], which can be used in conjunction with many other types of learning algorithms to improve performance. The output of the other learning algorithms, called as “weak learners” is combined into a weighted sum that represents the final output of the boosted classifier.

The training process of a classifier of this type requires sets of positive and negative samples, where the negative samples correspond to arbitrary images that are not objects. Once the classifier is trained, it can be applied to a region of an image and detect the object in question. To search the object in the entire image must be defined a Sliding Window, which is moved to check all the image places for the classifier. This process is most commonly used in image processing for object detection and tracking.

The first Cascade Classifier was developed by Viola and Jones [68], used as a facial detector, where one of its requirements was to be fast to be implemented on low-power CPUs, such as cameras and phones.

Cascade Classifiers are available in the OpenCV library [69], with pre-trained cascades for the front and upper body. Despite the existence of pre-trained cascades, it is possible to train a new cascade in the OpenCV with the haar_training or train_cascades methods for the quick detection of objects of more specific targets, including non-human objects with Haar-like features.

3.5. Performance metrics and model evaluation

The performance metrics quantify the performance of a given classifier, ensuring the results reliability [70]. The following subsections present the different metrics found in the literature that are used to measure their quality.

3.5.1. Confusion matrix

Confusion matrices, also called of contingency, consist in a table of double entry constituted by the real values and by the values predicted by the automatic classifier, thus the

values of the cells match to the number of instances corresponding to the crossing of both entries. Therefore, this matrix shows the results obtained from a certain classifier [71]. Figure 19 shows a schema that represents a confusion matrix definition.

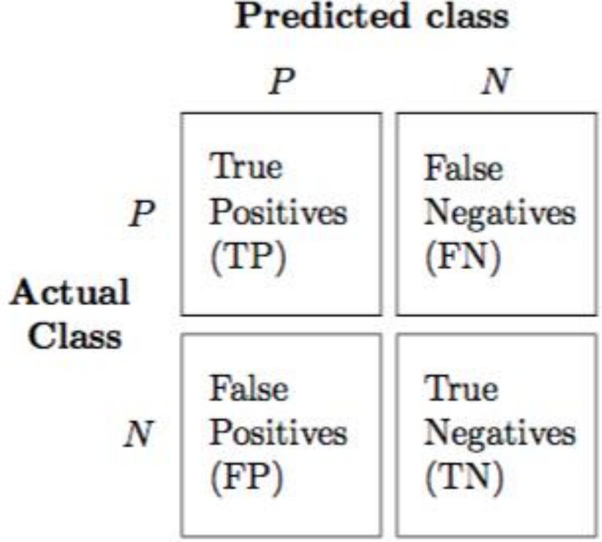


Figure 19 - Confusion matrix definition (transcript from [72]*)*

Where,

P represents positive images (that shows the condition to be predicted);

N represents the negative images (that don't show the condition to be predicted);

TP corresponds to the number of correct previsions for a positive instance;

FN corresponds to the number of incorrect previsions for a positive instance;

FP corresponds to the number of incorrect previsions for a negative instance;

TN corresponds to the number of correct previsions for a negative instance.

3.5.2. Accuracy (ACC)

Accuracy corresponds to the percentage of instances that were predicted correctly by the classifier, so that it corresponds to the rate of positive and negative examples correctly classified. Therefore, accuracy is calculated by the following formula:

$$Accuracy(ACC) = \frac{(TP + TN)}{TP + TN + FP + FN}$$

In turn, the number of instances that were incorrectly predicted by the classifier is obtained by the following formula:

$$ICI = \frac{(FP + FN)}{TP + TN + FP + FN}$$

3.5.3. Precision (PPV)

The main objective of precision (also known as Positive Predicted Value) is to measure the previsions effectiveness made by classifier. Troy Raeder [70], defines precision as:

$$Precision = PPV = \frac{TP}{(TP + FP)}$$

3.5.4. Specificity (TNR)

Specificity, known too as True Negative Rate (TNR) quantifies the proportion of negative samples that were correctly classified and is obtained by the following formula:

$$TNR = \frac{TN}{(TN + FP)}$$

3.5.5. False Positive Rate (FPR)

False positive rate, as the proper name says, quantifies the proportion of negative samples that were incorrectly classified as positives. This calculation is obtained by the following formula:

$$FPR = \frac{FP}{(FP + TN)} = 1 - specificity$$

3.5.6. False negative rate (FNR)

False Negative Rate (FNR) quantifies the proportion of positive samples that were incorrectly classified as negatives. This calculation is obtained by the following formula:

$$FNR = \frac{FN}{(TP + FN)} = 1 - sensibility$$

3.5.7. Recall (Sensitivity, True Positive Rate)

At classifiers evaluation, Recall, Sensitivity and True Positive Rate (TPR) are defined by the following formula:

$$Recall = Sensitivity = TPR = \frac{TP}{(TP + FN)}$$

3.5.8. Cohen's Kappa coefficient

The Kappa statistic (or Kappa coefficient) addresses problems where there are usually situations where several opinions for the same classification, measuring the agreement between two evaluators who classify N items into mutually exclusive categories. The value of this coefficient shall be calculated by the following formula:

$$k = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)}$$

Where,

$\mathbf{Pr}(\boldsymbol{a})$ corresponds to the observed relative agreement between the evaluators

$\mathbf{Pr}(\boldsymbol{e})$ corresponds to the hypothetical probability of agreement occurring by simple chance, using the observed data to calculate the probabilities of each observation.

The kappa value of 0 indicates a level of agreement equivalent to a simple chance, whereas the kappa value equal to 1 indicates a perfect agreement level [73].

3.5.9. F-measure

Is possible to define a measure that assign arbitrary weight to precision and to recall. This measure is known as F-measure because it gives equal importance to both metrics [70]. Therefore, this calculation is obtained by the following formula:

$$F = \frac{2\ x\ precision\ x\ recall}{precision + recall}$$

3.5.10. Receiver Operating Characteristic curve (ROC curve)

The Receiver Operating Characteristic curve (ROC) consists on a graphical representation of the diagnostic capability of a binary classifier system as its discrimination threshold is varied. Therefore, the ROC curve represents the true positive rate (TPR) (also called sensitivity [74]), as a variation of the false positive rate (FPR), also known as the false alarm probability [74] and which can be calculated by (1-specificity).

Figure 20 shows an example of a ROC curve graphical representation.

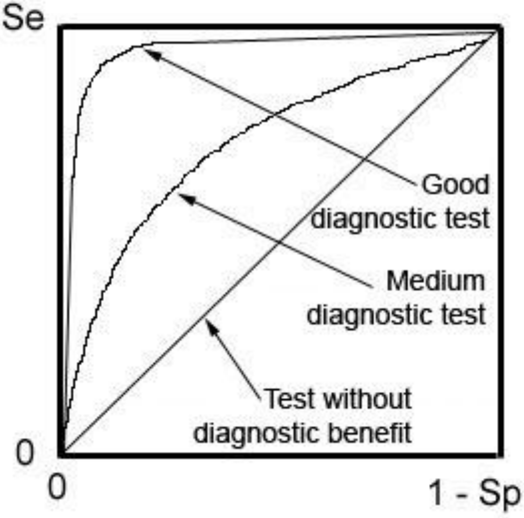


Figure 20 - ROC curve representation example (transcript from [75]*)*

The ROC analysis is done by analyzing its graphical representation but the area under the curve can also be calculated. This ranges from 0 to 1, where 1 represents a perfect classifier, 0.5 shows that the classifier is approximately random (which is considered a bad result), and a ROC area of 0 indicates that the classifier is always wrong. Using balanced data sets, the ROC curve more effectively captures the equilibrium between true positives and true negatives [70].

3.5.11. Precision-Recall curves

The Precision-Recall (PR) curve shows the tradeoff between accuracy and recall for different thresholds. This is a useful measure to evaluate the classifier's outputs quality when the classes are very unbalanced, thus presenting an alternative to the ROC curves for cases involving unbalanced data sets. In information retrieval, accuracy is a result relevance measure, while recall is a measure of how many correct results are returned [76].

There is a large difference between the visual representation of both curves: the goal of the ROC space is in the upper left corner while the Precision-Recall space objective is in the upper right corner.

Thus, a high area under the Precision-Recall curve represents both high recall and high precision where high accuracy is associated with a low false positive rate and high recall refers to a low false negative rate. The classifier will be returning results with high accuracy and mostly returning all positive results when their scores are high for both factors [76].

Figure 21 shows an example of a Precision-Recall curve graphical representation.

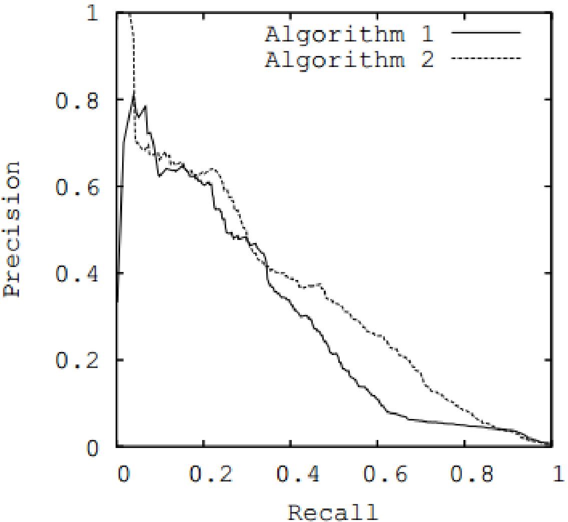


Figure 21 - Precision-Recall curve representation example (transcript from [77]*)*

3.6. Conclusion

According to the literature review carried out for this dissertation (on 3.2 - Anomaly Detection - State of art), it was considered that for this project, the areas to be investigated will be based on the image processing along with the automatic pattern recognition.

In some studied projects, image size proved to be an obstacle, due to the used machines dimensions and limitations. Therefore, the use of a distributed environment using clusters will be considered. The tool chosen for this purpose will be Hadoop since there are several versions and sandboxes for this open-source software. Other factors will also be investigated, as image segmentation, image alignment [38] and the limits of optical resolution [20], insofar as the target area to be inspected tends to be decreasing over time.

In the project to be implemented from this dissertation, the analyses are always carried out under the same context, so data will be divided by classes where each class will correspond to an anomaly type, so that the solution can identify the predominant pattern in each class and thus, be able to categorize the different anomalies types that may occur because, as Bouma H. et al. [26] proved, sometimes the anomalies would only be considered as such, depending on the context in which they are found.

Since the literature indicates that the use of OpenCV together with Python in the scope of Image processing for wafer anomalies detection has not yet been resolved and Epanchintsev et al. [34] enunciate the use of OpenCV Hadoop Cluster for distributed image processing in languages such as C++ and Python for future work perspectives, the present project will be implemented using Python and the OpenCV library, due to the large amount of documentation available and the wide scope in terms of its functionalities.

This implementation will consider the tests carried out by the authors of the analyzed literature and the results which they have obtained, as well as their recommendations regarding future work. With this, stands out the importance of a good research phase, since this will help to better understand the approaches to follow, the image processing and training techniques, as well as the statistical formulas to measure and evaluate the classifier's performance.

4. System implementation

This chapter presents the new proposed architecture to be followed considering the alterations on the existent one, refers some software specifications and the used algorithms to perform the automatic detections.

4.1. Wafer failure detection system

Amkor Technology Portugal uses two types of machines to carry out the optical inspections of their wafers: the Camteks and the iFocus. (Camteks are an older machine and iFocus will replace them in a long term.)

Each product is defined by a set of lots and each one has a set of twelve wafers. In the optical inspection, both machines perform a first automatic inspection to each wafer. A set of images and the respective alarms are generated. This data is recorded in a database and will be available on a software (called Offline) to the production line operators so that they can perform the manual classification on each alarm generated previously with the help of the images scanned. All the wafer units without any anomalies registered by the previous inspection are discarded for inspection purposes.

An alarm is characterized by a recipe, the respective lot code, the wafer code, the area and the coordinates of the anomaly in the image, the operator classification, the identification of where the anomaly was found (if were on dies or on vias and the luminosity which can be bright or dark field) and the algorithm used by the machine to identify the anomaly.

For each alarm, Camteks generate two images and iFocus generate four images. They are all the same but were taken using different luminosities. The difference is that Camteks only generate black and white images (using dark field and bright field) and iFocus generates black and white and color images (using dark and bright fields for both colorspaces). These definitions are very important for optical inspections because some anomalies can only be detected with a certain luminosity.

Table 2 shows the example of an alarm and the respective images presented for both machines.

Table 2 - Camteks vs iFocus: images and alarms

<table>
<tr><th>Machine</th><th>Generated images</th><th>Alarm data (example)</th></tr>
<tr><td>Camtek</td><td>(B&W)
Bright field Dark field</td><td>Image sequence ID: 1
Zone: BF_Tight
Area: 225
Width: 9
Length: 47
Algorithm: Clustered
Col: 78
Row: 18
X: 688
Y: 1614
Classify: Good</td></tr>
<tr><td>iFocus</td><td>(B&W)
Bright field Dark field
(Color)
Bright field Dark field</td><td>Image sequence ID: 247
Zone: BF_Relax
Area: 2167
Width: 49.5
Length: 56.2
Algorithm: ImgSub - STD3.0_THLD60 - Black
Col: 5
Row: 13
X: 2957
Y: 4934
Classify: Particles under Cu</td></tr>
</table>

The problem is that a huge number of false alarms is generated, and operators spend much time classifying images that don't have any anomalies, which becomes a very tiring activity and affects their productivity.

The main purpose of this project is to create a solution that can scan all the alarms and the respective images generated by the machines automatically and filter at least 50% of these false alarms, contributing to increase the productivity and the quality of the operator's inspections.

4.2. Solution design

In the production line, after passing through the fabrication processes, each lot with twelve wafers is transported to the Optical Inspection zone. There, the Camtek or the iFocus pulls the wafers to its inside automatically and the first optical inspection is processed. After this process, the wafer is placed again in the lot’s carrier automatically and all the data generated (images and alarms) is stored in a database and is identified by recipe ID, lot ID and wafer ID.

Each generated alarm is shown to the operator with the respective images through a software, where are processed the manual inspections. This software updates the data in the database and add the field with the operator inspections results. Then, this data is exported to .csv files and to .jpg images and stored in a Microsoft Shared Folder for approximately a month. Figure 22 shows the flow of this process, currently operated in the company.

Process “As Is”

Figure 22 - Process "As is"

As explained above, the main objective is to reduce the huge quantity of false alarms that operators must inspect, so the proposed architecture for this solution is demonstrated in Figure 23.

Process “To Be”

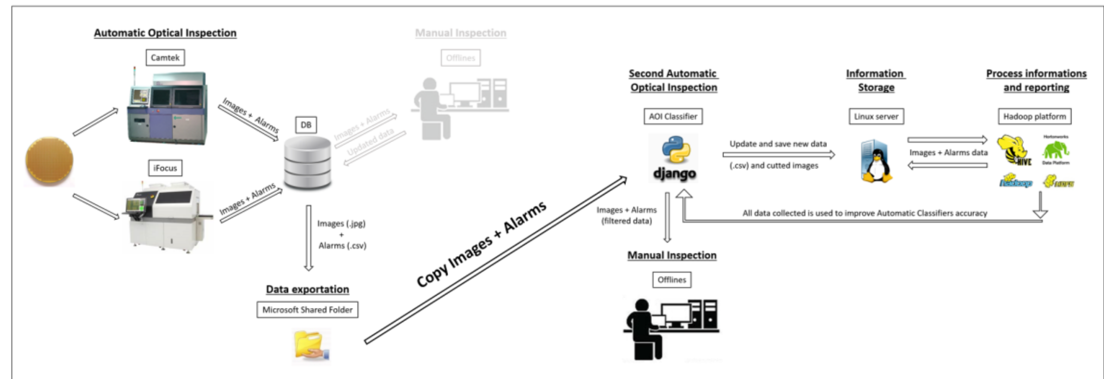


Figure 23 - Process "To be"

In the proposed architecture, the original process is maintained but before the operator execute the manual inspection, all the alarms and respective images are processed in the web portal called AOI Classifier, developed in this project’s implementation, where an automatic classifier runs through all the images and their alarms, doing a second Automatic Optical Inspection, to filter at least 50% of the false alarms given by the machines first inspection. With this, the operators must only inspect the alarms that could not be inspected automatically.

All the relevant information of this process (images cut by the anomaly area given by the alarm, the original alarm data and some additional information generated by AOI Classifier portal like the classification given by the automatic inspection) is stored in a Linux server and will serve to improve and train new iterations of the automatic classifiers and to generate reports to help in the tool’s performance evaluation and improvement.

To reduce the complexity of the problem, in an early stage, the tests will be implemented considering three products (AOI1DL2-AADI-X-P05, AOI1DL2-PM8953-X-P02, AOI1DL2-PM8937-X-P06), one step (AOI1DL2) and four detractors (Good, Particles under Copper, Copper Defect Damage and Contaminations) because these are the most frequent and easily detectable anomalies. The data used will be copied from the current repository so there is no productive impact on the tool’s testing phase.

At a later stage, this implementation will have a productive impact because the AOI Classifier will process the inspections autonomously (although with the supervision of people who will be responsible for the process).

4.3. Architecture

As mentioned in the previous section, the data obtention in AOI Classifier portal comes from the inspections results performed by the AOI machines (Camtek and iFocus). All data processed in the portal is copied and transferred to a server (Automatic Defect Classification (ADC) Server) where additional data is generated on the performed automatic inspection alarms, such as images for the training and test repositories of the automatic classifiers and data that will help in its evaluation through comparisons with the manual inspections already carried out on the selected lots. Even if this information is currently copied to the ADC Server, in the future, this information will also feed the productive Oracle databases used in the original process.

In addition, the AOI Classifier portal also has a manual classification interface that in the future may complement the gaps of the automatic inspection, and pages to consult data reports that allow a careful evaluation of the automatic classifiers status and what needs to be corrected in the next iterations and where the new iterations of automatic classifiers are trained using pattern analysis and recognition mechanisms. Figure 24 shows the architecture design implemented.

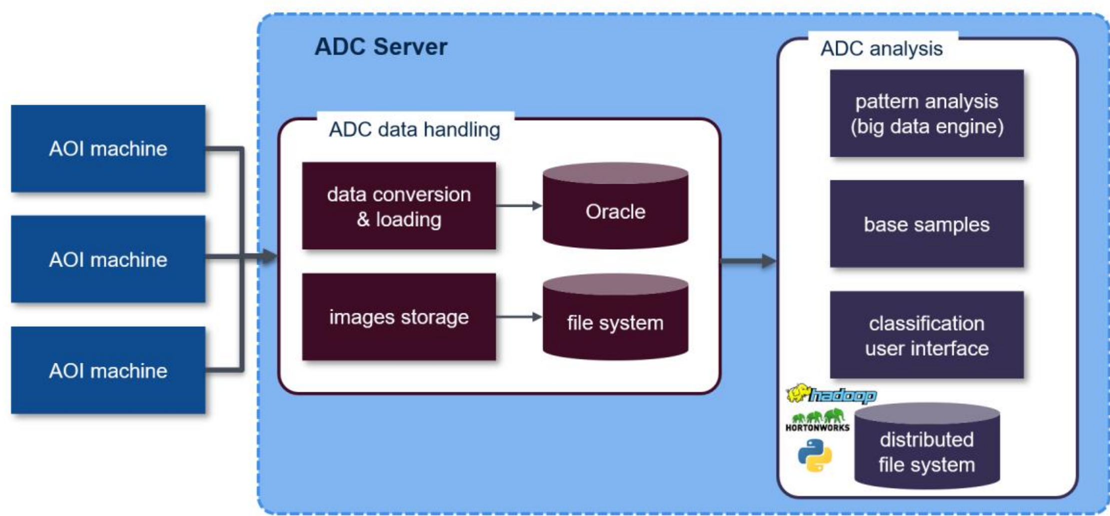


Figure 24 – Architecture

4.4. Software specification

The developed system uses the software:

Table 3 - Software specification

	Name	Version	Observations
Operative System	CentOS	6.9	• Linux distribution; • Provides access to Hortonworks Data Platform software.
Image Processing library	OpenCV	3.0	• Development of applications in the field of Computer Vision; • Application of Image filters, camera calibration, object recognition, structural analysis and others.
Programming languages	Python	2.7	• High level programming language; • Runs OpenCV and Django.
	Django	1.11	• Free and open source framework for creating web applications written in Python.
Storage and data processing	Hortonworks Data Platform	2.6	• Virtual machine for personal desktop to start learning, developing, testing and experimenting with new features - including Hadoop and Apache services like Hive. • Store the processed data in the portal; • Portal's Data processing; • Supports the training process of the new classifiers.

4.5. Implementation

This section describes the used algorithms for the automatic detection and classification.

4.5.1. Used algorithms for detection and classification

The implemented classifiers consist of cascade classification methods based on Haar-like features for object detection.

Haar-like features consider adjacent rectangular regions at a specific location in the detection window, sums up the pixel intensities of those regions, and calculate the difference between these sums, which is then used to classify the image subsections. These features, proposed by *Viola and Jones* [68], are used in the object detection phase, where the target size window is moved over the input image, and for each sub-section the *Haar-like feature* is calculated and then compared to a learned limit that makes the separation between non-objects and objects. This kind of detection requires many *Haar-like features* to describe an object accurately, so they are organized into something called a cascade classifier to form a strong object detector.

The implementation of the automatic classifier integrated in a web portal called AOI Classifier was developed using the *opencv_traincascade* application, which is responsible for training a cascade of a boosted classifier for a given set of samples.

The features used in classifiers are specified by its shape, position within the region of interest, and scale. So, a simple rectangular *Haar-like feature* can be defined as the difference of the sum of the pixels of areas within the rectangle (at any position and scale within the original image).

Methods available in the OpenCV library for image processing are also included in the implementation of this solution, such as thresholds, template matching, among others.

5. Results

This section will explain the experiences carried out to evaluate the first tests phase results performed on the automatic classifiers, as well as the results of the feedback given by users about the release of the AOI Classifier portal’s first version on the production line. As mentioned above, to simplify the process of obtaining results and drawing conclusions, the automatic classifier will initially focus its implementation on three products and four detractors (defects): Contaminations, Particles under Copper, Copper Defect Damage and Good.

5.1. Experiences carried out

While the Camteks and the iFocus scan the Wafers, they detect possible areas of anomalies, which generate alarms and process images to highlight possible failures. This detection is performed by comparing the reference images (existing for each product and representing a template of the wafer’s components disposition) with the shots taken by the machine during the scan, calculating deviations between both images and recognizing as defect every image that presents a significant deviation. Each product can vary in scale, components arrangement and its shape.

AOI Classifier uses data from the inspections repository that stores information with about one month old.

In a first randomized test, about 300 samples were selected for each detractor of the three products to be tested and an automatic classifier was trained. Then, the AOI Classifier portal’s first version was deployed on the production line, where process technicians inspected some lots and the tool compared the ratings given by the operator’s manual inspection, the automatic classifier and the process technician.

These comparisons are made automatically by the portal and generate some labels as True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN), as indicated by the confusion matrix's cross table in chapter 3 (3.5.1 - Confusion matrix). With these data, in addition to calculating the False Positive Rate, False Negative Rate and Accuracy (or Effectiveness) rates, it is also allowed to calculate the remaining rates considered for the classifiers performance evaluation, chosen together with the project’s stakeholders focusing on the company's needs and concerns. The data from the first evaluation performed on the automatic classifiers is shown in Table 4.

Table 4 - Reclassification's general information

	Automatic Classifier (V1)		Automatic Classifier (V2)	
COLLECTED DATA	**Operators**	**Auto Classifier**	**Operators**	**Auto Classifier**
Total occurrences	1977		1738	
% of correct classifications	$\frac{1894}{1977} \approx 95.80\ \%$	$\frac{1292}{1977} \approx 65.35\ \%$	$\frac{1713}{1738} \approx 98.56\ \%$	$\frac{1056}{1738} \approx 60.76\ \%$
% of overkill	$\frac{4}{1977} \approx 0.20\ \%$	$\frac{24}{1977} \approx 1.21\ \%$	$\frac{8}{1738} \approx 0.46\ \%$	$\frac{603}{1738} \approx 34.70\ \%$
% of underkill	$\frac{49}{1977} \approx 2.48\ \%$	$\frac{648}{1977} \approx 32.78\ \%$	$\frac{6}{1738} \approx 0.35\ \%$	$\frac{14}{1738} \approx 0.81\ \%$
% of disagreement	$\frac{30}{1977} \approx 1.52\ \%$	$\frac{13}{1977} \approx 0.66\ \%$	$\frac{11}{1738} \approx 0.63\ \%$	$\frac{65}{1738} \approx 3.74\ \%$

After one month, the automatic classifier was recalibrated using samples that were not correctly classified during that time.

By analyzing Table 4, we can see that from the first version of the automatic classifier for the second version the percentage of correct classifications (where the automatic classification agreed with the process technician) decrease in a range of 4.59%. This happens because the automatic classifier is trying to predict the differentiation between different types of anomalies, which in terms of results is not necessarily negative.

In addition, the overkill and underkill rate has been reversed, which is quite positive, since the overkill rate is given by the quantity of samples classified by the process technician as “Good” and classified as an anomaly by the automatic classifier (false positive alarms) and the underkill rate is given by the quantity of samples classified by the process technician as anomalous and classified as “Good” by the automatic classifier (false negative alarms). This means that the probability of classifying an image with anomalies as "Good" is much lower than the probability of classifying an image without anomalies as anomalous (about 33.89% lower).

The disagreement rate has increased a little, which means that we are able to classify more images with anomalies, but we still cannot distinguish well from each other. However, throughout this experience, it was found that this would not be a good approach to consider, since we could not know for sure where the problem was.

Thus, it was defined a careful set of iterations to be performed to evaluate the automatic classifiers performance, where experiments are performed for a fixed number of lots for each product and each iteration is performed under the same test conditions and parameters. A target was defined for each metric to be used and to define that all iterations that revealed values below that target would be discarded.

After some discussions with the company's intern stakeholders the metrics defined as the most important for the company's quality requirements were the Effectiveness (also called Accuracy), Efficiency, Overkill rate, Underkill rate, number of mismatched classifications, and rate of non-reclassified samples even though metrics such as False Positive Rates, True Positive Rates among others were considered to statistical purposes. Table 5 shows the target values for each of the chosen metrics.

Table 5 - Metrics targets

Targets		Formulas
Efficiency	80 %	1 - ($\frac{\sum NR}{Total}$)
Effectiveness	50 %	$\frac{\sum TP + TN + TPr}{Total}$
Overkill	< 14%	$\frac{\sum FP}{Total}$
Underkill	< 3%	$\frac{FN}{Total}$

Where,

NR are the samples not reclassified by the automatic classifier;

TP are the samples classified as fail by operator and automatic classifier;

TN are the samples classified as good by operator and automatic classifier;

TPr are the samples classified as fail by operator and automatic classifier but the detractors mismatch;

FP are the samples classified as Good by operator and as Fail by the automatic classifier;

FN are the samples classified as Fail by operator and as Good by the automatic classifier;

To keep test parameters equal in all iterations, for each product being tested 13 lots were selected. During the last four months this set of 13 Lots has been updated, so that we can verify that the results are maintained in different types of lots for the same product.

During the different iterations made to the automatic classifiers the following considerations were followed:

- The automatic classifier results will be compared with the operator's classifications so that we can have a classification reference point;
- For a given set of samples, a part of these would be used for training and another part would be used to test the classifiers;
- In each classifier's recalibration, the samples of the previous iteration and the samples referring to situations where the automatic classifier was not able to assign a classification are considered.
- The choice of an automatic classifier as being the best is done by comparing the obtained metrics in the iterations made over time.
- Each iteration will also have a pie chart associated which helps to easily understand what needs to be improved in the next automatic classifiers iterations being tested and the meanings of the colors illustrated in this pie chart are explained on Table 6.

Table 6 - Pie chart legends

Color	Description	Operator result	Automatic Classifier result
(pink)	Not reclassified per Automatic Classifier failure	Good	Not reclassified
(red)	Not reclassified per training	Failure not trained (yet)	Not reclassified
(yellow)	Mismatch (F | G)	Fail	Good
(light blue)	Mismatch (G | F)	Good	Fail
(green)	Correct reclassifications	Good	Good
(dark blue)	Not reclassified per Automatic Classifier failure (F | NR)	Fail	Not reclassified
(ochre)	Mismatch (F1 | F2), with F1 ≠F2	Fail1	Fail2

To make the image repositories homogeneous in terms of luminosities, a test was performed where the bright field and the dark field samples were separated into two different repositories, which gave rise to two classifiers for each one of the detractors.

According to the new conditions defined for the tests, 500 different images were used for each one of the luminosities and for each one of the detractors to train new classifiers. So, for example, to train the detractor "Good" were used 500 bright field images and 500 dark field images. Since the products AOI1DL2-PM8937-X-P06 and AOI1DL2-PM8953-X-P02 are very similar in terms of scale and components dispositions, the samples used to perform this test

belong 50% to product AOI1DL2-PM8937-X-P06 and the others 50% belong to product AOI1DL2-PM8953-X-P02.

In addition, the third version of the automatic classifier (V3) uses about 30% of the selected samples to be trained and the remaining 70% are used to test the classifiers, so that, 18192 images were selected, of which 5504 were used for training and 12688 were used for testing the classifiers for this product. And the fourth version of the automatic classifier (V4) uses about 40% of the selected samples to be trained and the remaining 60% are used to test the classifiers, so that, 18192 images were selected, of which 7280 were used for training and 10912 were used for testing the classifiers. To verify that the new classifiers are acceptable for both products considered, the data analysis will be separated by product.

Table 7 shows the performance evaluation results of the new trained classifiers for product AOI1DL2-PM8937-X-P06, for two different iterations of the automatic classifiers, where the first column (Lots of P06 QND product) shows the sample's actual values, in the second column (Average) the sample's average values and in the third column (Standard deviation) are shown the obtained values for the sample's standard deviations. The column containing the graphs shows the mismatches distribution (inconsistency between operator inspection and automatic classifier), not reclassified samples (where the automatic classifier could not assign any classification) and the classifications considered correct compared to the classifications performed by operators in manual inspection.

Table 7 - Performance evaluation for product AOI1DL2-PM8937-X-P06

Overall statistics	LOTS of P06 QND product		Average		Standard deviation		Charts	
	AC V3	AC V4	AC V3	AC V4	AC V3	AC V4	AC V3	AC V4
Total samples	10682							
Total Mismatches	2305	2338	177	180	76	60		
Total Not Reclassified	3421	3010	263	232	165	130		
Efficiency (%)	67.97	71.82	69.94	73.08	9.88	8.08		
Effectiveness (%) (Accuracy)	47.02	50.54	49.14	51.18	9.67	7.64		
Overkill (%)	18.47	18.62	18.01	19.15	4.05	5.11		
Underkill (%)	2.48	2.66	2.39	1.34	2.51	1.35		

Legend:

From an iteration (V3) of the automatic classifiers to the other (V4) it is possible to verify that the mismatches number, that are the number of classifications that did not correspond between the automatic classifier and the operator increased in 33 samples, therefore, this is a value that is not considered alarming, because the number of samples that the automatic classifier could not assign a classification has decreased 411 units, which means that in these 411 samples, 378 were correctly classified in the second iteration (V4).

Overkill and Underkill rates have remained in the same range of values, however, the fact that the overkill rate is higher than the underkill rate is a good thing, since this means that the automatic classifiers have rated as "Good" only about 2% of the samples that should be classified as having anomalies.

It is also verified that from one iteration to the other (V3 to V4) the efficiency rate increased by 3.85% and the Effectiveness rate, also called Accuracy, increased by 3.52%, which is a good result since in this way we can reach the target for the Effectiveness rate and we are closer to reach the target for the efficiency rate. In contrast, 18% of samples classified by operators as "Good" are being classified as having anomalies by the automatic classifier.

An analysis carried out on the samples tested indicated that these values can be influenced by the criteria change in the catalog of acceptance and rejection in the optical inspections that can happen dynamically and in very specific situations. For example, if in a wafer, for some reason the tape (that in some steps covers the wafer) is not totally removed, in terms of automatic optical inspection there will be detected anomalies in the images. However, in the manual inspection, the operator with the process technicians can realize that the type of residues in the images does not constitute an anomaly, but rather that some process in a previous step on the production line has failed.

Moreover, we may consider that the underkill rate is within the targets and the overkill rate is slightly above. However, these values could be improved along the iterations made to the automatic classifiers, or through the signaling of lots that are classified with dynamic criteria in the production line, because if we consider the values of standard deviation, we conclude that for some lots the overkill rate value lies within the targets.

Considering the average values obtained in the test sample for the product AOI1DL2-PM8937-X-P06, it is verified that the values of the Effectiveness and Underkill rates are within

the defined targets and that the efficiency rate is 6.92% of reaching the target and the Overkill rate exceeded the target value by 5.15% and therefore it can be concluded that despite everything, there was a positive evolution of the results from the iteration V3 to the iteration V4.

Similarly to the previous data, the same test was performed for the AOI1DL2-PM8953-X-P02 product, since the purpose of this test was to realize the extent to which it would make sense to separate the automatic classifier's repositories by product.

Table 8 shows the performance evaluation results of the trained classifiers for product AOI1DL2-PM8953-X-P02.

Table 8 - Performance evaluation for product PM8953-X-P02

Overall statistics	LOTS of P02 QND product		Average		Standard deviation		Charts	
	AC V3	AC V4	AC V3	AC V4	AC V3	AC V4	AC V3	AC V4
Total samples	7510							
Total Mismatches	946	1048	73	81	56	63		
Total Not Reclassified	1807	1662	139	128	169	153		
Efficiency (%)	75.94	77.87	73.11	75.26	17.05	15.53		
Effectiveness (%) (Accuracy)	63.35	64.36	59.24	59.86	18.93	17.13		
Overkill (%)	9.43	10.55	10.76	11.62	4.60	4.74		
Underkill (%)	2.68	2.96	2.91	3.25	1.28	1.35		

Legend:

NR per AC failure; NR per training; Mismatch (F | G); Mismatch (G | F); Correct reclassifications; NR per AC failure (F | NR); Mismatch (F1 | F2), with F1 ≠F2

In the obtained results from the test performed in the product AOI1DL2-PM8953-X-P02, it is verified that, the number of mismatches increased 102 units from iteration V3 to V4 and the number of samples given as not reclassified decreased by 145 units, which means that, in iteration V4 the automatic classifier got the correct classification of more 43 samples than in the iteration V3 which resulted in a 1.93% increase in the efficiency rate and a 1.01% increase in the Effectiveness rate.

It is also verified that the value of efficiency and effectiveness rates are higher for AOI1DL2-PM8953-X-P02 than for AOI1DL2-PM8937-X-P06, where the efficiency rate in iteration V4 increased 6.05% and Effectiveness increased by 13.82% in this second product.

According to the targets defined and taking into account the average values obtained for the iteration V4, it is considered that the efficiency rate needs only 4.74% to reach the target, the Effectiveness, overkill and underkill rates are within the target values, even though the underkill rate passes 0.25%, which is not a bad sign, since the registered value for its standard deviation was only 1.35% which means that for some lots this value can be around 1.90%.

Considering the obtained results, it can be concluded that these automatic classifiers can also be used for the product AOI1DL2-PM8953-X-P02 and that there has been a (although small) evolution from the iteration V3 to the iteration V4.

Looking at the reference images and shots of the automatic inspections, the products chosen for testing appeared to be very similar even though AOI1DL2-PM8937-X-P06 and AOI1DL2-PM8953-X-P02 were more similar than AOI1DL2-AADI- X-P05.

Therefore, the same classifiers used previously were used to test 13 Lots of the AOI1DL2-AADI-X-P05 product so that it can be verified through evidence of data whether it is necessary to separate the training repositories by products or if some products can be trained together, such as was verified in the two previous analyzes.

Table 9 (below) shows that in iteration V3 the obtained results for the AOI1DL2-AADI-X-P05 product were quite different from the products tested previously. By direct observation of the graph generated in the iteration V3, it was concluded that the automatic classifier failed especially in the samples where the operator classified as Good. So, instead of testing a new iteration of this classifier for this product, a new strategy was thought that could bridge these faults in samples classified as Good.

Therefore, a separated automatic classifier was created for this product for the detractor Good because this was the major problem shown by data (keeping the other detractors classifiers only separated by brightness and the same for all products), but, instead of considering images from the automatic batch as was made before, were considered samples taken from the reference image of this product to test an approach that can simplify the training process of the detractor Good.

In addition, a cut-off strategy of the reference image was defined for two different types of scales, since the reference image of this product contained fewer components than

previously tested products, which meant that they were arranged on a larger scale in the image. First, the reference image was divided into 16 squares of equal size, and then the same image was divided into 64 squares of equal size, where each square corresponded to one sample for the repository. The process was repeated to the brightness of bright field and dark field.

Thus, were used a total of 160 training images for the detractor Good and 5460 training images for the remaining detractors and were used 10797 samples (corresponding to 13 Lots) to test this new classifier for this product.

Table 9 shows the performance evaluation results of the trained classifiers for product AOI1DL2-AADI-X-P05.

Table 9 - Performance evaluation for product AADI-X-P05

Overall statistics	LOTS of AADI QND product		Average		Standard deviation		Charts	
	AC V3	AC V4.1	AC V3	AC V4.1	AC V3	AC V4.1	AC V3	AC V4.1
Total samples	10797							
Total Mismatches	1117	968	68	75	72	89		
Total Not Reclassified	6923	2893	440	220	459	153		
Efficiency (%)	64.12	73.21	49.95	69.72	7.76	7.25		
Effectiveness (%) (Accuracy)	25.03	65.34	42.06	61.36	8.61	7.48		
Overkill (%)	9.13	7.52	9.08	6.59	3.34	2.21		
Underkill (%)	0.88	1.26	3.23	1.75	2.42	0.96		

Legend:

NR per AC failure; NR per training; Mismatch (F | G); Mismatch (G | F); Correct reclassifications; NR per AC failure (F | NR); Mismatch (F1 | F2), with F1 ≠F2

Through the observation of the generated graphs, there is an enormous evolution between the iteration V3 and the iteration V4.1. This is because the AOI1DL2-AADI-X-P05 product presents some particularities in relation to the other two products previously tested, namely in the reference image's scale and the components arrangement in the image, which proves that for this product, using reference images to create training samples for the detractor Good from the reference image is a good strategy.

Table 9 shows also that from iteration V3 to V4.1 the number of mismatches decreased by 149 units and the number of samples not reclassified decreased by 4030 units, which is an excellent result. With this new strategy, it was also possible to increase the efficiency rate by 9.09% and the effectiveness rate by 40.31%. The overkill rate has dropped 1.61% and is about 10% lower than in the previously tested products. The underkill rate increased by 0.38%, but since it has a standard deviation of 0.96%, this is also considered a good value.

Regarding the targets analysis and considering the obtained average values, it is verified that the Effectiveness, Overkill and Underkill rates are within the defined targets and that the Efficiency rate requires only 10.28% to reach the desired target, which means that in the short term this value can be improved through new classifiers iterations.

Therefore, we can conclude that, for the AOI1DL2-AADI-X-P05 product, a specific automatic classifier should be used, at least for the Good detractor.

5.2. User's feedback

Initially, the portal was available to the process technicians, where they could test the features, ask questions and curiosities, as well as make some suggestions for improvement. For this, a meeting was held to present the portal to users, explaining the main processes to handling it.

Firstly, the process technicians referred the importance of portal's data processing higher speed, because initially the image processing techniques were calculated and shown to user in the main window, which significantly affected the portal's speed and performance since the images only appeared after processing all calculations and techniques.

So, it was decided to perform this process in the final part of the AOI Classifier portal's data flow. Therefore, the portal does the processing that has to be done internally, not causing waiting times for the user.

Other factors that were highlighted were the lack of some buttons and functions, as the one to consult the previous inspected images after a reclassification and a function that disable the necessity of a re-authentication after the reclassification of an entire wafer. These buttons were immediately added, so in the case of the system authentication, this must be re-validated only every time the user closes the browser, that is, if he does not stop working on the

portal continuously, there is no need to do login again. Moreover, there was no further indication of improvement or correction by users, as they themselves stated.

5.3. Results conclusions

According to the tests carried out, it is concluded that it is important to separate the samples referring to the dark field and bright field luminosities in different repositories for the training of each one of the detractors. Furthermore, the results obtained show that there is an improvement in Accuracy and in the other metrics considered when the training repositories are separated by products with similar characteristics, namely having similar component arrangement and reference image’s scale.

For the AOI1DL2-AADI-X-P05 product, the strategy of using reference image cuts for training the "Good" detractor revealed a significant improvement in results. In the remaining tested products, this strategy will not be applied in the same way, since the AOI1DL2-PM8937-X-P06 and AOI1DL2-PM8953-X-P02 products differ in size, which affects the scale and the image quality to include in the repositories and affects the results of the automatic classifiers.

It can be concluded that the results obtained through the metrics chosen for the automatic classifiers evaluation and considering the targets previously established with the metrology department, are quite satisfactory and show evolution throughout the iterations, which means that this approach will serve to answer the problem that gave rise to this dissertation.

In addition, through the obtained results by the chosen metrics and with the help of the pie chart, it is easy to perceive the needs to be considered on the next automatic classifier’s iterations, as this one can give an overall about the faults and the correct reclassifications.

For example, if there were many samples where the automatic classifier result was “Not Reclassified" and the operator’s result was "Good", by visualizing the chart, we can quickly perceive that the detractor that needs to be improved is the "Good ". On the other hand, if there are many mismatches, it means that we should invest in the iterations of the failure detractors like Particles under Copper, Copper Defect Damage or Contaminations and not in the detractor “Good”.

6. Final considerations

This chapter reports the final conclusions reached at the end of this dissertation, as well as the suggestions for future work enhancements.

6.1. Final conclusions

Automatic pattern detection is an area of interest across multiple industries. As seen in chapter two of this dissertation, there are several projects related to pattern recognition in areas such as medicine, information security, biometrics and in industries such as semiconductors, among others.

The main objective of this dissertation was to study several techniques for automatic object detection, so that they could be applied on the wafer's automatic anomaly detection. This way, it is considered that the main objective has been fulfilled, since an infrastructure was in fact built, capable of demonstrating the operation of an automated classifier that has been implemented and that can help process technicians and operators in their reclassifications.

In a first phase, the results were satisfactory, but the time interval is still a bit short for making final decisions. It is also concluded that this approach, if well trained and well recalibrated, can obtain even better results.

As demonstrated, the proposed architecture can be integrated in the current inspections process easily and for testing purposes, it is possible to release the AOI Classifier tool in production line without affecting the productive results.

With this solution, it will be also possible to expand the portal's usefulness to help operators in their training since the tool can be easily scaled to a test strand for operators to improve their technical knowledge.

Thus, one of the goals to be fulfilled with this portal's development was the implementation of an infrastructure that could be easily improved if it later evolved into another way of thinking and, above all, that the portal allowed its beneficiaries to carry out complex processes without programming knowledge needed. It led to the implementation of the admin users possibility to recalibrate an automatic classifier through, for example, two clicks on the mouse.

The major difficulties encountered on this project's development were undoubtedly the fact that we are dealing with a huge variability of parameters that can be associated with the defects automatic reclassification, since the consideration of an object in a snapshot as a defect depends on uncountable factors, whether in size or position, the intensity of brightness that is used in capturing the snapshot also influences the defect's visibility, since the semiconductor area is in fact one that involves the need to have a lot of knowledge in the most diverse areas (for example: physics, chemistry, optics, mathematics, engineering, etc.).

In addition to all this variability there is still similarity between defects of different types that sometimes, the images processed induce people in error, which is also a constant challenge in the training process of the automatic classifiers.

Therefore, with the AOI Classifier portal it was created not an end, but a starting point for a robust and efficient solution to detect and classify defects in wafers automatically. If desired, its implementation can also be expanded for detections of other objects types, such as faces recognition, to create a new kind of authentication, for example.

6.2. Future work

In terms of future work, it is concluded that there's not a unique way, but many other ways to lead to better results. However, it is suggested to continue this project in order to improve the automatic classifier's results, including more parameters and their combination to extract the defect with more accuracy and with more factors of differentiation and, if necessary, choose another algorithm which can be adapted in the created infrastructure and that could be revealed as a most appropriate to the needs in terms of parameter changes and the products specificity, combining this practice with the most appropriate image processing techniques to remove noise from images and improve the new classifiers accuracy.

In addition, it would also be interesting to promote the implementation of an automatic classifier that acts in real time and does not require periodic recalibrations, making it as independent as possible. In the long term, this could make the wafer's reclassification autonomous by evolving the system from a supervised to an unsupervised type of learning. It is sought to reduce underkill and overkill rates, resulting from human intervention in the reclassification process and to improve the accuracy rate of the automatic system.

It is also suggested to add new clusters to improve data processing as the amount of data used to process new classifiers and the portal's complexity increases.

References

[1] B. &. B. Dom, "Recent advances in the automatic inspection of integrated circuits for pattern defects," *V. Machine Vis. Apps.,* 1995.

[2] J. Prateek, D. M. Escriva and V. Godoy, OpenCV By Example, Packt Publishing Ltd, 2016, pp. 99-123.

[3] OpenCV, "OpenCV 2.4.13.4 documentation," opencv dev team, 2017. [Online]. Available: https://docs.opencv.org/2.4/doc/tutorials/introduction/load_save_image/load_save_image.html.

[4] A. Mordvintsev and A. R. K, "OpenCV-Python Tutorials Documentation," 05 November 2017. [Online]. Available: https://media.readthedocs.org/pdf/opencv-python-tutroals/latest/opencv-python-tutroals.pdf.

[5] "OpenCV Python :: Multiple color object detection and tracking," 22 November 2016. [Online]. Available: https://www.youtube.com/watch?v=mT4gmPCnU58.

[6] I. The MathWorks, "mathworks," [Online]. Available: https://www.mathworks.com/discovery/edge-detection.html. [Accessed 2018].

[7] L. Shapiro and G. Stockman, Computer Vision, Prentice-Hall, Inc., 2001.

[8] P. Hough, "Method and means for recognizing complex patterns". United States Patent 3069654, 18 December 1962.

[9] R. O. Duda and P. E. Hart, "Use of the Hough Transformation to Detect Lines and Curves in Pictures," *Communications of the ACM,* vol. 15, pp. 11-15, 1972.

[10] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition,* vol. 13, pp. 111-122, 1981.

[11] A. Rosebrock, "Detecting Circles in Images using OpenCV and Hough Circles," 21 July 2014. [Online]. Available: https://www.pyimagesearch.com/2014/07/21/detecting-circles-images-using-opencv-hough-circles/.

[12] M. Hardwick, "Simple Lane Detection with OpenCV," Medium, 20 March 2017. [Online]. Available: https://medium.com/@mrhwick/simple-lane-detection-with-opencv-bfeb6ae54ec0.

[13] "ex2tron's Blog," 10 October 2017. [Online]. Available: http://pic.ex2tron.top/cv2_min_rect_rect_bounding.jpg.

[14] L. Najman and M. Schmitt, "Watershed of a continuous function," *In Signal Processing (Special issue on Mathematical Morphology),* vol. 38, pp. 99-112, 1994.

[15] S. PK, "Edge Based Template Matching," Codeproject, 5 August 2010. [Online]. Available: https://www.codeproject.com/Articles/99457/Edge-Based-Template-Matching.

[16] A. Rosebrock, "Image Pyramids with Python and OpenCV," pyimagesearch, 16 March 2015. [Online]. Available: https://www.pyimagesearch.com/2015/03/16/image-pyramids-with-python-and-opencv/.

[17] J. Goutsias and H. J. A. M. Heijmans, "Multiresolution Signal Decomposition Schemes – Part 1: Linear and Morphological Pyramids," Centrum voor Wiskunde en Informatica (CWI), Netherlands, 1998.

[18] W. &. Z. Y. &. G. C. &. Z. J. Zhang, "Action Recognition by Joint Spatial-Temporal Motion Feature," *Journal of Applied Mathematics,* 2013.

[19] P. A. Laplante, Comprehensive dictionary of electrical engineering, 2nd Ed. ed., CRC Press, 2005.

[20] B. E. Dom and V. Brecher, "Recent advances in the automatic inspection of integrated circuits for pattern defects," *Machine Vision and Applications,* 1995.

[21] M. H. Bennett , K. W. Tobin and S. S. Gleason, "Automatic defect classification: status and industry trends," in *Integrated Circuit Metrology, Inspection, and Process Control*, 1995.

[22] K. W. Tobin, S. S. Gleason, T. P. Karnowski, S. L. Cohen and F. Lakhani, "Automatic classification of spatial signatures on semiconductor wafer maps," in *Metrology, Inspection, and Process Control for Microlithography*, 1997.

[23] U. S. Patent, "United States Patent [19]". 2000.

[24] T. Mota, "Identification and Quantification of Oncocytes in Microscopic Images," 2014.

[25] P. García-Teodoro, J. Díaz-Verdejo, M.-F. G. and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security,* vol. 28, pp. 18-28, February–March 2009.

[26] H. Bouma, S. Raaijmakers, A. Halma and H. Wedemeijer, "Anomaly detection for internet surveillance," 2012.

[27] A. Ross and A. Jain, "Information fusion in biometrics," *Pattern Recognition Letters,* vol. 24, pp. 2115-2125, 2003.

[28] J. A. K, S. Prabhakar and S. Chen, "Combining multiple matchers for a high security fingerprint verification system," *Pattern Recognition Letters,* vol. 20, pp. 1371-1379, 1999.

[29] T. K. Ho, J. Hull and S. Srihari, "Decision Combination in Multiple Classifier Systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence ,* vol. 16, pp. 66-75, 1994.

[30] L. Hong and A. Jain, "Integrating Faces and Fingerprints for Personal Identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, pp. 1295-1307, 1998.

[31] S. Ben-Yacoub, Y. Abdeljaoued and E. Mayoraz, "Fusion of Face and Speech Data for Person Identity Verification," *IEEE Transactions on Neural Networks,* vol. 10, pp. 1065-1074, 1999.

[32] U. Dieckmann, P. Plankensteiner, R. Schamburger, B. Fröba and S. Meller, "SESAM : A biometric person identification system using sensor fusion," in *Audio- and Video-based Biometric Person Authentication*, Springer, 1997, pp. 301-310.

[33] J. V. Kittler, M. Hatef, R. Duin and J. Matas, "On Combining Classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, pp. 226-239, 1998.

[34] T. Epanchintsev and A. Sozykin, "Processing Large Amounts of Images on hadoop with OpenCV," *CEUR Workshop Proceedings,* vol. 1513, pp. 137-143, 2015.

[35] G. Bradski, "The OpenCV Library," *Dr Dobb's Journal of Software Tools,* 2000.

[36] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker and I. Stoica, "Spark : Cluster Computing with Working Sets," in *HotCloud'10 Proceedings*, 2010.

[37] I. I. M. Alghussein, W. Aly and M. A. El-Nasr, "Anomaly Detection using Hadoop and MapReduce Technique in Cloud with Sensor Data," *International Journal of Computer Applications,* vol. 125, pp. 22-26, 2015.

[38] M. Schulze, M. A. Hunt, E. Voelkl and C. Thomas, "Semiconductor Wafer Defect Detection Using Digital Holography," in *Process and Materials Characterization and Diagnostics in IC Manufacturing*, Proc. SPIE 5041, 2003.

[39] R. C. F. Jr., A. Kanaan and J. M. S. d. M. Gomes, "As ferramentas do Astrônomo," March 2002. [Online]. Available: http://www.telescopiosnaescola.pro.br/ferramentas.pdf.

[40] W. L. Hosch, "Encyclopædia Britannica," 11 August 2009. [Online]. Available: https://global.britannica.com/technology/machine-learning.

[41] R. Schapire, *COS 511: Theoretical Machine Learning,* Princeton, 2014.

[42] R. Kohavi and F. Provost, "Glossary of Terms (Special Issue on Applications of Machine Learning and the Knowledge Discovery Process)," *Kluwer Academic Publishers,* pp. 271-274, 1998.

[43] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 2nd ed., Prentice Hall, 1995.

[44] M. I. Jordan, "Statistics and machine learning," 9 October 2014. [Online]. Available: https://www.reddit.com/r/MachineLearning/comments/2fxi6v/ama_michael_i_jordan/ckelmtt/?context=3. [Accessed 5 November 2017].

[45] L. Breiman, "Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author)," *Project Euclid,* vol. 16, pp. 199-231, 2001.

[46] J. Gareth, D. Witten, T. Hastie and R. Tibshirani, An Introduction to Statistical Learning, Springer, 2013.

[47] J. Gama, *Combining Classification Algorithms,* 1999.

[48] T. M. Mitchell, Machine Learning, McGraw-Hill, 1999.

[49] N. A. Fonseca, "Parallelism in Inductive Logic Programming Systems," 2006.

[50] P. M. d. S. Ferreira, "Aplicação de Algoritmos de Aprendizagem Automática para a Previsão de Cancro de Mama," 2010.

[51] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand and D. Steinberg, Top 10 algorithms in Data Mining, London: Springer-Verlag, 2007.

[52] J. Han and M. Kamber, Data Mining: Concepts and Techniques, 2nd ed., ELSEVIER, 2006.

[53] J. Davis, V. S. Costa, I. M. Ong, D. Page and I. Dutra, "Using Bayesian Classifiers to Combine Rules," Department of Biostatistics and Medical Informatics, University of Madison-Wisconsin, 2004.

[54] D. Michie, D. J. Spiegelhalter and C. C. Taylor, Machine learning, neural and statistical classification (edited collection), New York: Ellis Horwood, 1994.

[55] U. Pompe and I. Kononenko, "Naive Bayesian classifier within ILP-R," in *Proceedings of the 5th International Workshop on Inductive Logic Programming*, Department of Computer Science, Katholieke Universiteit Leuven, 1995.

[56] H. D. Lee, "Seleção de atributos importantes para a extração de conhecimento de bases de dados," PhD thesis, Instituto de Ciências Matemáticas e de Computação, ICMC-USP, 2005.

[57] Bayesialab, "Bayesian Networks," [Online]. Available: http://www.bayesia.com/bayesian-networks-examples. [Accessed 2017].

[58] R. D. Snee, "Graphical display of two-way contingency tables," *The American Statistician ,* vol. 28, pp. 9-12, 1974.

[59] P. Weber and L. Jouffe, "Reliability modelling with dynamic bayesian networks," *IFAC Proceedings,* pp. 57-62, 2003.

[60] Y. LeCun, B. Boser, J. S. Denker, D. Henderson and R. E. Howard, "BACKPROPAGATION Applied to Handwritten," *Neural Computation,* pp. 541-551, 1989.

[61] K. J. Lang, A. H. Waibel and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural Networks,* vol. 3, pp. 23-43, 1990.

[62] G. W. Cottrell, "Extracting features from faces using compression networks: Face, identity, emotion, and gender recognition using holons," in *Connection Models: Proceedings of the 1990 Summer School*, San Mateo, CA, Morgan Kaufmann Publishers, Inc., 1990.

[63] M. Cilimkovic, "Neural Networks and Back Propagation Algorithm," Institute of Technology Blanchardstown, Ireland.

[64] D. Graupe, Principles of Artificial Neural Networks, World Scientific, 2013.

[65] J. A. Hertz, A. S. Krogh and R. G. Palmer, Introduction to the Theory of Neural Computation, Perseus Books, 1990.

[66] J. Gama and P. Brazdil, "Cascade Generalization," *Machine Learning,* vol. 41, pp. 315-343, 2000.

[67] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences,* vol. 55, pp. 119-139, 1997.

[68] P. Viola and J. Michael, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition*, Kauai, HI, USA, USA, 2001.

[69] OpenCV, "Object detection: OpenCV 2.4.13.4 API reference," Opencv dev team, 2017. [Online]. Available: https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html.

[70] T. Raeder, "Model Monitor User's Guide version 1.0," Department of Computer Science and Engineering, University of Notre Dame, 2008.

[71] R. Kohavi and F. Provost, "Glossary of terms," *Machine Learning,* vol. 30, pp. 271-274, 1998.

[72] S. Raschka, "Confusion Matrix," [Online]. Available: https://rasbt.github.io/mlxtend/user_guide/evaluate/confusion_matrix/.

[73] A. J. Viera and J. M. Garrett, "Understanding interobserver agreement: the kappa statistic.," PhD, University of North Carolina, USA, 2005.

[74] Mathworks, "Detector Performance Analysis Using ROC Curves - MATLAB & Simulink Example"," [Online]. Available: http://www.mathworks.com/help/phased/examples/detector-performance-analysis-using-roc-curves.html. [Accessed 2017].

[75] "StatEL: the Excel integrated set of statistical softwares, ROC (Receiver Operating Characteristic) Curve," ad Science Company, [Online]. Available: http://www.adscience.eu/uploads/ckfiles/files/html_files/StatEL/statel_ROC_curve.htm. [Accessed 2017].

[76] Pedregosa, Fabian; Varoquaux, Gaël; Gramfort, Alexandre; Michel, Vincent; Thirion, Bertrand; Grisel, Olivier; Blondel, Mathieu; Prettenhofer, Peter; Weiss, Ron; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research,* vol. 17, pp. 2825-2830, 2011.

[77] A. Tolay, "What is Precision-Recall (PR) curve?," Quora, 3 June 2014. [Online]. Available: https://www.quora.com/What-is-Precision-Recall-PR-curve. [Accessed 2017].

[78] Zentroid, "PCB/CAD/AOI Basics, Component and Package names," 17 May 2013. [Online]. Available: https://en.wikipedia.org/wiki/Automated_optical_inspection.

[79] M. H. Bennett , K. W. Tobin and S. S. Gleason, "Automatic defect classification: status and industry trends," *Integrated Circuit Metrology, Inspection, and Process Control,* 1995.

[80] O. Fadare and G. Tallini, "Atlas of Genetics and Cytogenetics in Oncology and Haematology," 2003. [Online]. Available: http://atlasgeneticsoncology.org/Tumors/OncocytThyroidID5068.html.

[81] K. W. Tobin, S. S. Gleason, T. P. Karnowski, S. L. Cohen and F. Lakhani, "Automatic classification of spatial signatures on semiconductor wafer maps," in *Metrology, Inspection, And Process Control For Microlithography XI*, 1997.

[82] T. M. Mitchell, Machine Learning, McGraw-Hill, 1999.

[83] E. N. Malamas, E. G. M. Petrakis, M. Zervakis, L. Petit and J.-D. Legat, "A survey on industrial vision systems, applications and tools," *Image and Vision Computing,* vol. 21, pp. 171-188, 2003.

[84] "Random Musings," 2014. [Online]. Available: https://dmm613.wordpress.com/2014/12/16/demystifying-artificial-neural-networks-part-2/.

[85] C. Papageorgiou, M. Oren and T. A. Poggio, "General framework for object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 1998.

[86] R. Lienhart, A. Kuranov and V. Pisarevsky, "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object," *Pattern Recognition,* vol. 2781, pp. 297-304, 2003.

[87] F. Crow, "Summed-area tables for texture mapping," in *Proceedings of SIGGRAPH*, 1984.

[88] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in *IEEE International Conference on Image Processing* , Rochester, NY, USA, USA , 2002.

[89] C. Messom and A. Barczak, "Fast and efficient rotated haar-like features using rotated integral images," in *Australian Conference on Robotics and Automation (ACRA2006)*, 2006.

[90] M. Mohri, A. Rostamizadeh and A. Talwalkar, Foundations of Machine Learning, The MIT Press, 2012.

[91] OpenCV, "OpenCV 2.4.13.4 documentation: Basic Thresholding Operations," OpenCV dev team, [Online]. Available: https://docs.opencv.org/2.4.13.4/doc/tutorials/imgproc/threshold/threshold.html. [Accessed 2017].

[92] V. Chandola, A. Banerjee and V. Kumar, "Anomaly Detection : A Survey," in *ACM Computing Surveys,*, University of Minnesota, 2009, p. 1–58.

Appendix

a) AOI Classifier portal user's manual

1. Introduction

This document contains a brief explanation about the main functions of AOI Classifier portal as well as some steps that must be considered to check if the portal is not running as expected.

2. AOI Classifier (Main functions)

a. Login Page

Figure 25 - Portal's login window

The portal's administrator provides the authorized users' access credentials.

b. Lot selection

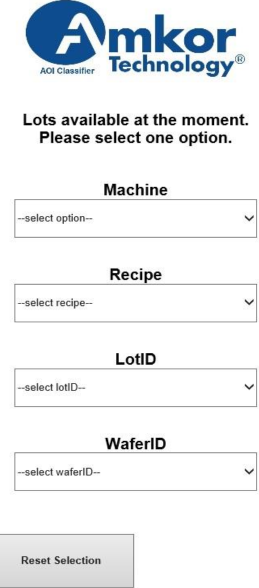


Figure 26 - Portal's information selection window

Each user must select the Machine, the Recipe, the LotID and the WaferID that want to use.

After select all the required information, the button “Submit Results” will be available.

On this page, there are three buttons available in the lower right corner.

Figure 27 - Portal's assistant buttons

The first one allows user to change his password.

The second one is used to consult the classifier’s evaluations.

The third one is used to consult the AOI Classifier documentation.

c. Main Menu

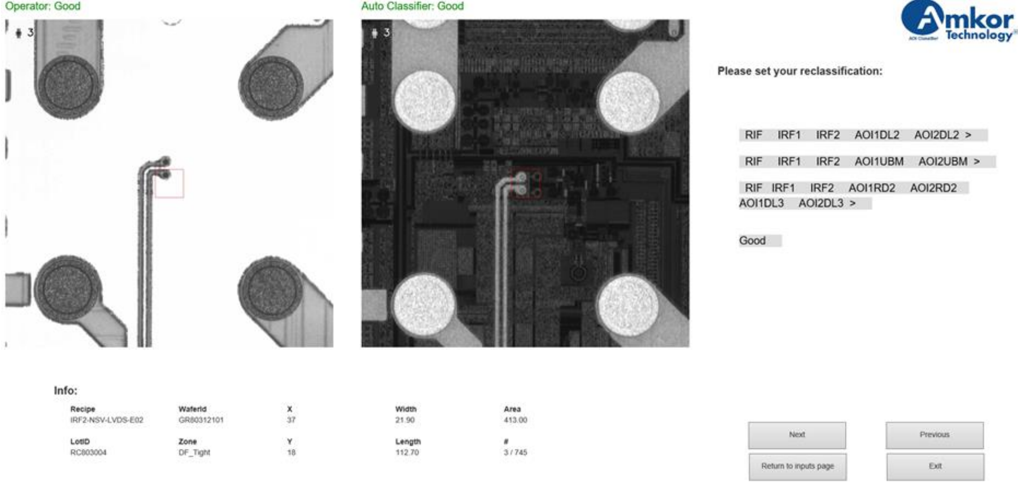


Figure 28 - Portal's main menu window

This menu displays the information related to defects previously classified by the operator, as well as the the images corresponding to this classification according to two different luminosities (bright field and dark field).

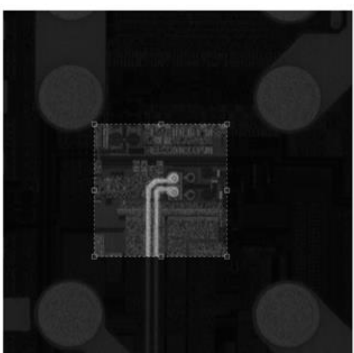
Here, the user can view the area that was selected by iFocus as belonging to a defect (the red rectangle), as well as view the automatic classifier's suggestion of reclassification.

To do this, the user should open the menu on the right side and choose the most suitable detractor.

Figure 29 - Portal's detractor selection menu

After clicking at the detractor chosen, will open a new page that will allow user to select the area correspondent to failure. Here, users must be careful with the area they select.

Then, users must cut the desired area, click "crop" and then send the results through the button "Send to repository".



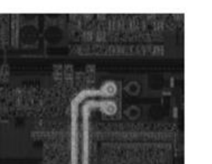


Figure 30 - Portal's create positive samples window

3. **Support**

The support for AOI Classifier during office hours is made:

5 days per week, 8 hours per day and through contact with the portal’s representatives.

b) AOI Classifier samples gallery page

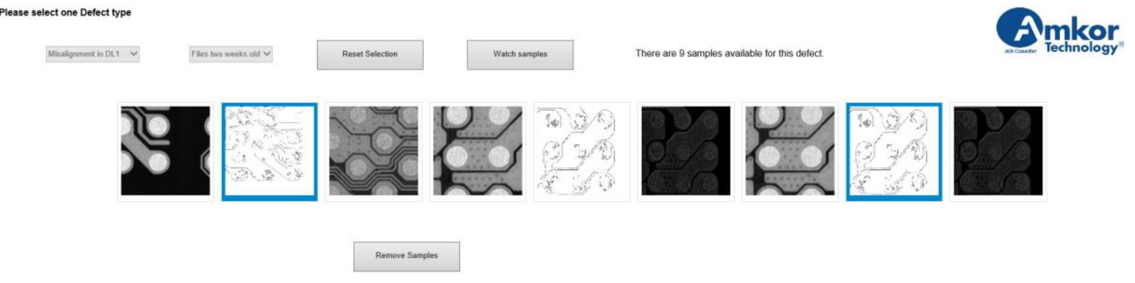


Figure 31 - Portal's image gallery page

c) AOI Classifier recalibration page

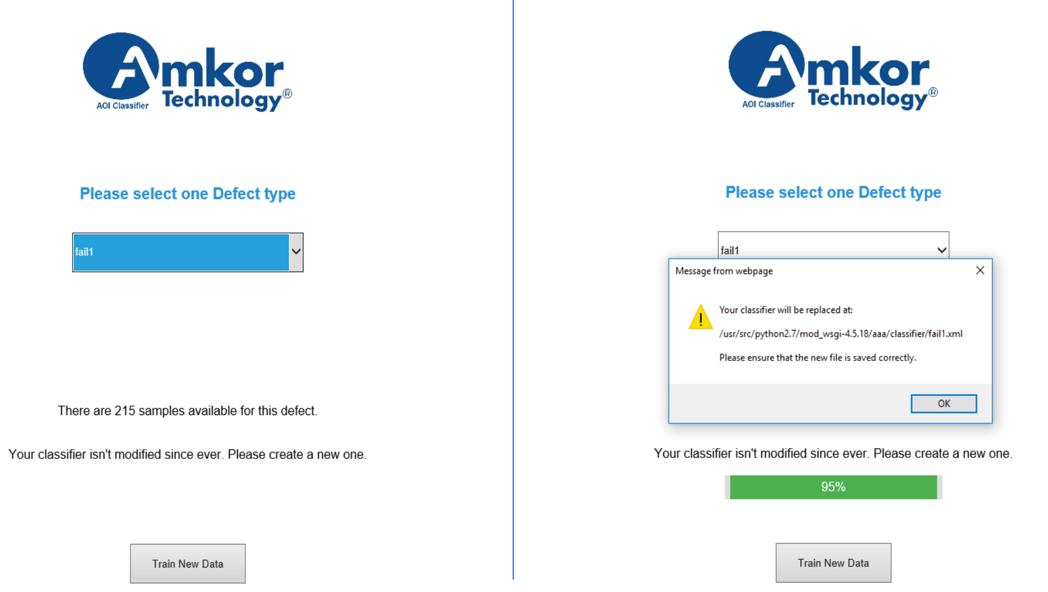


Figure 32 - Portal's automatic classifier's recalibration page

d) Advantages and disadvantages

Thus, after the tests were carried out, a set of advantages and disadvantages inherent to the autonomous and automatic classification approach in the solution is highlighted, as shown in Table 10.

Table 10 -Project's implementation approach advantages and disadvantages

Advantages	Disadvantages
→ Dramatically decrease the time at inspections (Automatic batch currently needs approx. 756ms/per image)	→ Increase of the time consuming required on the first approaches and to select the correct samples to train each detractor - Dedicated to a similar products design (it can always be changed but requires some time of training and tests)
→ Reduce effort of operators and process technicians	→ Necessity of have somebody (maybe not permanently) allocated to inspect the computer's reclassifications (it doesn't inspect as a human being)
→ Reduce human errors and inconsistency between different inspectors	→ Requires very stable and accurate classifier's versions
→ Increase inspections quality and productivity (no pauses, no fatigue, no time of day neither day of week)	→ The autonomous systems don't have real intelligence: they learn based in rules → Low tolerance in differentiating errors that might be considered Pass due to its location
→ Can be easily adaptable and reprogrammable to inspect products with significant differences - Can improve costs in a long-term perspective	→ Parameters and image samples must be carefully defined and adjusted (as much as needed)